НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОБЛЕМ МОДЕЛЮВАННЯ В ЕНЕРГЕТИЦІ ІМ. Г. Є. ПУХОВА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кваліфікаційна наукова
праця на правах рукопису

ВИСОЦЬКА ОЛЕНА ОЛЕКСАНДРІВНА

УДК 004.056.523:57.087.1(043.3)

ДИСЕРТАЦІЯ

МЕТОДИ БІОМЕТРИЧНОЇ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ ІНФОРМАЦІЙНИХ СИСТЕМ ЗА ЇХ КЛАВІАТУРНИМ ТА РУКОПИСНИМ ПОЧЕРКОМ

Спеціальність: 05.13.21 – «Системи захисту інформації»
Галузь знань: 12 – «Інформаційні технології»

Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____О.О.Висоцька

Науковий керівник: **Давиденко Анатолій Миколайович**, кандидат технічних наук, с.н.с., провідний науковий співробітник Інституту проблем моделювання в енергетиці ім. Г. Є. Пухова НАН України

Київ – 2019

АНОТАЦІЯ

Висоцька О.О. Методи біометричної автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.21 «Системи захисту інформації». – Інститут проблем моделювання в енергетиці ім. Г. Є. Пухова НАН України, Національний авіаційний університет, Київ, 2019.

Надійність роботи будь-якої інформаційної системи залежить від засобів захисту, які використовуються для забезпечення захищеності інформації, що зберігається, обробляється та передається в даній інформаційній системі. Одним із способів захисту інформації є використання систем автентифікації. Серед інших типів автентифікації користувачів, найбільш оптимальна це біометрична автентифікація. Існують різні механізми вирішення задачі біометричної автентифікації, серед яких доцільно звернути увагу на нейронні мережі.

Дисертаційна робота присвячена вирішенню задачі автентифікації користувачів інформаційних систем за їх біометричними характеристиками, використовуючи в якості механізму розпізнавання нейронні мережі.

Для цього необхідно розв’язати наступні задачі: проаналізувати існуючи біометричні методи розпізнавання та програмні і апаратні засоби на їх основі, з метою здійснення вибору методів автентифікації користувачів інформаційних систем, застосування яких забезпечить задану імовірність правильного розпізнавання користувачів та не потребуватиме суттєвих витрат на впровадження; розробити методи первинної обробки навчальних даних, використання яких дозволить збільшити імовірність правильного розпізнавання користувачів та зменшити затрачувані ресурси; розробити методи автентифікації користувачів інформаційних систем, які базуються на обраних біометричних методах розпізнавання і використовують для цього обраний різновид нейронної мережі; створити програмне забезпечення, яке на основі розроблених методів розпізнавання користувачів, виконуватиме автентифікацію користувачів інформаційних систем за

обраними біометричними характеристиками, та по-перше, оцінюватиме імовірність їх правильного розпізнавання, по-друге, на основі аналізу накопичених даних даватиме змогу здійснити вибір конфігураційних параметрів систем розпізнавання.

Для вирішення поставлених задач спочатку були проаналізовані існуючі типи автентифікації користувачів інформаційних систем та вдосконалено класифікацію біометричних систем розпізнавання та, на основі данної класифікації, проведено аналіз біометричних методів та систем розпізнавання. За результатами проведеного аналізу, обрані біометричні характеристики, оптимальні для автентифікації, за їх допомогою, користувачів інформаційних систем, а саме клавіатурний почерк та рукописний почерк. Після чтого виконано порівняльний аналіз найбільш поширених методів розпізнавання користувачів за обраними біометричними характеристиками (з відкритих джерел). Для вибору механізму розпізнавання користувачів інформаційних систем, були проаналізовані існуючі види нейронних мереж та обраний найбільш придатний їх різновид для вирішення даної задачі, а саме імовірнісна нейронна мережа.

На основі проведеного аналізу і здійсненого вибору запропоновано та програмно реалізовано метод автентифікації користувачів інформаційних систем за їх клавіатурним почерком, за допомогою імовірнісної нейронної мережі. Під час автентифікації аналізуються часові інтервали між вводом користувачем двох сусідніх символів ключової фрази. Для передачі зразка клавіатурного почерку користувача в комп’ютер, використовувалася клавіатура комп’ютера. Даний метод складається з наступних дев'ятьох етапів: попереднє формування множини ознак клавіатурного почерку користувача; налаштування параметрів, які є найбільш критичними при автентифікації користувачів за їх клавіатурним почерком; накопичення пробної бази даних навчальних зразків, для вибору оптимальної ключової фрази; вибір ключової фрази на основі аналізу накопичених зразків почерку з пробної бази даних; накопичення бази даних навчальних зразків; вибір характеристик, що будуть аналізуватися при автентифікації; попередній відбір навчальних зразків, які будуть використовуватись для розпізнавання; виконання, за необхідністю, відбору за словами навчальних зразків з бази даних навчальних

зразків; виконання автентифікації користувачів за їх клавіатурним почерком.

Важливою складовою даного методу є розроблений метод первинної обробки зразків клавіатурного почерку, використання якого збільшує імовірність правильного розпізнавання користувачів. Цей метод складається з наступних двох етапів: вибір ключової фрази та характеристик, що будуть аналізуватися під час автентифікації (аналіз зразків з пробної бази даних навчальних зразків для вибору набору ключових фраз та символів ключової фрази, динаміка вводу яких буде аналізуватися під час автентифікації); попередній відбір навчальних зразків, які будуть використовуватись для розпізнавання (видалення з бази даних навчальних зразків, зразків з грубими помилками, за допомогою порівняння кожної ознаки з її середньоарифметичним значенням для даного користувача).

На наступному етапі дисертаційних досліджень було запропоновано та програмно реалізовано метод автентифікації користувачів інформаційних систем за їх рукописним почерком, за допомогою імовірнісної нейронної мережі. Основними з ознак зразка рукописного почерку користувачів, що аналізувалися, в даному методі, під час автентифікації, є: координати X і Y та тип точок зображення; тиск, з яким користувач натискає ручкою (чи іншим подібним пристроєм) на сенсорний екран, під час створення точок; інш. Для передачі зразка рукописного почерку користувача в комп’ютер, використовувався графічний планшет, як один з варіантів пристрою з сенсорним екраном, який застосовується для динамічного передавання характеристик рукописного почерку користувача. Даний метод складається з наступних десяти етапів: попереднє формування множини ознак рукописного почерку користувача; налаштування параметрів, які є найбільш критичними при автентифікації користувачів за їх рукописним почерком; формування бази даних навчальних зразків; видалення помилок; умовне розділення зображення ключової фрази на зображення окремих символів; корекція даних, які будуть використовуватись для розпізнавання зразків почерку користувачів; формування множини контрольних точок; виконання першого раунду автентифікації користувачів за їх рукописним почерком (розпізнавання ключової фрази, що написана); первинна обробка зразків рукописного почерку користувачів, яка

необхідна для виконання другого раунду автентифікації за їх рукописним почерком; виконання другого раунду автентифікації користувачів за їх рукописним почерком (розпізнавання стилю написання ключової фрази). Для забезпечення можливості використання, в якості механізму розпізнавання, імовірнісної нейронної мережі та для зменшення затрачуваних ресурсів, в даному методі розпізнавання, під час автентифікації, аналізуються характеристики не всіх точок, а тільки найбільш значимих для кожного символу – контрольних точок. В роботі виділени контрольні точки наступних трьох типів: початкові та кінцеві точки кожної лінії; кутові точки ліній; точки перетинання ліній.

Як і в методі автентифікації користувачів за їх клавіатурним почерком, важливою складовою даного методу також є розроблений метод первинної обробки зразків рукопистного почерку, використання якого збільшує імовірність правильного розпізнавання користувачім інформаційних систем. Даний метод первинної обробки складається з наступних двох етапів, виконання яких необхідно на різних стадіях роботи методу автентифікації: попередній відбір даних, які будуть використовуватись для розпізнавання (видалення помилок п’яти типів); корекція даних, які будуть використовуватись для розпізнавання зразків почерку (виконання посимвольних повороту, зсуву та пропорційного масштабування зображень кожного символу на всю робочу область обраного розміру; необхідність викликана наявністю помилок трьох типів, а саме неправильним розміщенням написаної ключової фрази на робочій області графічного планшету).

Далі було виконано тестування розроблених в роботі методів автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком, з використанням розроблених методів первинної обробки зразків почерку. В якості механізму розпізнавання використовувалась імовірнісна нейронна мережа. Для розроблених методів оцінювалась імовірність правильного розпізнавання користувачів, при різних значеннях критичних конфігураційних параметрів системи розпізнавання. Для даного тестування, на основі розроблених методів, були створені автоматизовані системи; за допомогою створеного програмного забезпечення, накопичені бази даних навчальних зразків клавіатурного та рукописного почерку

користувачів; на основі цих даних проведено ряд експериментів.

До наукової новизни та практичної цінності, які були отримані в результаті дисертаційних досліджень, можно віднести наступне.

Вдосконалено класифікацію біометричних систем розпізнавання, що дало змогу здійснити вибір біометричних методів автентифікації користувачів інформаційних систем, застосування яких забезпечує задану імовірність правильного розпізнавання користувачів та не потребує суттєвих витрат на впровадження.

Вперше запропоновано метод первинної обробки зразків клавіатурного почерку, який за рахунок аналізу спектральних характеристик почерку користувача, дозволяє виключити хибні зразки почерку, які є нехарактерними і викликані випадковими помилками користувача при наборі тексту, що забезпечує більш високу і рівномірну якість характеристик почерку і, завдяки цьому, збільшується імовірність правильного розпізнавання користувачів.

Вдосконалено метод автентифікації користувачів інформаційних систем за їх клавіатурним почерком, який за рахунок використання для розпізнавання імовірнісної нейронної мережі та виконання первинної обробки зразків почерку, збільшує імовірність правильного розпізнавання користувачів.

Вперше запропоновано метод первинної обробки зразків рукописного почерку, в якому за рахунок автоматизації процесу відбору контрольних точок в зразках рукописного почерку, чиї характеристики аналізуються, видаленню помилкових точок п’яти типів та проведенню корекції даних трьох типів, досягається збільшення імовірності правильного розпізнавання користувачів та, завдяки зменшенню кількості ознак в зразках, що аналізуються, зменшення затрачуваних ресурсів.

Вдосконалено метод автентифікації користувачів інформаційних систем за їх рукописним почерком, який за рахунок використання для розпізнавання імовірнісної нейронної мережі та виконання первинної обробки зразків рукописного почерку, збільшує імовірність правильного розпізнавання користувачів.

Створене в роботі програмне забезпечення, на основі розроблених методів розпізнавання людей за клавіатурним та рукописним почерком, на базі імовірнісної нейронної мережі, може використовуватися для автентифікації користувачів в

інформаційних систем; для розпізнавання працівників в системах контролю доступу та обліку користувачів; для визначення аномального стану працівника на підприємствах, на яких для їх нормального функціонування критична наявність у працівників уважності під час роботи.

На основі запропонованого методу розпізнавання користувачів за їх клавіатурним почерком, з використанням методу обробки навчальних даних, створено програмне забезпечення для реалізації біометричної автентифікації користувачів інформаційних систем за їх клавіатурним почерком, яке дозволяє збільшити ступінь багатофакторності автентифікації інформаційних систем, не вимагаючи для цього додаткового обладнання.

На основі запропонованого методу розпізнавання користувачів за їх рукописним почерком, з використанням методу обробки навчальних даних, створено програмне забезпечення для реалізації біометричної автентифікації користувачів інформаційних систем за їх рукописним почерком, що дозволяє збільшити ступінь багатофакторності автентифікації інформаційних систем, при наявності стандартних сенсорних засобів вводу графічної інформації.

На основі результатів проведених експериментів, за допомогою розробленого програмного забезпечення, здійснено вибір конфігураційних параметрів, налаштування яких є найбільш критичним для збільшення імовірності правильного розпізнавання користувачів інформаційних систем та отримана оцінка імовірності правильного розпізнавання користувачів за обраними біометричними характеристиками, яку забезпечує використання імовірнісної нейронної мережі.

Результати дисертаційних досліджень впроваджені в наступних організаціях: в Управлінні верифікації Генерального штабу Збройних сил України (акт від 25.03.2010р.), в управлінні Пенсійного фонду України у Києво-Святошинському районі (акт від 27.05.2010р.), в підприємстві «Інтегратор» (акт від 24.12.2013р.), в ТОВ «НВЦ «ІНФОЗАХИСТ» (акт від 12.12.2018р.), в ІПМЕ ім. Г.Є. Пухова НАН України (акт від 11.01.2019р.), а також використовуються у навчальному процесі кафедри комп’ютеризованих систем захисту інформації Національного авіаційного університету (акт від 29.05.2019р.).

Ключові слова: автентифікація, біометрія, клавіатурний почерк, рукописний почерк, імовірнісні нейронні мережі, розпізнавання, інформаційні системи.

ABSTRACT

Vysotska O. Methods of biometric authentication of information systems users by their keystroke pattern and handwriting. – Qualifying scientific work as a manuscript.

Thesis for a Candidate of Technical Science degree in specialty 05.13.21 «Information security systems». – Pukhov Institute for Modelling in Energy Engineering NAS of Ukraine, National Aviation University, Kyiv, 2019.

The reliability of any information system depends on the security measures used to secure the information stored, processed and transmitted in that information system. One way to protect information is to use authentication systems. Among other types of authentication of users, the most optimal is biometric authentication. There are various mechanisms for solving the problem of biometric authentication, among which it is advisable to pay attention to neural networks.

Thesis is devoted to solving the problem of authentication of information systems users by their biometric characteristics, using as a mechanism of recognition of neural networks.

For this purpose, the following tasks should be solved: to analyze the existing biometric recognition methods and software based on them, in order to choose the methods of authentication of information systems users, the application of which will provide a given probability of correct recognition of users and will not require significant implementation costs; to develop methods of primary processing of training data, the use of which will increase the probability of correct recognition of users and will reduce the resources expended; to develop methods of authentication of information systems users, which are based on the selected biometric recognition methods and use for this purpose the chosen kind of neural network; to create software that, based on the developed methods of user recognition, will authenticate users of information systems according to the selected biometric characteristics, and, firstly, will evaluate the probability of their correct recognition, and secondly, based on the analysis of the accumulated data will make the

choice of configuration parameters of recognition systems.

For the solution of these tasks, the first types of authentication of information systems users were analyzed and the classification of biometric recognition systems was improved and, based on this classification, biometric recognition methods and recognition systems were analyzed. According to the results of the analysis, biometric characteristics, which are optimal for authentication of information systems users, namely keystroke pattern and handwriting, were selected. After that a comparative analysis of the most common methods of user recognition was performed according to the selected biometric characteristics (from open sources). In order to choose the mechanism of recognition of users of information systems, existing types of neural networks were analyzed and the most suitable variant of them was selected for this task, namely probabilistic neural network.

Based on the analysis and the selection, the method of authentication of information systems users by their keystroke pattern, using a probabilistic neural network, is proposed and implemented. During authentication, time intervals between user input of two adjacent characters of a key phrase are analyzed. A computer keyboard was used to send a sample of a user's keystroke pattern to a computer. This method consists of the following nine steps: pre-forming a plurality of features of a user's keystroke pattern; setting the parameters that are most critical when authenticating users to their keystroke pattern; accumulation of a test database of training samples, to choose the optimal key phrase; selection of a key phrase based on the analysis of the accumulated samples of handwriting from the test database; accumulation of a database of training samples; the choice of characteristics to be analyzed during authentication; pre-selection of training samples to be used for recognition; performing, if necessary, the selection by words of training samples from the training samples database; authentication of users on their keystroke pattern.

An important component of this method is the developed method of primary processing of samples of keystroke pattern, the use of which increases the probability of correct recognition of users. This method consists of the following two steps: selecting a key phrase and characteristics to be analyzed during authentication (analysis of samples from a training sample test database to select a set of key phrases and key phrase

characters whose input dynamics will be analyzed during authentication); pre-selection of training samples to be used for recognition (removal from training samples, samples with gross errors, by comparing each feature with its arithmetic mean for a given user).

At the next stage of the thesis research, a method of authentication of information systems users by their handwriting, using a probabilistic neural network, was proposed and programmatically implemented. The main features of the sample handwriting of the analyzed users in this method, during authentication, are: *X* and *Y* coordinates and type of image points; the pressure the user pushes on the touch screen with the handle (or other similar device) when creating points. In order to transfer a handwriting sample of a user to a computer, a graphic tablet was used as one of the touch screen devices used to dynamically convey the characteristics of a user's handwriting. This method consists of the following ten steps: pre-forming a plurality of handwriting features of a user; setting the parameters that are most critical when authenticating users by their handwriting; formation of a database of training samples; debugging; conditional division of the image of the key phrase into images of individual characters; correction of data that will be used to identify patterns of users' handwriting; formation of multiple control points; performing the first round of user authentication by their handwriting (recognizing the key phrase that is written); primary processing of handwritten samples of users that is required to complete the second round of authentication on their handwriting; performing the second round of user authentication for their handwriting (recognizing the style of writing a key phrase). In order to allow for the use, as a recognition mechanism, of the probabilistic neural network and in order to reduction of wasted resources, this authentication method analyzes the characteristics of not all points, but only the most significant for each character, so called checkpoints. The checkpoints of the following three types are distinguished: starting and ending points of each line; angular points of lines; points of intersection of lines.

As with the method of authenticating users by their keystroke pattern, an important component of this method is also the developed method of primary processing of handwriting samples, the use of which increases the probability of correct recognition of information systems users. This primary processing method consists of the following two steps, which must be performed at different stages of the authentication method: pre-

selection of the data to be used for recognition (removal of five types of errors); correction of data to be used to recognize handwriting (performing character rotation, shifting and proportional scaling of images of each character over the entire workspace of the selected size; necessity is caused by the presence of errors of three types, namely incorrect placement of the written key phrase on the work area of graphic tablet).

The testing of the methods of authentication of information systems users using their keystroke patterns and handwriting, using the methods of primary processing of handwriting samples, was further performed. A probabilistic neural network was used as the recognition mechanism. For the methods developed, the probability of correct user recognition was estimated, with different values of the critical configuration parameters of the recognition system. For this testing, automated systems were created based on the methods developed; using the created software, accumulated databases of training samples of keystroke patterns and handwriting of users; a number of experiments were conducted on the basis of these data.

The scientific novelty and practical value gained from the thesis research can be attributed to the following.

The classification of biometric recognition systems has been improved, which made it possible to select biometric methods of authentication of information systems users, the application of which provides a given probability of correct user recognition and does not require significant implementation costs.

For the first time, a method of primary processing of keystroke patterns samples is proposed, which, by analyzing the spectral characteristics of the user's handwriting, eliminates erroneous handwriting samples that are uncharacteristic and caused by random typing errors, which ensures a higher and more uniform quality of handwriting characteristics and, due to this, increases the probability of correct user recognition.

The method of authentication of information systems users by their keystroke patterns has been improved, which, by using probabilistic neural network recognition and performing the primary processing of keystroke patterns samples, increases the probability of correct user recognition.

For the first time, a method of primary processing of handwriting samples is

proposed, in which, by automating the process of selection of checkpoints in handwriting samples, the characteristics of which are analyzed, the elimination of five types of erroneous points, and the correction of three types of data, the probability of correct recognition of users is increased, due to reducing the number of features in the samples being analyzed, reducing the resources expended.

The method of authentication of information systems users by their handwriting has been improved, which, by using probabilistic neural network recognition and performing the primary processing of handwriting samples, increases the probability of correct user recognition.

Software created during the thesis research, based on the developed methods of recognizing people by keystroke patterns and handwriting, based on probabilistic neural network, can be used for authentication of information systems users; to identify employees in access control systems and user account systems; to determine the anomalous status of the employee in the enterprises, which for their normal functioning is critical to the presence of employees mindfulness during work.

Based on the proposed method of recognizing users by their keystroke patterns, using the method of processing training data, software was created in order to implement biometric authentication of users of information systems by their keystroke patterns, which allows to increase the degree of multifactor authentication of information systems without requiring additional equipment.

Based on the proposed method of recognizing users by their handwriting, using the method of processing training data, software was created in order to implement biometric authentication of information systems users by their handwriting, which allows to increase the degree of multifactor authentication of information systems, in the presence of standard sensory input graphic tools.

Based on the results of the experiments, using the developed software, the choice of configuration parameters, settings for which is the most critical of increase the probability of correct recognition of information system users, was accomplished and the estimation of probability of correct recognition of users by the selected biometric characteristics, which is provided by the use of probabilistic neural network, was obtained.

The results of the thesis research have been implemented in the following organizations: in the Verification Office of the General Staff of the Armed Forces of Ukraine (act dated 25.03.2010), in the management of the Pension Fund of Ukraine in the Kiev-Svyatoshinsky district (act dated 27.05.2010), in the company "Integrator" (act dated 24.12.2013), in Limited Liability Company «MFG «Infozahyst» (act dated 12.12.2018), in the Pukhov Institute for Modelling in Energy Engineering of NAS of Ukraine (act dated 01.11.2019), and also are used in educational process of the Department of Computerized information security systems, National Aviation University (act dated 29.05.2019).

Keywords: authentication, biometrics, keystroke pattern, handwriting, probabilistic neural network, recognition, information systems.

Список публікацій здобувача

1. O. Vysotska, A. Davydenko, «Keystroke Pattern Authentication of Computer Systems Users as One of the Steps of Multifactor Authentication», *Advances in Computer Science for Engineering and Education II. Advances in Intelligent Systems and Computing*, vol 938, pp. 356-368, 2019.

2. O. Vysotska, A. Davydenko, «Authentication of information systems users, based on the analysis of their handwriting», *Scientific and Practical Cyber Security Journal (SPCSJ)*, *vol.* 2, no. 4, pp. 51-63, 2018.

3. О. Корченко, А. Давиденко, О. Висоцька, «Метод автентифікації користувачів інформаційних систем за їх рукописним почерком з багатокроковою корекцією первинних даних», *Захист інформації*, Том 21, №1, С. 40-51, 2019.

4. Е.А. Высоцкая, А.Н. Давиденко, «Количественный и качественный анализ учебных данных с целью повышения эффективности аутентификации пользователей компьютерной системы при помощи нейронных сетей», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 24, С. 110-116, 2003.

5. Е.А. Высоцкая, «Компьютерное моделирование задач аутентификации пользователя компьютерных систем с помощью вероятностных нейронных сетей», *Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г. Є. Пухова*, Вип. 24, С. 3-9, 2004.

6. Е. Высоцкая, А. Давиденко, «Исследование эффективности применения вероятностных нейронных сетей для решения задачи аутентификации пользователя компьютерных систем», *Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні. Наук.-техн. зб.*, Вип. 9, С. 103-110, 2004.

7. Е.А. Высоцкая, А.Н. Давиденко, «Классификация биометрических систем аутентификации», *Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г. Є. Пухова*, Вип. 27, С. 108-114, 2004.

8. Е.А. Высоцкая, А.Н. Давиденко, «Определение критичных параметров при выборе биометрической системы аутентификации», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 27, С. 80-86, 2004.

9. Е.А. Высоцкая, «Оценка качества методов биометрической аутентификации и способы его повышения», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 28, С. 94-102, 2004.

10. Е.А. Высоцкая, «Исключение учебных данных с грубыми ошибками, как один из способов повышения эффективности применения вероятностных нейронных сетей для аутентификации пользователей компьютерных систем по клавиатурному почерку», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 29, С. 52-59, 2005.

11. Е.А. Высоцкая, «Выбор анализируемых параметров при аутентификации пользователей компьютерных систем по клавиатурному почерку при помощи вероятностной нейронной сети», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 30, С. 45-52, 2005.

12. Е.А. Высоцкая, «Влияние исключения учебных данных с грубыми ошибками на зависимость эффективности применения вероятностных нейронных сетей для аутентификации пользователей компьютерных систем по клавиатурному почерку от различных параметров», *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН Украіни*, Вип. 28, С. 3-10, 2005.

13. Е.А. Высоцкая, «Влияние параметров учебных данных на качество аутентификации при помощи вероятностной нейронной сети», *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН Украіни*,

Вип. 32, С. 10-17, 2006.

14. Е.А. Высоцкая, «Задача распознавания написанного ключевого слова, как одна из задач, решаемых при выполнении аутентификации пользователей компьютерных систем по рукописному почерку», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 36, С. 67-76, 2006.

15. Е.А. Высоцкая, А.Н. Давиденко, «Анализ технологии предварительной обработки данных при аутентификации пользователей компьютерных систем по клавиатурному и рукописному почеркам», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 55, С. 34-41, 2010.

16. Е.А. Высоцкая, «Выбор анализируемых характеристик при аутентификации пользователей компьютерных систем по рукописному почерку на разных этапах развития вычислительной техники», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 56, С. 31-39, 2010.

17. О.О. Висоцька, «Моніторинг роботи користувачів комп'ютерних систем за допомогою технологій розпізнавання за клавіатурним почерком», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 84, С. 119-125, 2018.

18. Е.А. Высоцкая, «Влияние параметров учебных данных на качество аутентификации пользователей компьютерных систем», *Моделювання: XXIV Науково-технічна конференція*, Київ, 2005, С. 3.

19. А.М.Давиденко, С.Я.Гільгурт, О.О.Висоцька, А.А.Кочурков, Ю.О.Чернова, «Експериментальне дослідження програми для автентифікації користувачів комп'ютерної системи за клавіатурним почерком за допомогою імовірнісної нейронної мереж», *Проблемы информатики и моделирования: пятая междунар. научно-технич. конф.*, Харьков, 2005, С. 24, 66-69.

20. Е.А. Высоцкая, «Метод проведения аутентификации пользователей компьютерных систем по рукописному почерку», *Моделювання: науково-технічна конф. молодих вчених і спеціалістів,* Київ, 2006, С. 9-10.

21. А.Н. Давиденко, Е.А. Высоцкая, «Использование биометрических систем защиты для уменьшения риска возникновения чрезвычайных ситуаций», *Декларування безпеки об'єктів підвищеної небезпеки як засіб регулювання безпеки*

регіону (держави): науково-методич. семінар у рамках IV Міжнародного виставкового форуму «Технології захисту – 2007», Київ, 2007, С. 123-126.

22. Е.А. Высоцкая, «Аутентификация пользователей компьютерных систем по клавиатурному почерку при помощи вероятностной нейронной сети», *Моделювання: XXIX Науково-технічна конф.*, Київ, 2010, С. 10.

23. О.О.Висоцька, «Використання біометричних технологій розпізнавання в системах моніторингу роботи користувачів комп'ютерних систем», *Проблеми створення, розвитку та застосування інформаційних систем спеціального призначення: 18-а науково-практична конф.*, Житомир, 2011, С.219-220.

24. О.О. Висоцька, «Підвищення рівня безпеки комп'ютерних систем за допомогою біометричної автентифікації», *Проблеми створення, розвитку та застосування високотехнологічних систем спеціального призначення: XX Всеукраїнська науково-практична конференція*, Житомир, 2014, С. 190-191.

25. O. Vysotska, A. Davydenko, «The usage of handwriting recognition systems of information systems users for their authentication», *La science et la technologie à l'ère de la société de l'information: conférence scientifique et pratique internationale*, Bordeaux, France, 2019, vol. 9, pp. 48-51.

26. О. Висоцька, А. Давиденко, В. Щербина, «Формалізація процедури аналізу рукописного почерку людини для організації розмежування доступу до інформаційних систем», *ITSec: Безпека інформаційних технологій: IX Міжнародна науково-технічна конф.*, Київ, 2019, С.22-23.

27. А.М. Давиденко, О.О. Висоцька, «Визначення функції моніторингу стану санкціонованих користувачів комп'ютерних систем за допомогою аналізу їх клавіатурного почерку», *Комп'ютерні системи та мережні технології" (CSNT-2019): XII Міжнародна науково-практична конф.*, Київ, 2019, С.41-42.

28. А.М. Давиденко, О.О. Висоцька, «Моніторинг функціонального стану представників критичних професій, за допомогою аналізу їх клавіатурного почерку», *Актуальні проблеми управління інформаційною безпекою держави: X Всеукраїнська науково-практична конф.*, Київ, 2019, С.201-203.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

FSRP – метод розпізнавання, який заснований на метод синтаксичного аналізу

А – необхідність додаткового апаратного забезпечення

АК – автентифікація користувачів

АККП – метод автентифікації користувачів інформаційних систем за їх клавіатурним почерком

АКРП – метод автентифікації користувачів інформаційних систем за їх рукописним почерком

АМОД – метод розпізнавання, який заснований на аналізі математичного очікування та дисперсії

АМОДДЗ – метод розпізнавання, який заснований на аналізі математичного очікування та дисперсії, з виконанням додаткового зважування

АРКНПЗК – метод розпізнавання, який заснований на аналізі ритму клавіатурного набору з пропорційним завданням класів

АРКНПРЗК – метод розпізнавання, який заснований на аналізі ритму клавіатурного набору з пороговим завданням класів

АСХ – метод розпізнавання, який заснований на аналізі статистичних характеристик

АСХФПФ – метод розпізнавання, який заснований на аналізі статистичних характеристик для фіксованої парольної фрази

АФЩРІВЗ – метод розпізнавання, який заснований на аналізі функції щільності розподілу імовірностей випадкових змінних

Б – використання в якості етапу багатофакторної автентифікації

БДНЗ – база даних навчальних зразків

БС – біометричні системи

В – велика

ВЯАП – алгоритм визначення якості аналізованих параметрів

ВБП – метод розпізнавання, який заснований на використанні багатошарового персептрона

ВБРНМ – метод розпізнавання, який заснований на використанні багатовимірних рекурентних нейронних мереж

ВКВБ – метод розпізнавання, який заснований на використанні класифікатора з векторним базисом

ВКНС – метод розпізнавання, який заснований на використанні К-найближчих сусідів

ВКР – метод розпізнавання, який заснований на використанні корекції рангів

ВНЗСВКЗ – алгоритм виключення навчальних зразків з грубими помилками за допомогою сортування та виключення крайніх значень

ВНЗПОСЗ – алгоритм виключення навчальних зразків з грубими помилками за допомогою порівняння ознаки з його середнім значенням

ВПММ – метод розпізнавання, який заснований на використанні прихованих Марковських моделей

ВПНК – метод розпізнавання, який заснований на використанні параметричного навчання класифікатора

ВРКС – метод розпізнавання, який заснований на використанні рангової кореляції Спірмена

ВШНМ – метод розпізнавання, який заснований на використанні штучних нейронних мереж

ГП – графічний планшет

Д – необхідна довжина зразка почерку

ДНВТ – динаміка набору вільного тексту

ДНПКФ – динаміка набору постійної ключової фрази

ДНПФСЗН – динамікою набору повторюваного фрагмента слова з набору

ДНСЗН – динаміка набору слова з набору

ІНМ – імовірнісні нейронні мережі

ІПР – імовірність правильного розпізнавання

ІС – інформаційні системи

К – стабільність

КП – клавіатурний почерк

КРДМ – код реалізацій дочірнього модуля

КТ – контрольна точка

КФ – ключова фраза

Л – невід'ємність

М – маленька

Н – імовірність помилкового надання доступу порушникові

НЗ – навчальний зразок

ННЗЗЗЗП – алгоритм накопичення навчальних зразків з запам'ятовуванням зроблених помилок

О – імовірність помилкової відмови доступу легальному користувачеві

П – непідробленість

ПАКПІНМ – алгоритм процесу автентифікації користувачів інформаційних систем за клавіатурним почерком за допомогою імовірнісної нейронної мережі

ПОЗКП – метод первинної обробки зразків клавіатурного почерку

ПОЗРП – метод первинної обробки зразків рукописного почерку

ПКВ – помилка клавіатурного вводу

РП – рукописний почерк

С – середня

САКПІНМ – алгоритм роботи системи автентифікації за клавіатурним почерком за допомогою імовірнісної нейронної мережі

Т – біометрична характеристика, яка використовується для автентифікації

ТААУ – метод розпізнавання, який заснований на технології аналізу Амфреса Д. і Вільямса Г.

ТАГК – метод розпізнавання, який заснований на технології аналізу Генуе Р. і Кечаді Т.

ТАІ – метод розпізнавання, який заснований на технології аналізу Іванова

У – унікальність

Ф – використання для аналізу функціонального стану користувача

Х – використання для аналізу характеристик, необхідних працівникам критичних професій (уважності, сконцентрованості, акуратності)

ВСТУП

Актуальність теми. Постійне зростання частоти використань на підприємствах майже всіх сфер діяльності, інформаційних систем (ІС), які реалізують різноманітні інформаційні технології та, насамперед, обробляють і зберігають конфіденційну інформацію, привело до загострення необхідності створення ефективних систем розмежування доступу до ІС. Одним з основних способів реалізації подібних систем є автентифікація користувачів ІС, які необхідно захищати. Існує декілька способів розв'язання задачі антентифікації, але у кожного з них є свої недоліки (неприпустимо велика імовірність неправильного розпізнавання користувача, необхідність носити з собою картки доступу або пам'ятати пароль, висока вартість та інші). Наявність вказаних недоліків робить актуальним пошук нових способів розв'язання задачі автентифікації користувачів (АК).

Завдяки використанню, для автентифікації, біометричних характеристик людини, відпадає велика кількість вказаних проблем існуючих методів розпізнавання користувачів. Останнім часом часто використовують статичні біометричні методи автентифікації, тобто ті, що використовують для розпізнавання притаманні людині фізичні параметри (відбиток пальця, параметри ока, форма обличчя та інші). Але ті з статичних методів, що забезпечують високу достовірність розпізнавання, є занадто дорогими. Цього недоліку позбавлені динамічні біометричні методи автентифікації, тобто ті, що використовують для розпізнавання поведінкові характеристики людини (голос, клавіатурний почерк, рукописний почерк та інші). Підпис людини вже дуже давно використовується для розпізнавання людини. Одним з дослідників в напрямку розпізнавання за рукописним почерком (РП) був Ланцман Р.М., а за клавіатурним почерком (КП) – Леггетт Дж. Але розвиток сучасних інформаційних технологій зробив можливим використовувати не тільки статичний підпис, а й динаміку його відтворення, що значно підвищує достовірність автентифікації. На сьогоднішній день найбільш відомі дослідницькі праці з напрямку динамічних біометричних методів автентифікації за КП та РП наступних науковців: Іванова О.І., Чалої Л.Е., Расторгуєва С.П., Брюхомицького Ю.А., Леггетт Дж., Вільямс Г., Колтел О., Кітлер Й., Вонг М.Х.,

Амфрес Д., Блеха С.А., Бергадано Ф., Натан К.С., Грейвс А., Фіцджеральд Дж., Генуе Р., Кечаді Т. та інш.

Розвиток методів біометричної автентифікації є актуальним не тільки для захисту ІС, а й для контролю та обліку доступу на різноманітні об'єкти, наприклад, заводи, банки, поліцейські відділи та інші об'єкти з обмеженим доступом.

Крім того, методи розпізнавання за допомогою динамічних біометричних параметрів можна використовувати не тільки для АК ІС з метою захисту від порушників, а й для виявлення факту аномального стану у людини (стресу, хвороби, тощо) та під час прийому на роботу, для аналізу таких характеристик людини, як уважність, вміння сконцентруватися, акуратність. Розв'язання останніх двох задач корисне на підприємствах для працівників найбільш критичних професій, наприклад для диспетчерів в аеропортах та для банківських працівників. Цей факт ще раз аргументує актуальність дослідження та розвитку динамічних біометричним методів розпізнавання об'єктів.

Існує декілька механізмів вирішення задачі біометричної автентифікації, одним з котрих є нейронні мережі. Враховуючи той факт, що автентифікація по суті представляє собою більш поширену задачу – задачу класифікації об'єктів, то можна сказати що найбільш придатним для вирішення цієї задачі є такий різновид нейронних мереж, як імовірнісні нейронні мережі (ІНМ). Але цей механізм вирішення даної задачі є ще не досить розвинутим, тому дослідження ефективності використання нейронних мереж та насамперед ІНМ для АК та створення систем на базі цього методу є актуальним питанням розвитку сучасних методів захисту інформації.

Таким чином, можна сказати, що створення біометричних методів АК ІС за їх КП та РП, використовуючи для розпізнавання ІНМ, є актуальною задачею розвитку сучасних систем захисту інформації, що зберігається, обробляється і передається в ІС.

Зв'язок роботи з науковими програмами, планами, темами. Дослідження, що проводились, при виконанні дисертаційної роботи виконувались у відповідності з планом науково-дослідних робіт Інституту проблем моделювання в енергетиці (ІПМЕ) ім. Г.Є. Пухова НАН України в рамках наступних науково-дослідних тем: НДР «Кріт» «Розробка методів побудови та формального опису критеріїв оцінки захищеності

інформації в комп'ютерних системах від несанкціонованого доступу» №0101U006700 (2001р.–2004р.). НДР «МодА» «Дослідження і розробка методів розпізнавання, які базуються на використанні спектральних перетворень, для інформаційного забезпечення безпеки енергетичних об'єктів» №0105U001296 (2005р.–2008р.). НДР «МодБ» «Дослідження та розробка методів підвищення безпеки та ефективності розподілених високопродуктивних інформаційних технологій при вирішенні задач енергетики» №0108U010588 (2009р.–2013р.). НДР «МОД-Д» «Дослідження та розробка методів оцінювання захищеності інформації в розподілених високопродуктивних інформаційних системах при вирішенні задач енергетики» №0114U002361 (2014р.–2018р.). Також частка досліджень виконувалась в рамках: Науково-технічної програми «Розвиток системи технічного захисту інформації в Україні», Постанова Кабінету Міністрів України від 21.06.2000р. №681-009 та програми робіт з організації, стандартизації та сертифікації в галузі ТЗІ це роботи, які проводились ІПМЕ ім. Г.Є. Пухова НАН України разом з КПІ в інтересах Державної служби спеціального зв'язку та захисту інформації України НДР «РизикМ» договір №239-01 від 15.09.2001р. (2001р.–2007р.). Результати дисертаційної роботи також застосовувалися при проведенні практичних робіт з експертизи технічних систем захисту. Прикладом такої роботи є експертиза «Створення та проведення первинної державної експертизи комплексної системи захисту інформації на об'єкті, що належить Департаменту військово-технічної політики, розвитку озброєння, та військової техніки Міністерства оборони України», яка виконувалась ІПМЕ ім. Г.Є. Пухова НАН України (27.06.2018р.–31.12.2018р.), відповідно до договору №149 від 27.06.2018р. Одержані результати дисертаційної роботи також застосовувались при виконанні НДР №116/09.01.09, Національного авіаційного університету «Криптографічні методи захисту в сучасних інформаційно-комунікаційних системах та мережах» (01.09.2018р.–30.06.2019р.).

Мета і завдання дослідження. Метою дисертаційної роботи є підвищення імовірності правильного розпізнавання користувачів інформаційних систем за рахунок розробки нових методів автентифікації користувачів, які використовують для розпізнавання біометричні характеристики користувача, застосовуючи нейронні

мережі для ідентифікації його біометричного образу.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

1. Проаналізувати існуючи біометричні методи розпізнавання та програмні і апаратні засоби на їх основі, з метою здійснення вибору методів автентифікації користувачів інформаційних систем, застосування яких забезпечить задану імовірність правильного розпізнавання користувачів та не потребуватиме суттєвих витрат на впровадження.

2. Розробити методи первинної обробки навчальних даних, використання яких дозволить збільшити імовірність правильного розпізнавання користувачів інформаційних систем та зменшить затрачувані ресурси.

3. Розробити методи автентифікації користувачів інформаційних систем, які базуються на обраних біометричних методах розпізнавання і використовують для цього обраний різновид нейронної мережі.

4. Створити програмне забезпечення, яке на основі розроблених методів розпізнавання користувачів, виконуватиме автентифікацію користувачів інформаційних систем за обраними біометричними характеристиками, та по-перше, оцінюватиме імовірність їх правильного розпізнавання, по-друге, на основі аналізу накопичених даних даватиме змогу здійснити вибір конфігураційних параметрів систем розпізнавання.

Об’єкт дослідження: процеси біометричної автентифікації користувачів інформаційних систем на основі їх клавіатурного та рукописного почерку.

Предмет дослідження: методи автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком.

Методи дослідження: комп’ютерне моделювання (для дослідження імовірнісної нейронної мережі), статистичний аналіз (для визначення імовірності правильного розпізнавання, яку забезпечують методи автентифікації), теорія імовірності (для кількісного аналізу параметрів систем захисту), лінійна алгебра (для визначення вагових коефіцієнтів, в методах навчання імовірнісної нейронної мережі), методи емпіричних та теоретичних досліджень (для побудови методів автентфікації), комбінаторний аналіз (для визначення кількісних параметрів надійності

розпізнавання), об’єктно-орієнтовані інформаційні технології (для програмної реалізації запропонованих методів), математичний апарат нейронних мереж (для автентифікації користувачів).

Наукова новизна одержаних результатів полягає в наступному:

1. Вдосконалено класифікацію біометричних систем розпізнавання, що дало змогу здійснити вибір біометричних методів автентифікації користувачів інформаційних систем, застосування яких забезпечує задану імовірність правильного розпізнавання користувачів та не потребує суттєвих витрат на впровадження.

2. Вперше запропоновано метод первинної обробки зразків клавіатурного почерку, який за рахунок аналізу спектральних характеристик почерку користувача, дозволяє виключити хибні зразки почерку, які є нехарактерними і викликані випадковими помилками користувача при наборі тексту, що забезпечує більш високу і рівномірну якість характеристик почерку і, завдяки цьому, збільшується імовірність правильного розпізнавання користувачів.

3. Вдосконалено метод автентифікації користувачів інформаційних систем за їх клавіатурним почерком, який за рахунок використання для розпізнавання імовірнісної нейронної мережі та виконання первинної обробки зразків клавіатурного почерку, збільшує імовірність правильного розпізнавання користувачів інформаційних систем.

4. Вперше запропоновано метод первинної обробки зразків рукописного почерку, в якому за рахунок автоматизації процесу відбору контрольних точок в зразках рукописного почерку, чиї характеристики аналізуються, видаленню помилкових точок п’яти типів та проведенню корекції даних трьох типів, досягається збільшення імовірності правильного розпізнавання користувачів інформаційних систем та, завдяки зменшенню кількості ознак в зразках, що аналізуються, зменшення затрачуваних ресурсів.

5. Вдосконалено метод автентифікації користувачів інформаційних систем за їх рукописним почерком, який за рахунок використання для розпізнавання імовірнісної нейронної мережі та виконання первинної обробки зразків рукописного почерку, збільшує імовірність правильного розпізнавання користувачів інформаційних систем.

Практичне значення одержаних результатів. Створене в роботі програмне

забезпечення, на основі розроблених методів розпізнавання людей за клавіатурним та рукописним почерком, на базі імовірнісної нейронної мережі, може використовуватися для автентифікації користувачів в інформаційних систем; для розпізнавання працівників в системах контролю доступу та обліку користувачів; для визначення аномального стану працівника на підприємствах, на яких для їх нормального функціонування критична наявність у працівників уважності під час роботи.

Практична цінність роботи полягає в наступному:

– на основі запропонованого методу розпізнавання користувачів за їх клавіатурним почерком, з використанням методу обробки навчальних даних, створено програмне забезпечення для реалізації біометричної автентифікації користувачів інформаційних систем за їх клавіатурним почерком, яке дозволяє збільшити ступінь багатофакторності автентифікації інформаційних систем, не вимагаючи для цього додаткового обладнання.

– на основі запропонованого методу розпізнавання користувачів за їх рукописним почерком, з використанням методу обробки навчальних даних, створено програмне забезпечення для реалізації біометричної автентифікації користувачів інформаційних систем за їх рукописним почерком, що дозволяє збільшити ступінь багатофакторності автентифікації інформаційних систем, при наявності стандартних сенсорних засобів вводу графічної інформації.

– на основі результатів проведених експериментів, за допомогою розробленого програмного забезпечення, здійснено вибір конфігураційних параметрів, налаштування яких є найбільш критичним для збільшення імовірності правильного розпізнавання користувачів інформаційних систем та отримана оцінка імовірності правильного розпізнавання користувачів інформаційних систем за обраними біометричними характеристиками, яку забезпечує використання імовірнісної нейронної мережі.

– результати дисертаційних досліджень впроваджені в наступних організаціях: в Управлінні верифікації Генерального штабу Збройних сил України (акт від 25.03.2010р.), в управлінні Пенсійного фонду України у Києво-Святошинському

районі (акт від 27.05.2010р.), в підприємстві «Інтегратор» (акт від 24.12.2013р.), в ТОВ «НВЦ «ІНФОЗАХИСТ» (акт від 12.12.2018р.), в ІПМЕ ім. Г.Є. Пухова НАН України (акт від 11.01.2019р.), а також використовуються у навчальному процесі кафедри комп'ютеризованих систем захисту інформації Національного авіаційного університету (акт від 29.05.2019р.).

Особистий внесок здобувача. Всі результати, які становлять основний зміст дисертації, автор отримав особисто. У роботах написаних у співавторстві, автору дисертації належить: [1] – розробка методу розпізнавання користувачів інформаційних систем за клавіатурним почерком та дослідження ефективності використання його при багатофакторній автентифікації; [2,3,25] – розробка методу автентифікації користувачів інформаційних систем за рукописним почерком; [4,6,19] – збір інформації, програмна реалізація системи біометричної автентифікації, проведення експериментів та дослідження, на основі аналізу їх результатів, процесу використання імовірнісної нейронної мережі для біометричної автентифікації з метою визначення критичних параметрів систем автентифікації; [7,8] – класифікація біометричних систем автентифікації; [15] – розробка технології попередньої обробки даних при автентифікації за клавіатурним та рукописним почерком та оцінка ефективності її застосування; [21] – дослідження особливостей використання біометричних методів в системах контролю доступу, побудова алгоритму роботи системи біометричного контролю доступу; [26] – розробка та програмна реалізація методу антентифікації користувачів за рукописним почерком для розмежування доступу до інформаційних систем; [27,28] – дослідження особливостей використання клавіатурного почерку людини для реалізації функції моніторингу.

Апробація результатів дисертації. Результати дисертації доповідались та обговорювались на конференціях та на науково-методичному семінарі, серед яких: XXIII та XXIV ЩНТК «Моделювання» (Київ, 2004р., 2005р.), V МНПК «Безпека інформації в інформаційно-телекомунікаційних системах» (Київ, 2002р.), пятая МНТК «Проблемы информатики и моделирования» (Харьков, 2005р.), VII ВНПК рятувальників «Пожежна безпека та аварійно-рятувальна справа: стан, проблеми і перспективи» (Київ, 2005р.), НТК молодих вчених і спеціалістів «Моделювання»

(Київ, 2006р., 2010р.), НМС «Декларування безпеки об’єктів підвищеної небезпеки як засіб регулювання безпеки регіону (держави)» у рамках IV МВФ «Технології захисту – 2007» (Київ, 2007р.), 18-а НПК «Проблеми створення, розвитку та застосування інформаційних систем спеціального призначення» (Житомир, 2011р.), XX ВНПК «Проблеми створення, розвитку та застосування високотехнологічних систем спеціального призначення» (Житомир, 2014р.), The Second International Conference on Computer Science, Engineering and Education Applications (ICCSEEA2019) (Kiev, 2019y.), Conférence scientifique et pratique internationale «La science et la technologie à l'ère de la société de l'information» (Bordeaux, France, 2019a.), IX МНТК «ITSec: Безпека інформаційних технологій» (Київ, 2019р.), XII МНПК «Комп'ютерні системи та мережні технології» (CSNT-2019) (Київ, 2019р.), X ВНПК «Актуальні проблеми управління інформаційною безпекою держави» (Київ, 2019), VII МНТК «Захист інформації і безпека інформаційних систем». (Львів, 2019р.).

Публікації. Матеріали дисертації опубліковано в 28 наукових працях, в тому числі в 1 науковій статі у міжнародному рецензованому виданні, що входить до бази даних Scopus [1], в 1 науковій статі у закордонному фаховому науковому журналі [2], в 1 науковій статі у міжнародному рецензованому виданні, що входить до бази даних Index Copernicus [3], в 14 статтях у наукових фахових журналах та збірниках (зокрема в 9 – без співавторів [5,9–14,16–17]) [4–17], а також в 11 тезах доповідей конференцій, в матеріалах конференцій, в тезах доповідей науково-методичного семінару (зокрема в 5 – без співавторів [18,20, 22–24]) [18–29].

Структура та обсяг дисертації. Дисертація складається з анотації, переліку умовних скорочень, вступу, чотирьох розділів, висновків, списку використаних джерел та трьох_ додатків. Робота містить 35 рисунків, 5 таблиць. Список використаних джерел складається з 200 найменувань і займає 20 сторінок. Додатки розміщені на 97 сторінках. Загальний обсяг дисертації складає 272 сторінки, основний текст роботи викладено на 140 сторінках.

РОЗДІЛ 1. АВТЕНТИФІКАЦІЯ КОРИСТУВАЧІВ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1.Автентифікація користувачів інформаційних систем і типи автентифікації

Інформаційна система (ІС) – це організаційно технічна система, в якій реалізується технологія обробки інформації з використанням технічних і програмних засобів [1]. Також можна сказати, що за ДСТУ 2392-94: Інформаційна система – це комунікаційна система, що забезпечує збирання, пошук, оброблення та пересилання інформації. Всі інформаційні системи можна класифікувати за наступними характеристиками: за типом даних, які зберігаються; за ступенем автоматизації; за сферою застосування; за характером обробки даних; за рівнем керування.

Однією з найбільш серйозних загроз для інформації, що зберігається і оброблюється за допомогою ІС, є несанкціонований доступ до неї [2-15]. І відповідно, одним із головних завдань систем захисту інформації - є завдання ідентифікації і автентифікації для забезпечення конфіденційності інформації та розмежовування до неї доступу (рис.1). Тому розглянемо цю задачу докладніше і проаналізуємо способи її вирішення [16-26]. Після чого, виберемо найбільш оптимальний з них.

Ідентифікація – процедура присвоєння ідентифікатора об'єкту ІС або встановлення відповідності між об'єктом і його ідентифікатором; впізнання. Автентифікація – процедура перевірки відповідності пред'явленого ідентифікатора об'єкта ІС на предмет приналежності його цьому

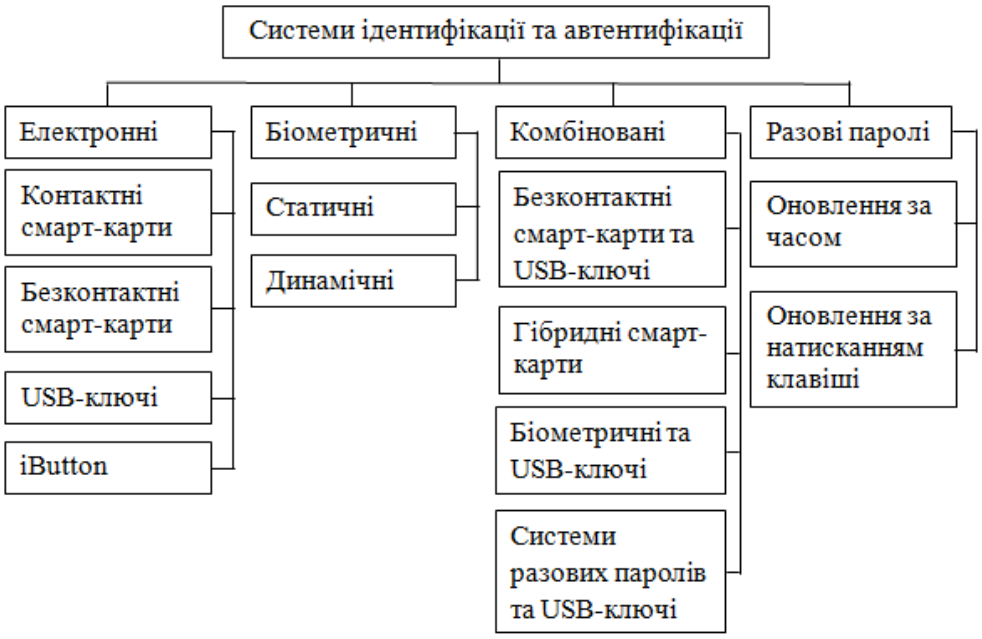


Рис. 1. Класифікація систем ідентифікації та автентифікації

об'єкту; встановлення або підтвердження автентичності. Більш детально розглянемо процес і типи автентифікації. Автентифікація, як правило, виконується у двух випадках: при вході в ІС або для отримання доступу до будь-яких інших ресурсів; періодично під час роботи в системі, для виконання моніторингу.

В різних випадках переважніше виявляються різні типи автентифікації. Існує *три типи автентифікації*, які використовують для розпізнання, відповідно: щось, відоме користувачеві; щось, чим володіє користувач; щось притаманне користувачеві. Для подальшого дослідження та використання обрана автентифікація, що використовує щось притаманне користувачеві.

1.2.Біометричні методи автентифікації користувачів інформаційних систем

1.2.1. Біометрія, як один із способів вирішення задачі автентифікації користувачів інформаційних систем

Біометрична автентифікація (біометричне управління доступом – BAC – Biometric Access Control) – це використання однієї або декількох особливостей будови тіла людини, його рухових навичок, характеру, психологічних і розумових характеристик для підтвердження з необхідною точністю і впевненістю того, що даний користувач дійсно той за кого себе видає, а не порушник [27-69]. Термін «біометрія» складається з двох грецьких слів, які означають життя і вимірювання. Біометрію можна використовувати при вирішенні трьох різних, але пов'язаних завдань:

1. Автентифікації.
2. Ідентифікації.
3. Визначення унікальності.

При вирішенні задачі *автентифікації* перевіряється, чи дійсно пред'явлене ім'я користувача відноситься до тієї людини, який його представив. В якості базового секрету або засобу верифікації в даному випадку використовуються біометричні показники. Для позитивного результату автентифікації, пред’явлені біометричні показники повинні близько збігатися із записом, який зберігається в базі даних для даного користувача. При вирішенні задачі *ідентифікації* визначається чи можна пред'явлений зразок біометричних показників пов'язати з конкретною людиною, дані якої присутні в базі даних, або, хоча б, з меншою кількістю людей з наявної бази да-

них. Це завдання може бути вирішене тільки в системах з великою базою зразків біометричних показників, в якій, імовірно знаходиться запропонований зразок. Завдання визначення *унікальності* є різновидом завдання ідентифікації. Дане завдання полягає у визначенні того, чи присутній власник пред'явленого зразка біометричних показників в наявній базі даних. Вирішення цього завдання необхідно для запобігання повторної реєстрації в будь-яких організаціях. Системи, які вирішують завдання ідентифікації і системи, які використовуються для визначення унікальності, використовують однакову методику, але у них різне призначення. Системи ідентифікації, на відміну від систем визначення унікальності, повинні прагнути до максимальної повноти бази даних навчальних зразків [16-69].

Розв'язання трьох розглянутих видів завдань, в яких використовуються біометричні технології, необхідно в різних областях. Однією з них є інформаційна безпека, а особливо під час розв'язанні задачі автентифікації користувачів ІС. В інформаційній безпеці біометрія використовується для заміни (або для посилення) стандартної процедури входу в різні програми за паролем, смарт-картою, таблеткою touch-memory тощо.

Головними перевагами біометричної автентифікації від інших методів вирішення тієї ж завдання є:

1. Людина є носієм свого «біометричного пароля». Тобто, немає необхідності пам'ятати пароль і його не можна забути.

2. Високий ступінь унікальності таких паролів.

3. Важче фальсифікувати такий пароль.

4. Задовольняють вимогам безпеки певних ІС, коли необхідно використовувати такі ключі доступу, які не можуть бути використані окремо від їх носія.

Тому одним з поширених застосувань біометричних систем є ідентифікація або автентифікація за біометричними характеристиками при вході користувача в мережу; при вході в операційну систему персонального комп'ютера; при вході в будь-які програми; для посилення доступу до криптографічних ключів; в якості криптографічного ключа; для моніторингу роботи користувачів в системі, що підсилює її захист.

Незалежно від сфери застосування, методу біометричної автентифікації і

біометричних характеристик людини, що аналізуються, використання біометричних систем автентифікації складається з двох етапів [27-69]:

1. Первісна реєстрація, при якій створюється база даних навчальних зразків відповідних біометричних характеристик необхідної кількості (в залежності від технології) для кожного користувача. При цьому за допомогою спеціальних біометричних (іноді стандартних) пристроїв зчитуються відповідні біометричні показники людини, яка реєструється, а потім з них формується цифровий біометричний зразок даного користувача, який після цього заноситься в базу даних.

2. Автентифікація (ідентифікація), при якій спочатку вводиться ім'я користувача і потім, якщо в базі даних є навчальні зразки для цього користувача, створюється зразок біометричних характеристик даної людини і порівнюється зі зразками перевіряємого користувача, що зберігаються в базі даних. Результатом порівняння зазвичай є число – імовірність того, що порівнювані зразки належать одній людині. Потім з використанням будь-якого математичного критерію приймається рішення про ідентичність зразків, тобто вирішується, чи дійсно цей користувач той за кого себе видає. Якщо виконується не автентифікація, а ідентифікація, тоді ім'я користувача не вводиться і, відповідно, не перевіряється наявність в базі даних інформації про нього, а відразу створюється зразок біометричних характеристик даної людини і порівнюється з усіма зразками всіх користувачів з бази даних. Тоді в результаті порівняння формується список з декількох найбільш схожих зразків (з найбільшими імовірностями, отриманими при порівнянні). Потім з використанням будь-якого математичного критерію приймається рішення про ідентичність зразків і таким чином визначається, хто з тих, чиї дані є в базі, найбільш імовірно є власником пред'явленого біометричного зразка.

Етап початкової реєстрації, в процесі роботи системи, необхідно повторити якщо:

1. Через якийсь час біометричні характеристики людини, що аналізуються, значно змінилися.

2. При початковій реєстрації, з якихось причин були зібрані неточні (помилкові) зразки, і внаслідок цього відбуваються часті відмови легальному користувачеві.

Біометричні системи (БС) автентифікації зазвичай оцінюють за наступними

параметрами (одним або декількома) [27-69]:

1. Імовірність помилки першого роду (P_{o1}) – імовірність відмови доступу до системи легальному користувачеві (не пропустити в систему «свого»).

2. Імовірність помилки другого роду (P_{o2}) – імовірність доступу до системи порушникові (пропустити в систему «чужого»).

3. Імовірність того, що одночасно відбудуться помилки і першого і другого роду (P_{o12}) – імовірність того, що система прийме одного легального користувача за іншого, теж легального користувача (але з іншими правами доступу).

4. Рівень помилок біометричної системи, при якому помилка першого роду дорівнює помилці другого роду (P_o) – точка рівності імовірностей (комплексний показник якості).

5. Імовірність відсутності помилки першого роду $P_1 = 1 - P_{o1}$.

6. Імовірність відсутності помилки другого роду $P_2 = 1 - P_{o2}$.

7. Імовірність правильного розпізнавання (ІПР) $P = P_1 \cdot P_2 = (1 - P_{o1}) \cdot (1 - P_{o2}) = 1 - -P_{o1} - P_{o2} + P_{o1} \cdot P_{o2}$.

В залежності від галузі використання даної системи захисту, значимість можливості пропуску «чужого» і значимість можливості відмови у пропуску «своєму» можуть бути різними, тому наведена формула ІПР матиме вигляд:

$$P = 1 - k_1 \cdot P_{o1} - k_2 \cdot P_{o2} + k_3 \cdot P_{o1} \cdot P_{o2},$$

де k_1 – коефіцієнт значущості забезпечення доступу «своєму», k_2 – коефіцієнт значущості забезпечення відмови «чужому», k_3 – коефіцієнт значущості забезпечення доступу «своєму» під своїм ім'ям (з його правами доступу).

1.2.2. Класифікація біометричних систем автентифікації

Наявність на сучасному ринку засобів захисту великої кількості біометричних систем ставить перед користувачами і розробниками систем захисту інформації проблему оцінки ефективності їх застосування. Це завдання виявляється нетривіальним, зважаючи на якісні відмінності різних біометричних систем і відсутності однорідної інформації відносно їхніх характеристик.

Для розв’язання даної проблеми класифікуємо існуючи БС [69-123], для чого спочатку визначимо параметри на підставі, яких можна побудувати подібну оцінку. Для цього проведемо аналіз систем даного типу автентифікації, на основі відкритої інформації, доступної через Інтернет. Крім того, проаналізуємо, механізми підвищення ефективності роботи біометричних систем захисту. Потім підсумуємо, за яким принципом обирати і налаштовувати БС захисту інформації.

На основі зібраних даних проведемо класифікацію біометричних систем автентифікації за обраними ознаками (рис.2). Розглянемо класифікацію за кожною ознакою детальніше [37-39].

БС автентифікації можна класифікувати за біометричним методом, який вони використовують. Біометричні методи автентифікації діляться на два класи (рис.3): *статичні* методи; *динамічні* методи. *Статичні* методи основані на вимірюванні фізичних (статичних) характеристик людини, які повинні бути унікальними для кожної людини, або хоча б для більшості людей. Ці характеристики не повинні істотно змінюватися протягом тривалого проміжку часу і на них не повинні впливати будь-які зовнішні чинники, наприклад косметичні засоби, будь-які погодні явища і т.д. *Динамічні* методи біометричної автентифікації ґрунтуються на поведінкових (динамічних) характеристиках людини, тобто, побудовані на особливостях, характерних для підсвідомих рухів в процесі відтворення якої-небудь дії. На відміну від фізичних відмінних характеристик, в даному випадку, біометрична система не обов'язково повинна вимірювати кожен раз одне і те саме явище: людині може бути запропоновано сказати, написати або пройтися певним чином, щоб зменшити ризик відтворення характеристики порушником [37-39].

БС захисту можна класифікувати за імовірністю правильного розпізнавання (надійностю). Цей параметр, як було сказано раніше, залежить від імовірності помилки першого роду, імовірності помилки другого роду і від значущості кожної з цих помилок для кожної конкретної ІС. Не обов'язково системи з найкращим показником імовірності помилки першого роду будуть мати оптимальне значення

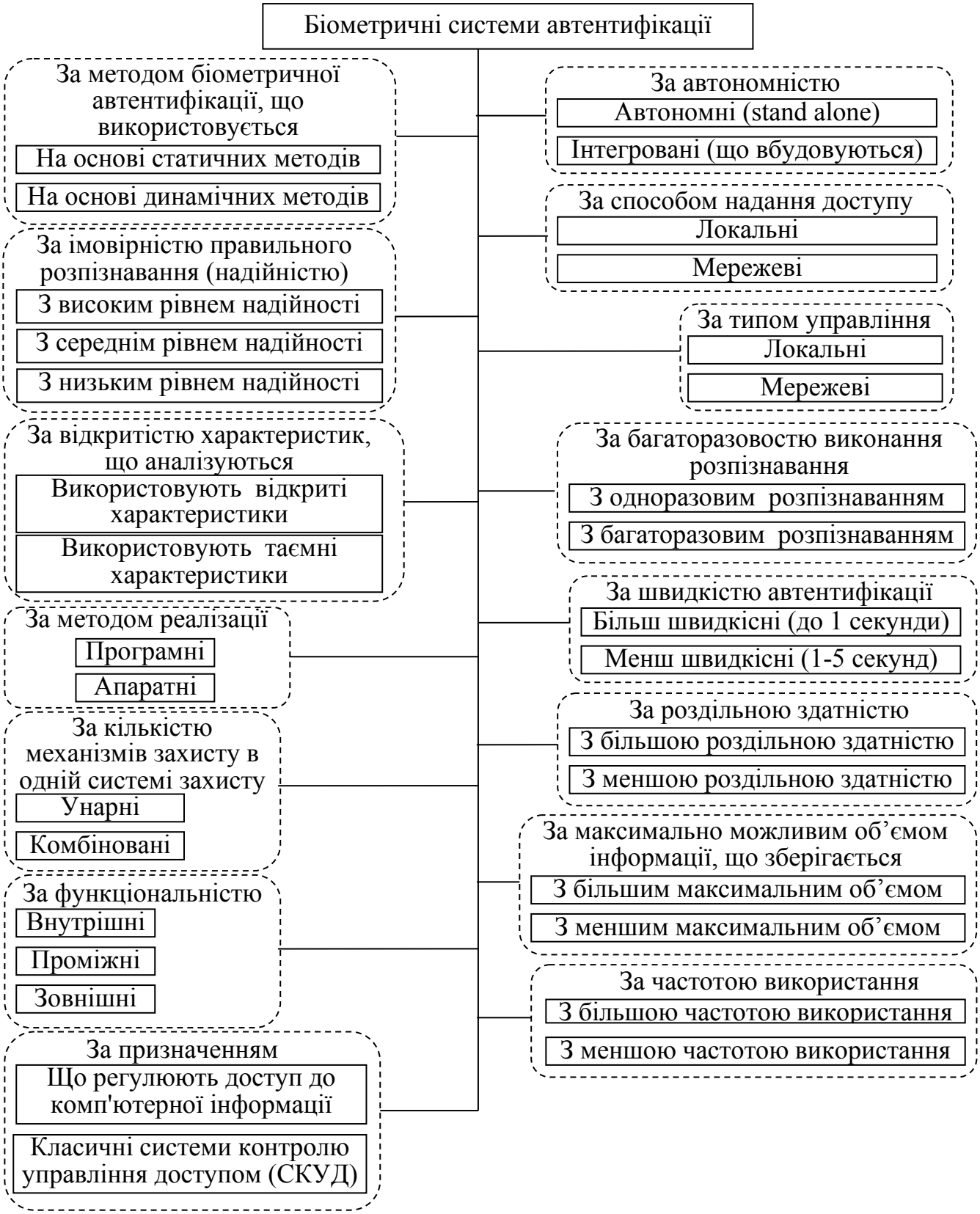


Рис. 2. Класифікація біометричних систем автентифікації

імовірності помилки другого роду. В залежності від ІПР БС пожно поділити на системи *з високим рівнем надійності*; *з середнім рівнем надійності* та *з низьким*

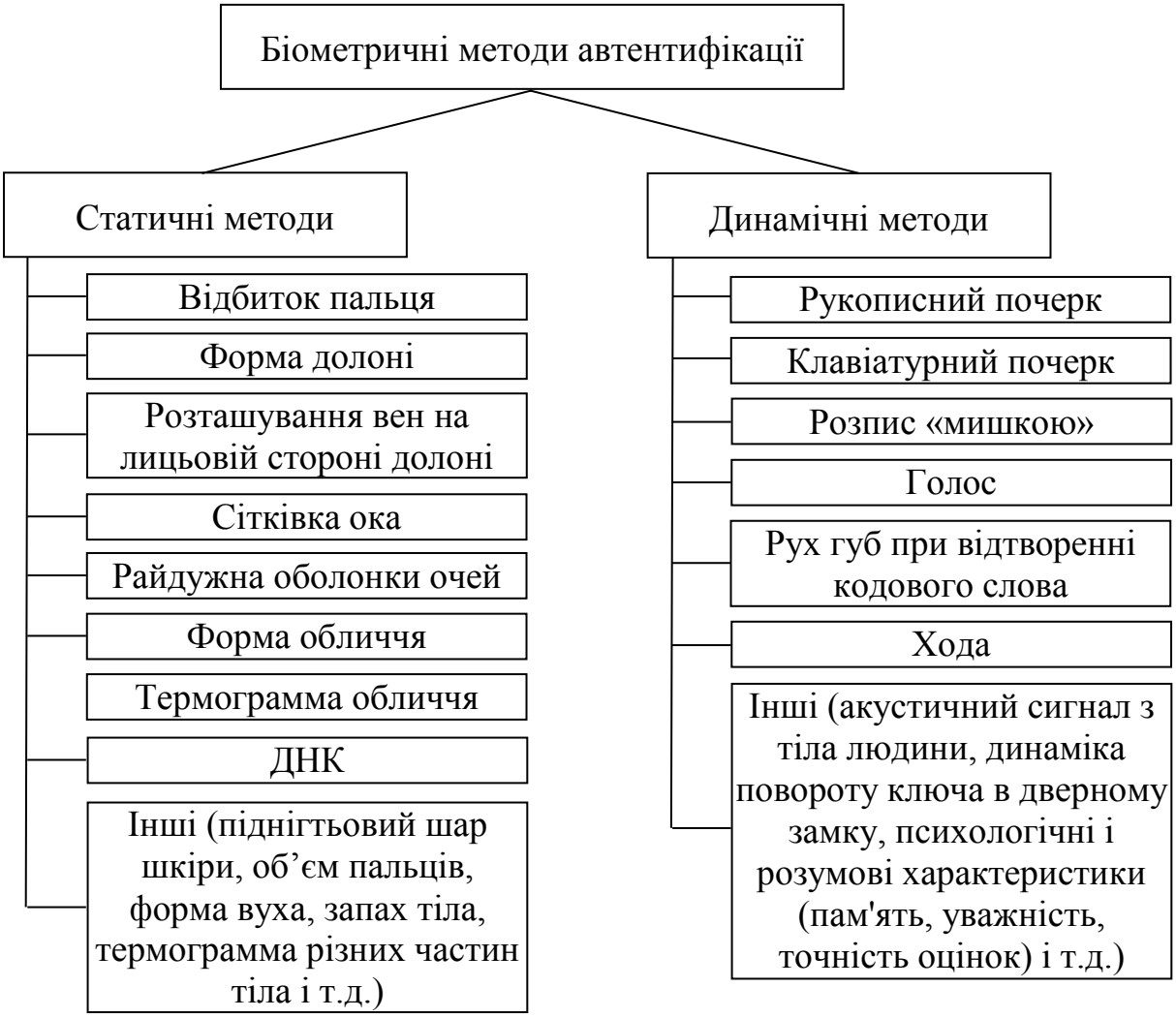


Рис.3. Класифікація біометричних методів автентифікації

рівнем надійності. Якщо вважати, що значимість наслідків від помилки першого роду дорівнює значимості наслідків від помилки другого роду, тоді за імовірністю правильного розпізнавання, за конкретним методом, БС можна розташувати в наступному порядку (від більш якісних до менш якісних): розпізнавання за ДНК; розпізнавання за райдужною оболонкою ока або за сітківкою ока; розпізнавання за відбитком пальця, або за термограмою обличчя, або за формою долоні; розпізнавання за формою обличчя або за розташуванням вен на кісті руки і долоні; розпізнавання за підписом; розпізнавання за клавіатурним почерком; розпізнавання за голосом. Тобто, можна сказати, що статичні методи якісніші, ніж динамічні, але вони значно дорожчі ніж динамічні методи. Але ця перевага статичних методів дійсна тільки в тому випадку, якщо в динамічних методах використовуються

відкриті біометричні характеристики [37-39].

БС за відкритістю характеристик, що використовуються, можна розділити на: системи, які використовують *відкриті характеристики*; системи, які використовують *таємні характеристики*. БС, які для розпізнавання використовують *відкриті характеристики* – це ті системи, які використовують будь-які характеристики людини, доступні для спостереження сторонніми. БС, які для розпізнавання використовують *таємні характеристики* – це ті системи, які використовують будь-які характеристики людини, недоступні для спостереження сторонніми. Всі системи, що використовують для розпізнавання статичні методи, є системами з відкритими характеристиками. Динамічні ж методи можуть використовувати як відкриті ознаки, так і таємні. Наприклад, якщо при розпізнаванні за клавіатурним почерком, вказується яке саме слово необхідно ввести, тоді це система з відкритими біометричними характеристиками, а якщо користувач повинен ввести відоме тільки йому слово-пароль, тоді це вже система з таємними біометричними характеристиками. Системи, що використовують таємні біометричні показники, є більш ефективними системами захисту, тому що, теоретично, будь-яку систему захисту можна подолати перебором всіх можливих значень всіх аналізованих параметрів, питання тільки у витраченому часі і кількості витрачених апаратних ресурсів. Кількість можливих комбінацій при переборі обчислюється за формулою: $Komb = (\prod_{i=1}^{k_{pr}} kol_{znpr_i})^{kol_s}$, де $Komb$ – кількість можливих комбінацій значень всіх аналізованих параметрів, kol_s – кількість введених яким-небудь чином (в залежності від методу) символів, k_{pr} – кількість ознак, що аналізуються, під час вводу кожного символу, kol_{znpr_i} – кількість можливих значень *i*-ої ознаки, що аналізується, при введенні кожного символу. Мова йде про символи тому, тому що в динамічних методах (більш детально розглядаються в даній роботі), як правило, одним із способів вводяться символи (вимовляються, пишуться, набираються на клавіатурі), а потім аналізуються будь-які параметри, заміряні при введенні кожного символу. При використанні таємних біометричних характеристик, тобто якщо сторонні не знають, які саме знаки розміщені, кількість

можливих комбінацій при переборі значно зростає. Це значення обчислюється за формулою: $Komb = (kol_{zns} \prod_{i=1}^{k_{pr}} kol_{znpr_i})^{kol_s}$, де kol_{zns} - кількість можливих значень символу, що вводиться. Зростання ефективності при використанні таємних біометричних образів, яке можливо тільки для динамічних методів, – очевидно [37-39, 45-46].

За методом реалізації біометричні пристрої можна розділити на: *програмні*; *апаратні*. Будь-який біометричний пристрій включає в себе і апаратуру, і програму. *Програмна* біометрична система – використовує стандартні пристрої комп'ютера (мишка, мікрофон, клавіатура) і програму для управління цим пристроєм та обробки отриманої з нього інформації. *Апаратна* біометрична система – для роботи, крім програми, необхідні спеціальні пристрої (спеціальний сканер, відеокамера) [37-39].

За кількістю механізмів захисту в одній системі захисту, пристрої захисту можна розділити на: *унарні*; *комбіновані*. *Унарна* система – використовує тільки один механізм захисту. *Комбінована* система – включає в себе кілька механізмів автентифікації (наприклад, відбиток пальця + смарт-карта + Pin-код). При виборі між унарною і комбінованою системою захисту треба керуватися, тим що саме важливіше, забезпечити пропуск «свого», або запобігти пропуску «чужого». Якщо в системі головне не пропустити «чужого», тоді, краще застосовувати комбіновані системи захисту, тому що в цьому випадку збільшується імовірність того, що система не пропустить «чужого» (система не пропустить «чужого», якщо хоч один з механізмів комбінованої системи захисту відмовить йому в доступі). Формула, за якою обчислюється імовірність відсутності помилки другого роду в комбінованій системі захисту має вигляд: $P_{2S} = 1 - P_{o2_1} \cdot P_{o2_2} \cdot ... \cdot P_{o2_i} \cdot ... \cdot P_{o2_n}$, де P_{2S} – імовірність відсутності помилки другого роду комбінованою системою захисту, P_{o2_i} – імовірність помилки другого роду і-го механізму комбінованої системи захисту, n – кількість механізмів захисту в комбінованій системі. Але якщо в системі головне пропустити «свого», тоді краще застосовувати унарні системи захисту, так як при застосуванні комбінованих систем захисту зменшується імовірність пропуску «свого». Це пояснюється тим, що система пропустить «свого», тільки в тому випадку, якщо всі механізми комбіно-

ваної системи захисту дозволять йому доступ. Отже, формула, за якою обчислюється імовірність відсутності помилки першого типу в комбінованій системі захисту має вигляд: $P_{1S} = (1 - P_{o1_1}) \cdot (1 - P_{o1_2}) \cdot ... \cdot (1 - P_{o1_i}) \cdot ... \cdot (1 - P_{o1_n})$, де P_{1S} – імовірність запобігання помилки першого роду комбінованою системою захисту, P_{o1_i} – імовірність помилки першого роду і-ої механізму комбінованої системи захисту [37-39].

За функціональностю (місце розташування) БС можна розділити на: *внутрішні*; *проміжні*; *зовнішні*. *Внутрішні* пристрої розташовуються усередині об'єкта, що захищається, наприклад сканер відбитка пальця, вбудований в мишку від комп'ютера який захищається. *Зовнішні* пристрої розташовуються зовні об’єкта який захищається, наприклад сканер відбитка пальця, вбудований в замок в двері охороняємого приміщення. *Проміжні* пристрої розташовуються зовні внутрішнього приміщення, але біля них періодично знаходяться або проходять повз, люди, які можуть перешкодити незаконному проходженню через дану біометричну систему захисту. Незалежно від типу біометричної системи захисту імовірність захищеності об'єкту, що охороняється залежить від: вірогідності правильної автентифікації (P); вірогідності того, що дану систему захисту не можна пройти за допомогою підбору потрібних значень біометричних характеристик, що аналізуються (P_p); вірогідності того, що дану систему захисту не можна фізично вивести з ладу (P_F). Таким чином, формула імовірності захищеності об'єкта, який захищається матиме вигляд: $P_z = P \cdot P_p \cdot P_F$. Значення P не залежить, від місця розташування даної біометричної системи захисту, а значення P_p і P_F досить сильно залежать від даного параметра. Внутрішню систему важко подолати за допомогою підбору або за допомогою виведення її з ладу, тому що біля неї постійно присутні люди. Зовнішню систему порушнику легше подолати, тому що над нею нема постійного контролю. Імовірність того, що порушник зможе проникнути на об'єкт, який захищається біометричною системою захисту проміжного типу, залежить від того, як часто повз цію систему захисту проходять люди, які можуть перешкодити зловмиснику. Імовірність того, що систему не можна пройти за допомогою підбору потрібних значень біометричних характеристик, що аналізуються можна обчислити за

формулою: $P_p = f_p(t_p, kp)$, де t_p – час, необхідний користувачу для підбору потрібних значень біометричних характеристик, що аналізуються, за допомогою перебору всіх можливих значень, kp – кількість людей, що проходять за одну секунду біля даної системи захисту і можуть перешкодити порушнику, f_p – функція залежності P_p від t_p і kp. Імовірність того, що дану систему захисту не можна фізично вивести з ладу, можна обчислити за формулою: $P_F = f_F(t_F, kp)$, де t_F – час, необхідний користувачу для фізичного виведення системи захисту з ладу, f_F – функція залежності P_F від t_F і kp. Треба зазначити, що більшість методів, які використовуються в цих системах можуть застосовуватися і у внутрішніх, і в зовнішніх, і в проміжних системах [37-39].

За призначенням БС, як правило, діляться на: ті, *що регулюють доступ до комп'ютерної інформації*; *класичні системи контролю управління доступом (СКУД)*. БС, *що регулюють доступ до комп'ютерної інформації* захищають безпосередньо сам комп'ютер, інформацію, збережену в ньому і інші інформаційні ресурси. Призначення *класичних СКУД*, як правило, обмежується захистом особливо відповідальних приміщень. Деякі біометричні методи можуть використовуватися в системах обох типів (за видбитком пальця), а деякі зазвичай використовуються в системах тільки одного з типів (за клавіатурним почерком) [37-39].

За автономностю БС можна розділити на: *автономні* (stand alone); *інтегровані* (що вбудовуються). *Автономні* системи поєднують в собі і пристрій розпізнавання людини, і пристрій, що реалізує функцію надання доступу до об’єкту, який захищається при позитивному результаті розпізнавання (пристрої, що поєднують біометричні сканери та електромеханічні замкові пристрої). *Інтегровані* пристрої вбудовуються в уже існуючі системи контролю управління доступом і інтегровані системи безпеки, тобто вони виконують тільки сканування певних біометричних характеристик людини, а вже інші пристрої, в які інтегруються дані пристрої, розпізнають людину за відсканованими параметрами і виконують якусь дію за результатами розпізнавання, наприклад, відкривають двері, або блокує доступ до до об’єкту (сканер відбитка пальця, вбудований в мишку) [37-39].

За способом надання доступу біометричні пристрої можна розділити на:

локальні; *мережеві*. При успішному проходженні через *локальну* систему захисту користувач отримує доступ до одного конкретного об'єкту, який захищається. При успішному проходженні через *мережеву* систему захисту користувач відразу отримує доступ з певними правами до кількох обєктів що захищаються, тобто до мережі. Імовірність захищеності біометричною системою захисту мережі менша, ніж імовірність захищеності локального комп'ютера, тому що імовірність захищеності локального комп'ютера залежить тільки від імовірності захищеності даною біометричною системою даного об'єкта, а в разі мережі, доступ до всіх комп'ютерів даної мережі можна отримати при проходженні захисту на будь-якому комп'ютері цієї мережі. Таким чином, якщо об'єктом, який захищається є мережа з одного комп'ютера (локального комп'ютера), тоді імовірність захищеності цього об'єкта можна обчислити за формулою: $P_z = P_l$, де P_l – імовірність захищеності локального комп'ютера. Якщо об'єктом, який захищається є мережа з n комп'ютерів, тоді імовірність захищеності цього об'єкта можна обчислити за формулою:

$$P_z = \frac{P_{l1} + P_{l2} + ... + P_{li} + ... + P_{ln}}{n},$$

де P_{li} – імовірність захищеності i-го комп'ютера, n – кількість комп'ютерів в мережі. Тому вимоги відносно імовірності захищеності об'єкта до мережевих біометричних систем повинні бути значно вищі, ніж до локальних біометричних систем [37-39].

За типом управління біометричні пристрої можна розділити на: *локальні*; *мережеві*. *Локальні* – управління знаходится всередині системи. *Мережеві* –управління відбувається з віддаленого пристрою управління (сервера). При віддаленому керуванні виникає додаткова вразливість – при передачі по мережі біометричного зразка він може бути перехоплений (може викликати помилку другого роду) або незаконно змінений (може викликати помилку першого роду). Тому в таких системах захисту треба передбачати додатковий захист при передачі зразка мережею [37-39].

За багаторазовостю виконання розпізнавання БС можна розділити на системи: *з одноразовим розпізнаванням*; *з багаторазовим розпізнаванням*. Системи *з одноразовим розпізнаванням* ідентифікують користувача тільки тоді, коли він намагається отримати доступ до ресурсів (або на територію), тобто таке розпізнавання викону-

ється тільки один раз, а далі в процесі роботи користувач даним пристроєм більше не перевіряється. Системи *з багаторазовим розпізнаванням* здійснюють моніторинг контролю вже в процесі роботи. Виконання моніторингу корисне у випадку, якщо авторизований користувач в процесі роботи залишить комп'ютер без нагляду і цим скористається порушник. Крім того, моніторинг необхідний в тих організаціях, в яких важлива реакція і уважність персоналу, наприклад, для диспетчерів в аеропортах. Моніторинг краще проводити непомітно для людей за якими спостерігається. Для проведення непомітного моніторингу підійдуть не всі біометричні методи, найкраще використовувати, в даному випадку, системи розпізнавання за клавіатурним почерком (контролювати певні параметри при наборі слів або сполучень символів, які часто повторюються); за відбитком пальця, якщо пристрій вбудовано в стандартну мишку або клавіатуру; за райдужною оболонкою ока за допомогою поруч розташованої відеокамери. Для систем з багаторазовим розпізнаванням, на відміну від систем з одноразовим розпізнаванням, важливо, щоб при інших рівних це були досить швидкісні пристрої розпізнавання [37-39].

За швидкістю автентифікації БС можна розділити на: *більш швидкісні* (до 1 секунди); *менш швидкісні* (1-5 секунд). Час, необхідний для вирішення системою задачі автентифікації користувача залежить від часу отримання (сканування) біометричних характеристик користувача і від часу розпізнавання (верифікації), тому формула для визначення часу, необхідного системі для вирішення задачі автентифікації, має вигляд: $t = t_r + t_v$, де t_r - час сканування біометричних характеристик користувача, t_v - час розпізнавання. Швидкість роботи пристрою залежить від: технології розпізнавання; алгоритму розпізнавання; технології сканування біометричних характеристик; розміру області сканування; кількості біометричних характеристик, які використовуються для автентифікації. Якщо процес автентифікації планується виконувати один раз – при вході в будь-який захищений простір (комп'ютер, приміщення), тоді фактор швидкості автентифікації грає відносно не велике значення, а якщо автентифікація буде виконуватися періодично протягом всієї роботи, тоді значення швидкості системи автентифікації має досить великий вплив при виборі конкретної

біометричної системи автентифікації [37-39].

За роздільною здатністю БС (в системах, в яких цей параметр присутній) можно розділити на: системи *з більшою роздільною здатністю*; системи *з меншою роздільною здатністю*. Чим більша роздільна здатність зображення, що сканується, тим більше якість сканування і, відповідно, більша імовірність правильної автентифікації, але і більше пам'яті треба для зберігання і обробки цього зразка [37-39].

За максимально можливим об’ємом інформації, що зберігається, БС діляться на: системи з *більшим максимальним об’ємом*; системи з *меншим максимальним об’ємом*. Розмір необхідної пам'яті для зберігання одного зразка залежить від розміру області сканування; від кількості аналізованих ознак; від роздільної здатності. При виборі конкретної біометричної системи захисту також має значення максимально можлива кількість реєстрованих зразків. По-перше, чим більше база даних біометричних параметрів, тим точніше буде працювати система захисту. А по-друге, цей критерій вибору конкретної біометричної системи захисту має велике значення для організацій з великою кількістю працівників, які повинні проходити автентифікацію. При збільшенні кількості збережених зразків збільшується необхідна для їх зберігання і обробки пам'ять [37-39].

За частотою використання БС можна розділити на: системи *з більшою частотою використання* при побудові систем захисту; системи *з меншою частотою використання* при побудові систем захисту [37-39].

1.2.3. Способи підвищення ефективності застосування біометричних систем автентифікації

Перед створенням будь-якої системи захисту необхідно оцінити можливу ефективність від її застосування і доступні способи підвищення якості цієї системи. Показники ефективності застосування біометричних систем автентифікації були розглянуті в пункті 1.2.1 даного розділу, а зараз проаналізуємо від чого залежать ці показники та розглянемо способи їх покращення [38-39].

На ефективності застосування біометричних систем автентифікації впливають:

1. Якість параметрів, що аналізуються аналізованих.

2. Кількість параметрів, що аналізуються.

3. Кількість навчальних зразків.

4. Метод (технологія) розпізнавання.

5. Якість пристрою і програми, які цей метод реалізують.

Чим більше в зразку параметрів високої якості, тим ефективніше застосування даної системи автентифікації. Тому в біометричній системі захисту, для вибору характеристик найбільш придатних для розпізнавання, необхідна наявність функції визначення якості цих параметрів, тому що для різних груп користувачів ці параметри можуть відрізнятися. Так, наприклад, в разі автентифікації за клавіатурним почерком, для однієї групи користувачів може бути характерним час набору символів одного, а для іншої групи – час набору символів іншого слова.

Один із критеріїв якості параметрів, що аналізуються, можна обчислити за формулою :

$$q = \sqrt{\frac{(m_{X_i} - m_{Y_i})^2}{\sigma_{X_i}^2 + \sigma_{Y_i}^2}},$$

де X_i – множина значень i-ого параметра зразків легального користувача, а Y_i – множина значень i-ого параметра зразків порушника; m_{X_i} – математичне очікування значень i-ого параметра множини X_i; m_{Y_i} – математичне очікування значень i-ого параметра множини Y_i; σ_{X_i} – дисперсія значень i-ого параметра множини X_i; σ_{Y_i} – дисперсія значень i-ого параметра множини Y_i. Тобто можна сказати, що якість i-ого параметра залежить від співвідношення дисперсій та від відстані між центрами множин, що розділяються [124-125]. Чим менше значення дисперсій і чим більша відстань між центрами множин, що розділяються, тим краще якість даного параметра. Якщо обчислювальні ресурси обмежені, тоді для визначення якості параметра, що аналізується, можна використовувати наступну формулу:

$$q = \frac{\left| m_{X_i} - m_{Y_i} \right|}{\sigma_{X_i} + \sigma_{Y_i}}.$$

Чим більше q , тим вище якість параметра, що аналізується.

Оптимальна кількість зразків і кількість ознак в них залежить не тільки від ме-

тоду автентифікації, але і від конкретних користувачів, які будуть розпізнаватися даною системою захисту. Тому що у різних людей різні ступені унікальності і стабільності тих чи інших біометричних характеристик. Тому в кожній системі захисту повинна бути підсистема для їх визначення в кожному конкретному випадку.

Для забезпечення ефективності застосування біометричних систем автентифікації, біометричним даним повинні бути притаманні наступні властивості:

1. Невід'ємність.
2. Непідробність.
3. Унікальність.
4. Стабільність.

Для різних біометричних методів і систем ступінь унікальності, стабільності, невід'ємності та непідробності біометричних характеристик може змінюватися в досить великому діапазоні.

Для статистичних систем характерна більш високий ступінь унікальності і стабільності біометричних даних при їх відносно невисокій невід'ємності і непідробності. Динамічні БС зазвичай мають меншу унікальність і стабільність біометричних даних, але при відповідних заходах, більш захищені з точки зору невід'ємності і непідробності біометричних даних. Менша унікальність і стабільність характеристик існуючих динамічних біометричних систем аргументується тим, що значення біометричних характеристик, в даному випадку, залежать від рухових навичок, які є сукупним усередненням результатів роботи великої кількості м'язів, що задіяні при виконанні цих рухів. Вплив великої кількості малих рухових навичок зазвичай призводить до нормалізації особливостей руху, а вплив окремих великих м'язів може привести до відхилення від цих усереднених особливостей. Тому якщо у виконанні руху бере участь велика кількість малих м'язів, то рух, швидше за все, буде більш стабільним, але менш унікальним, ніж в разі, коли в русі беруть участь окремі великі м'язи [38-39].

Так як ступінь унікальності і стабільності біометричних характеристик в динамічних біометричних системах визначається, більшою мірою, унікальністю і стабільністю самого руху, то воно повинно бути звично для людини яка розпізнається і незвично для фальсифікатора. І відповідно, такі БС доцільно використовувати там,

де більшість людей мають стійкі та індивідуальні відповідні навички.

У деяких випадках ступінь унікальності можна штучно підвищити. Так, наприклад, якщо для розпізнавання використовується рукописний почерк, то для підвищення унікальності можна в автограф (або слово-пароль) ввести розчерки, накладення букви на букву, зміну прийнятого порядку написання букв.

Незважаючи на існуючі недоліки методів біометричної автентифікації, вони надійніші, ніж більшість інших, більш відомих методів автентифікації. Переваги методів біометричної автентифікації [38-39]:

1. Зразок біометричної характеристики, що аналізується, зазвичай, піддається односторонньому перетворенню, тобто з нього не можна назад відновити відбиток пальця або малюнок райдужної оболонки ока.

2. «Біометричний пароль» завжди при людині, а при використанні парольного або криптографічного захисту користувач повинен запам'ятовувати пароль (або криптографічний ключ) або носити на будь-якому носії (смарт- карті, таблетці touch-memory тощо.).

3. «Біометричний пароль» більш унікальний, хоча, як вже було сказано, його унікальність, в залежності від використовуваних для автентифікації біометричних характеристик і їх кількості, сильно відрізняється.

Навіть динамічні методи біометричної автентифікації, яким властива менша, ніж статичним методам, унікальність характеристик, що аналізуються, надійніші, ніж парольний захист, особливо при використанні таємних біометричних характеристик [45-46]. Наприклад, недостатньо надійний пароль з 5 цифр і букв, набраний в двох регістрах містить $90^5 \approx 10^{10}$ можливих комбінацій. Якщо цей же пароль вводити в комп'ютер за допомогою пристрою введення рукописної графічної інформації (наприклад, за допомогою графічного планшета) і аналізувати крім самого введеного пароля ще і почерк, тоді стійкість до перебору паролів істотно збільшується, тому що зазвичай БС можуть надійно розрізняти близько 100 різних почерків, що відтворюють одні й ті ж букви. Це рівнозначно тому, що вихідний алфавіт збільшується приблизно в 100 разів. Тоді число можливих комбінацій пароля з 5 символів складе $(90*100)^5 \approx 10^{20}$. Ці показники можна ще поліпшити, якщо використовувати техноло-

гії аналізу тривимірних рукописних образів, які враховують дві координати $Y(t)$, $X(t)$ і третю координату - тиск пера на графічний планшет - $Z(t)$. Це еквівалентно збільшенню вихідного алфавіту вже не в 100 разів, а приблизно в 1000. При цьому число можливих комбінацій пароля з 5 знаків буде дорівнювати $(90*1000)^5 \approx 10^{25}$. При цьому відповідно значно зростає час, який необхідно витратити на перебір всіх можливих комбінацій цього пароля. Таким чином, можна сказати, що з'єднання унікальності і таємниці біометричних паролів може забезпечити майже будь-яку ступінь безпеки інформації. Якщо використовувати відкриті біометричні характеристики, тоді імовірність пропуску порушника значно збільшується. Однак можливість використовувати таємні біометричні показники є тільки у динамічних методів. Крім того, при використанні динамічних методів, у випадку витоку інформації фразу пароль можна поміняти, а при використанні статичних методів це неможливо; перехоплення відкритого статичного параметра (параметра, доступного для спостереження усіма) важче помітити і його важче уникнути. Таким чином, незважаючи на те, що статичні методи значно дорожче динамічних, вони менш ефективніші.

1.2.4. Аналіз існуючих методів розпізнавання користувачів інформаційних систем за їх клавіатурним та рукописним почерком

Виконаємо порівняльний аналіз найбільш поширених методів розпізнавання (з відкритих джерел) за обраними біометричними характеристиками (табл.1) [70-98].

Проаналізовані наступні методи розпізнавання, які засновані на: АФЩРІВЗ – аналізі функції щільності розподілу імовірностей випадкових змінних; ВКВБ – використанні класифікатора з векторним базисом; ВШНМ – використанні штучних нейронних мереж; АСХ – аналізі статистичних характеристик; АСХФПФ – аналізі статистичних характеристик для фіксованої парольної фрази; АМОД – аналізі математичного очікування та дисперсії; АМОДДЗ – аналізі математичного очікування та дисперсії, з виконанням додаткового зважування; ВРКС – використанні рангової кореляції Спірмена; ВКР – використанні корекції рангів; АРКНПРЗК – аналізі ритму клавіатурного набору з пороговим завданням класів; АРКНПЗК – аналізі ритму клавіатурного набору з пропорційним завданням класів; ВКНС – використанні К-найближчих сусідів; ВБП – використанні багатошарового персептрона; ТААУ – техно-

логії аналізу Амфреса Д. і Вільямса Г.; ТАІ – технології аналізу Іванова; ВПММ – використанні прихованих Марковських моделей; ВБРНМ – використанні багатовимірних рекурентних нейронних мереж; FSRP – метод синтаксичного аналізу; ТАГК – технології аналізу Генуе Р. і Кечаді Т.; ВПНК – використанні параметричного навчання класифікатора. Данні методи проаналізовані за наступними характеристиками: Т – біометрична характеристика, яка використовується для автентифікації; О – імовірність помилкової відмови доступу легальному користувачеві; Н – імовірність помилкового надання доступу порушникові; Д – необхідна довжина зразка почерку; Ф – використання для аналізу функціонального стану користувача; Х – використання для аналізу характеристик, необхідних працівникам критичних професій (уважності, сконцентрованості, акуратності); Б – використання в якості етапу багатофакторної автентифікації; А – необхідність додаткового апаратного забезпечення; Л – невід'ємність; П – непідробленість; У – унікальність; К – стабільність. Деякі позначення значень характеристик: В – велика; С – середня; М – маленька.

Таблиця 1

Порівняння найбільш поширених методів розпізнавання

Характеристика / Метод	Т	О	Н	Д	Ф	Х	Б	А	Л	П	У	К
АФЩРІВЗ	КП	М	М	В	-	-	-	-	В	С	С	С
ВКВБ	КП	В	М	В	-	-	-	-	В	С	С	С
ВШНМ	КП	В	М	С	-	-	-	-	В	С	С	С
АСХ	КП	С	С	В	-	-	-	-	В	С	С	С
АСХФПФ	КП	С	В	С	-	-	-	-	В	С	С	С
АМОД	КП	М	М	В	-	-	-	-	В	С	С	С
АМОДДЗ	КП	М	М	В	-	-	-	-	В	С	С	С
ВРКС	КП	М	М	В	-	-	-	-	В	С	С	С
ВКР	КП	М	М	В	-	-	-	-	В	С	С	С
АРКНПРЗК	КП	М	М	В	-	-	-	-	В	С	С	С
АРКНПЗК	КП	М	М	В	-	-	-	-	В	С	С	С
ВКНС	КП	В	М	С	-	-	-	-	В	С	С	С
ВБП	КП	М	В	С	-	-	-	-	В	С	С	С
ТААВ	КП	В	С	В	-	-	-	-	В	С	С	С
ТАІ	КП	С	С	М	+	-	+	-	В	С	С	С
ВПММ	РП	В	В	В	-	-	-	+	В	С	С	С
ВБРНМ	РП	С	С	М	-	-	-	+	В	С	С	С
FSRP	РП	С	С	С	-	-	-	+	В	С	С	С
ТАГК	РП	М	М	С	-	-	-	+	В	С	С	С
ВПНК	РП	М	М	С	-	-	-	+	В	С	С	С

1.3.Нейронні мережі. Порівняння нейронних мереж різний типів

Нейронная мережа являє собою сукупність елементів, які з’єднані між собою певним чином так, щоб між ними забезпечувалось взаємодія. Ці елементи (рис.4), які називаються нейронами або вузлами, представляють собою прості процесори,

обчислювальні можливості котрих зазвичай обмежуються деяким правилом комбінування вхідних сигналів і правилом активізації, яке дозволяє обрахувати вихідний сигнал за сукупністю вхідних сигналів. Вихідний сигнал елемента може посилатися іншим елементам по зваженим зв'язкам, з кожним з котрих пов'язаний ваговий коефіцієнт або вага. В залежності від значення вагового коефіцієнту сигнал, що передається, або посилюється або послаблюється [126-191].

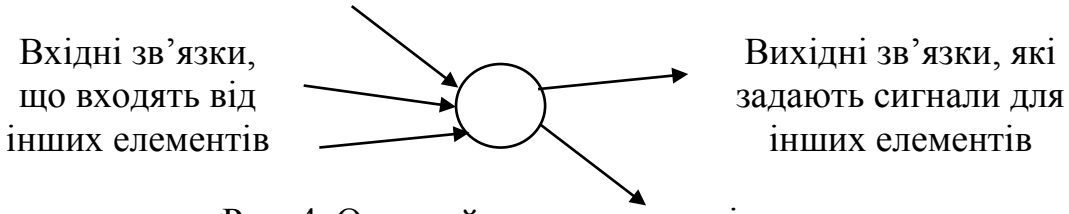
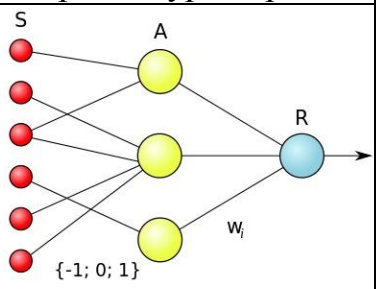
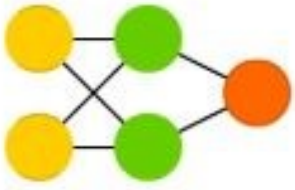


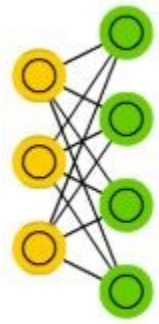
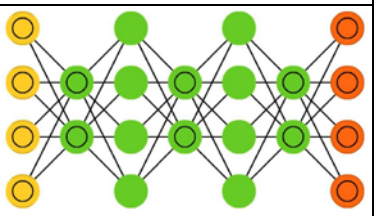
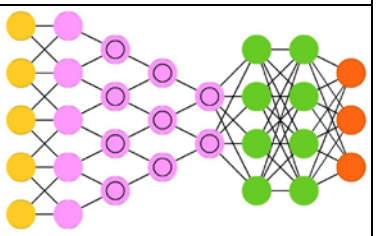
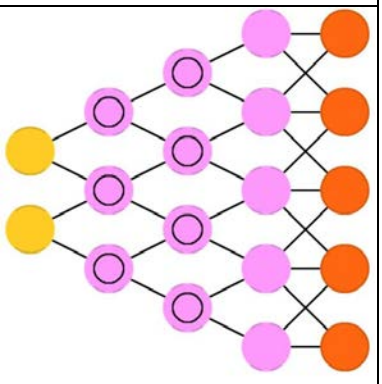
Рис. 4. Окремий елемент мережі

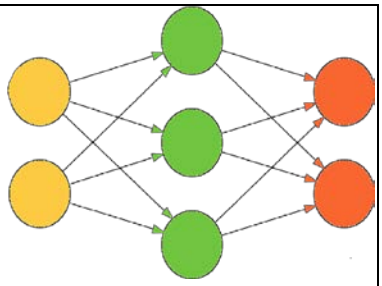
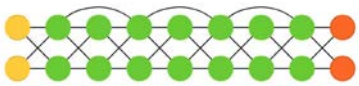
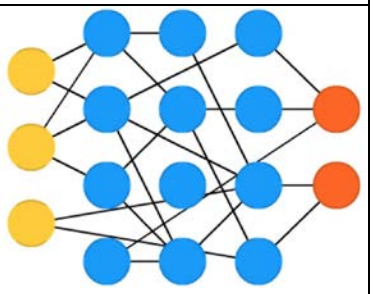
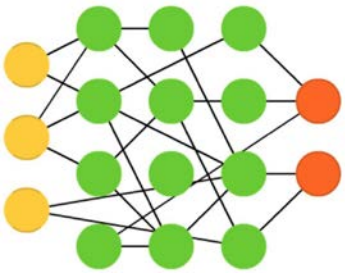
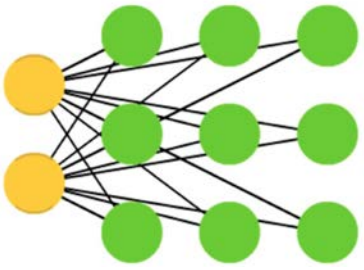
Розглянемо і порівняємо деяки типи нейронних мереж [126-191], які можно застосовувати для вирішення задачі класифікації об'єктів [192-198], до якої можна звести задачу автентифікації користувачів .

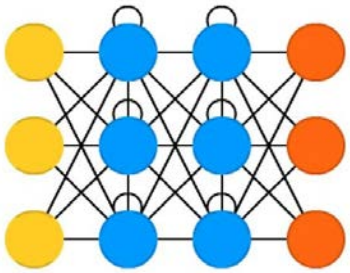
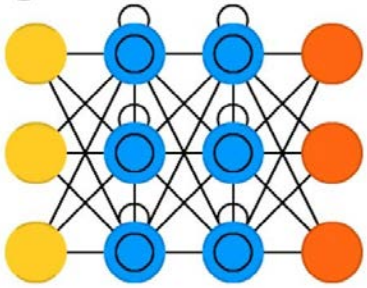
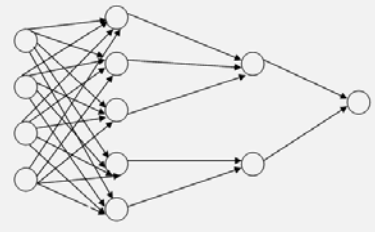
Таблиця 2

Порівняння нейронних мереж різний типів

Опис мережі	Архітектура мережі
Назва: Перцептрон Рік появи: 1957 Кількість шарів: 1 або декілька Призначення: Прогнозування, розпізнавання образів, керування агентами та інші. Алгоритм навчання: Навчання з корекцією похибки, алгоритм зворотного поширення похибки Переваги: Програмні та апаратні реалізації моделі дуже прості. Простий і швидкий алгоритм навчання Недоліки: Примітивні поділяючі поверхні (гіперплощини) дають можливість вирішувати лише найпростіші задачі розпізнавання.	
Назва: Мережа радіально -базисних функцій Рік появи: 1988 Кількість шарів: 2+ Призначення: Функції наближення, прогнозування часових рядів, класифікація і системи управління Алгоритм навчання: Генетичні алгоритми Переваги: Моделюють нелінійну функцію за допомогою всього 1 проміжного шару. Параметри лінійної комбінації в вихідному шарі можна повністю оптимізувати за допомогою методів лінійної оптимізації, які працюють швидко і не викликають труднощів з локальними мінімумами	

Недоліки: Мають погані екстраполюючі властивості і виявляють-ся досить об'ємними при великій розмірності вектора входів	
Назва: Обмежена машина Больцмана Рік появи: 1986 Кількість шарів: 1+ Призначення: Завдання зниження розмірності даних, завдання класифікації, коллаборативна фільтрація, виділення ознак і тематичне моделювання Алгоритм навчання: Алгоритм контрастивної дивергенції Переваги: Може застосовуватись для моделювання невідомого заздалегідь розподілу. Після навчання мережа здатна працювати з неповними даними і доповнювати їх за необхідності. Недоліки: Тривалість процесу навчання - семплування за Гіббсом є повільним процесом	
Назва: Глибинні мережі переконань Рік появи: 2006 Кількість шарів: 1+ Призначення: Глибинне навчання Алгоритм навчання: Алгоритм контрастивної дивергенції Переваги: Швидке навчання Недоліки: -	
Назва: Згорткові нейронні мережі Рік появи: 1988 Кількість шарів: 3+ Призначення: Обробка зображень, аналіз відео, програмування Алгоритм навчання: Навчання з вчителем, алгоритм кластерізації, логістична регресія, дерево рішень Переваги: Один з найкращих алгоритмів для розпізнання зображень Недоліки: Забагато варійованих параметрів мережі, які складно конфігурувати	
Назва: Розгорткові нейронні мережі Рік появи: 1991 Кількість шарів: 3+ Призначення: У наукових цілях Алгоритм навчання: Початкова мережна модель може зробити первинне навчання, а інша модель може візуально сегментувати цільове зображення Переваги: Гарний алгоритм для розпізнавання зображень, гарно використовуються у програмуванні Недоліки: Складне налаштування	

Назва: Двонаправлені періодичні нейронні мережі Рік появи: 1997 Кількість шарів: 1+ Призначення: Розпізнавання мовлення чи рукописного введення, також широко задіюються для перекладу Алгоритм навчання: Подібні алгоритми в порівнянні з рекурентними мережами Переваги: Наявність контексту вхідних даних покращує результат. Не вимагає фіксації вхідних даних Недоліки: Великі обчислювальні затрати, схильність до переналагодження	
Назва: Глибокі залишкові мережі Рік появи: 1996 Кількість шарів: 1+ Призначення: Розпізнавання зображень Алгоритм навчання: Навчання з корекцією похибки Переваги: Можливе використання великої кількості шарів Недоліки: Час на обчислення зі збільшенням шарів стрімко зростає	
Назва: Нейронна ехо-мережа Рік появи: 1990 Кількість шарів: 1+ Призначення: Класифікація тактильних даних Алгоритм навчання: Асимптотичні Переваги: Добре відтворюють тимчасові ряди Недоліки: Велика залежність від початкових установок	
Назва: Машина екстремального навчання Рік появи: 2001 Кількість шарів: 1+ Призначення: Задачі регресії та класифікації Алгоритм навчання: Алгоритми схожі до алгоритмів перцептрона, але зв’язки випадкові Переваги: Добре працює з даними, що генеруються в реальному часі Недоліки: Імовірнісний характер мережі впливає на стабільність роботи	
Назва: Нейронні мережі Кохонена Рік появи: 1963 Кількість шарів: 3+ Призначення: Розпізнавання рукописного тексту Алгоритм навчання: Метод адаптивних лінійних суматорів Переваги: Можливо досягти високої точності Недоліки: Необхідна велика вибірка вхідних даних для кращого результату	

Назва: Рекурентні нейронні мережі Рік появи: 1990 Кількість шарів: 3+ Призначення: Застосовуються в розпізнаванні і обробці текстових даних Алгоритм навчання: Метод зворотного поширення в часі Переваги: Вище швидкість роботи; точність розпізнавання більше; краще працюють в умовах підвищеного шуму; добре працює в умовах неточності та незавершеності промовлених слів Недоліки: Вимагає великих обчислювальні потужності; необхідна велика кількість прикладів для навчання; треба багато часу для навчання	
Назва: Довга короткочасна пам'ять Рік появи: 1997 Кількість шарів: 3 Призначення: Стиснення тексту природною мовою, розпізнавання несегментованого неперервного рукописного тексту. Алгоритм навчання: Метод зворотного поширення в часі з деякими покращеннями Переваги: Добре підходить для навчання в умовах, коли між важливими подіями існують часові затримки невідомої тривалості. Недоліки: Умовою продуктивної праці є те, що вона повинна мати належну матрицю вагових коефіцієнтів, що являється непростою задачею	
Назва: Імовірнісна нейронна мережа Рік появи: 1990 Кількість шарів: 4 Призначення: Класифікація об’єктів Алгоритм навчання: Модель з керованим навчанням Переваги: Зручно використовувати для класифікації Недоліки: Може виконувати тільки класифікацію, вимогливі щодо ресурсів	

Розглянемо більш детально **імовірнісні нейронні мережі** (мережа PNN – Probabilistic Neural Network) можна використовувати для класифікації зразків. Мережа ІНМ являє собою паралельну реалізацію відомих статистичних методів. Розглянемо принцип роботи такої мережі [126-191].

В основу класифікації в мережі PNN покладено використання методів Байеса (Bayes). Сенс полягає в тому, що для кожного зразка можна прийняти рішення на основі вибору найбільш імовірного класу з тих, яким міг би належати цей зразок. Таке рішення вимагає оцінки функції щільності імовірностей для кожного класу. Для оцінки функції щільності розподілу ймовірностей необхідна використовують метод Пар-

зена (Parzen), в якому використовується вагова функція, що має центр в точці, що представляє навчальний зразок. Ця вагова функція називається функцією потенціалу або ядром. Найчастіше в якості ядра використовується функція Гаусса [124-125].

Розглянемо принцип побудови ІНМ (рис.5) [126].

Архітектура мережі визначається структурою навчальних даних:

- ♦ число вхідних елементів дорівнює числу ознак;
- ♦ число елементів шару зразків дорівнює числу навчальних зразків;
- ♦ число елементів шару підсумовування дорівнює числу класів.
- ♦ число елементів вихідного шару завжди дорівнює 1.

Вхідний шар розподіляє дані вхідного зразка для шару зразків, тобто вхідний шар має стільки елементів, скільки ознак. Шар зразків має по одному елементу для кожного зразка з набору навчальних даних. Вхідний шар і шар зразків утворюють повнозв'язну структуру. Для вхідних в елемент шару зразків зв'язків вагові значення встановлюються рівними елементам відповідного вектора-зразка, тобто для першого елемента шару зразків значення його першої вхідної ваги буде встановлено рівним значенню першого елемента першого вектора зразка, значення другої вхідної ваги другого елемента вектора і т. д. Якщо всі вхідні вектори мають одиничну довжину, тобто є нормалізованими (сума квадратів елементів вектора дорівнює одиниці), то функція активності для елемента шару зразків обраховується за наступною формулою:

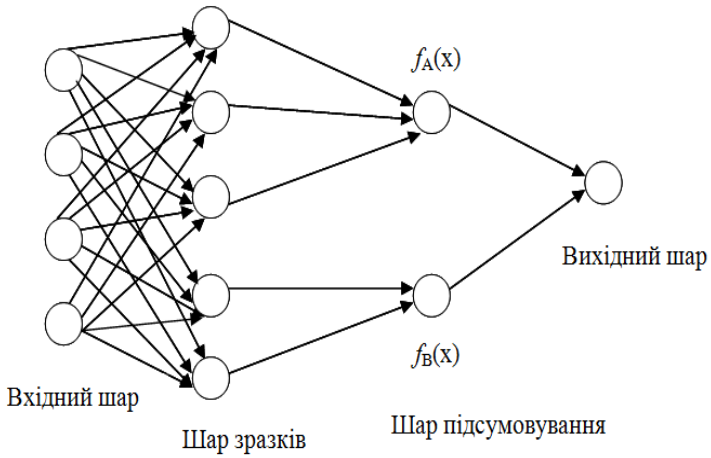


Рис.5. Приклад архітектури мережі ІНМ

$$O_j = \exp\left(\frac{\sum x_i w_{ij} - 1}{\sigma^2}\right),$$

де x – невідомий вхідний зразок. Шар підсумовування має по одному елементу для кожного класу з навчальної множини даних. До будь-якого елементу шару підсумо-

вування йдуть зв'язки тільки від елементів шару зразків, що належать відповідному класу. Вагові значення зв'язків, що йдуть від елементів шару зразків до елементів шару підсумовування, встановлюються рівним 1. Елемент шару підсумовування складає вихідні значення елементів шару зразків. Отримана сума дає оцінку значення функції щільності розподілу імовірностей для всього набору екземплярів відповідного класу. Вихідний елемент являє собою дискримінатор порогової величини, що вказує елемент шару підсумовування з максимальним значенням, тобто вказує до якого класу найбільш імовірно належить невідомий екземпляр [126].

До переваг ІНМ можно віднести [126]:

1. Для них не потрібне навчання так, як потрібне для мереж зі зворотним поширенням помилок, тому що всі параметри мережі ІНМ, визначаються безпосередньо навчальними даними.

2. Швидко навчаються.

3. Дуже зручно використовувати для класифікації.

4. Допускають наявність помилкових даних.

5. Забезпечують хороші результати навіть на малих наборах даних.

Проаналізувавши все вищесказане, можна підсумувати: ІНМ може використовуватися для класифікації зразків. Тому вона і була обрана для подальшого дослідження і використання для вирішення задачі розпізнавання користувачів.

1.4. Визначення основних задач та напрямків наукового дослідження дисертаційної роботи

В результаті всього вищесказаного сформульована наступна мета дисертаційної роботи: підвищення імовірності правильного розпізнавання користувачів ІС за рахунок розробки нових методів автентифікації користувачів, які використовують для розпізнавання біометричні характеристики користувача, застосовуючи нейронні мережі для ідентифікації його біометричного образу.

Для досягнення поставленої мети були сформульовані наступні задачі:

1. На основі проведеного аналізу існуючих біометричних методів розпізнавання та програмних і апаратних засобів на їх основі, обрати методи автентифікації користувачів ІС, застосування яких забезпечить задану імовірність правильного розпіз-

навання користувачів та не потребуватиме суттєвих витрат на впровадження.

2. Розробити методи первинної обробки навчальних даних, використання яких дозволить збільшити імовірність правильного розпізнавання користувачів ІС та зменшить затрачувані ресурси.

3. Розробити методи автентифікації користувачів ІС, які базуються на обраних біометричних методах розпізнавання і використовують для цього обраний різновид нейронної мережі.

4. Створити програмне забезпечення, яке на основі розроблених методів розпізнавання користувачів, виконуватиме автентифікацію користувачів ІС за обраними біометричними характеристиками, та по-перше, оцінюватиме імовірність їх правильного розпізнавання, по-друге, на основі аналізу накопичених даних даватиме змогу здійснити вибір конфігураційних параметрів систем розпізнавання.

Для вирішення поставлених задач використовувались наступні методи дослідження: комп'ютерне моделювання, статистичний аналіз, теорія імовірності, лінійна алгебра, методи емпіричних та теоретичних досліджень, комбінаторний аналіз, об'єктно-орієнтовані інформаційні технології, математичний апарат нейронних мереж.

1.5. Висновки до першого розділу

В даному розділі спочатку проаналізовані існуючи типи АК ІС. Потым удосконалена класифікація існуючих біометричних систем розпізнавання. На основі проведеної класифікації існуючих біометричних систем розпізнавання, їх порівняльного аналізу та аналізу біометричних методів, які використовують дані системи, здійснено вибір біометричних методів АК ІС, застосування яких забезпечує задану ІПР користувачів та не потребує суттєвих витрат на впровадження. Після чого проаналізовані існуючі методи розпізнавання користувачів ІС за їх КП та РП. Крім цього проаналізовані існуючі види нейронних мереж та обраний найбільш придатний їх різновид для вирішення задачі АК, а саме ІНМ. Наприкінці на основі проведеного аналізу визначені основні задачі та напрямки наукового дослідження, яким присвячена дисертаційна робота.

РОЗДІЛ 2. МЕТОД АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ ІНФОРМАЦІЙНИХ СИСТЕМ ЗА ЇХ КЛАВІАТУРНИМ ПОЧЕРКОМ ТА МЕТОД ПЕРВИННОЇ ОБРОБКИ ЗРАЗКІВ КЛАВІАТУРНОГО ПОЧЕРКУ

2.1. Постановка задачі автентифікації за клавіатурним почерком

Данний розділ присвячено розробці методу АК ІС за їх КП (АККП) та методу первинної обробки зразків КП (ПОЗКП), необхідного для виконання АК ІС за їх КП. В якості механізму розпізнавання використана ІНМ. Для передачі зразка КП користувача в комп'ютер, використовувалася клавіатура комп'ютера. Тобто в даному розділі будується функція *R_K* – функція розпізнавання людей за динамікою вводу пароля, який вводиться за допомогою клавіатури комп'ютера [99-116].

Для розробки методів АККП та ПОЗКП використовується наступна модель даних:

$$\boldsymbol{US\_K}=\{\bigcup_{p=1}^{m} US\_K_p\}=\{US\_K_1, US\_K_2, \ldots, US\_K_p, \ldots, US\_K_x \ldots, US\_K_m\};\ \ p=\overline{1,m};\ m$$

– кількість членів множини ***US_K***; ***US_K*** – множина користувачів, які можуть спробувати отримати доступ до системи, що захищається;

$$\boldsymbol{USL\_K}=\{\bigcup_{t=1}^{l} USL\_K_t\}=\{USL\_K_1,\ USL\_K_2,\ \ldots,\ USL\_K_t,\ \ldots,\ USL\_K_d,\ \ldots,\ USL\_K_l\};$$

$t=\overline{1,l};\ \ l$ – кількість легальних користувачів; ***USL_K*** – множина легальних користувачів даної системи; здійснюється умова, що $\boldsymbol{USL\_K} \subset \boldsymbol{US\_K}$;

US_K_X – користувач ІС, який проходить автентифікацію; здійснюється умова, що $US\_K_X \in \boldsymbol{US\_K}$;

USL_K_d – легальний користувач, за якого видає себе авторизована сторона, що автентифікується; здійснюється умова, що $((USL\_K_d \in \boldsymbol{US\_K}) \wedge (USL\_K_d \in \in \boldsymbol{USL\_K}))$;

$$\boldsymbol{USB\_K}=\{\bigcup_{tb=1}^{lb} USB\_K_{tb}\}=\{USL\_K_1, USL\_K_2, \ldots, USL\_K_{tb}, \ldots, USL\_K_{lb}\};\ \ tb=\overline{1,lb};\ lb$$

– кількість порушників даної системи; ***USB_K*** – множина порушників даної системи, тобто користувачів, які в ній не зареєстровані, але намагаються отримати

до неї доступ; здійснюється умова, що $((\boldsymbol{USB\_K} \subset \boldsymbol{US\_K}) \wedge (\boldsymbol{USB\_K} \not\subset \boldsymbol{USL\_K}))$;

$$\boldsymbol{AT\_K}=\{\bigcup_{if=1}^{ks\_kf+1} AT\_K_i\}=\{TM\_K_1,\ TM\_K_2,\ \ldots,\ TM\_K_{if},\ \ldots,\ TM\_K_{ks\_kf},\ OCH\_K\};$$

$if=\overline{1,ks\_kf}$; ks_kf – кількість символів в ключовій фразі (КФ); $\boldsymbol{AT\_K}$ – множина ознак КП користувача; TM_K_{if} – часовий інтервал між вводом (if-1)-ого та if-ого символів КФ;

$$\boldsymbol{NSL}=\{\bigcup_{in=1}^{kns} NSL_{in}\}=\{\boldsymbol{NSL}_1,\ \boldsymbol{NSL}_2,\ \ldots,\ \boldsymbol{NSL}_{in},\ \ldots,\ \boldsymbol{NSL}_{mao},\ \ldots,\ \boldsymbol{NSL}_{kns}\};\ \boldsymbol{NSL}_{in}=$$

$$=\{\bigcup_{ns=1}^{ksn} SL_{in,ns}\}=\{SL_{in,1},\ NSL_{in,2},\ \ldots,\ SL_{in,ns},\ \ldots,\ SL_{in,ksn}\};$$ відповідно множина $\boldsymbol{NSL}$ приймає наступний вид: $$\boldsymbol{NSL}=\{\bigcup_{in=1}^{kns}\bigcup_{ns=1}^{ksn} SL_{in,ns}\}=\{\{\ SL_{1,1},\ NSL_{1,2},\ \ldots,\ SL_{1,ns},\ \ldots,\ SL_{1,ksn}\ \},\ \{SL_{2,1},$$

$NSL_{2,2},\ \ldots,\ SL_{2,ns},\ \ldots,\ SL_{2,ksn}\},\ \ldots\ \{SL_{in,1},\ NSL_{in,2},\ \ldots,\ SL_{in,ns},\ \ldots,\ SL_{in,ksn}\},\ \ldots\ \{SL_{mao,1},$ $NSL_{mao,2},\ \ldots,\ SL_{mao,ns},\ \ldots,\ SL_{mao,ksn}\},\ \ldots\ \{SL_{kns,1},\ NSL_{kns,2},\ \ldots,\ SL_{kns,ns},\ \ldots,\ SL_{kns,ksn}\}\};$ $in=\overline{1,kns};$ kns – кількість наборів слів; $\boldsymbol{NSL}$ – множина наборів слів, НЗ динаміки вводу яких є в базі даних навчальних зразків (БДНЗ); $ns=\overline{1,ksn}$; ksn – кількість слів в in-ому наборі; $\boldsymbol{NSL}_{in}$ – множина слів в in-ому наборі; $\boldsymbol{NSL}_{mao}$ – набір слів, для якого амплітуда відсотка помилок у користувачів мінімальна;

$$\boldsymbol{O\_K}=\{\bigcup_{t=1}^{l}\bigcup_{nz=1}^{knz_t} \boldsymbol{O\_K}_{t,nz}\}=\{\{\boldsymbol{O\_K}_{1,1},\boldsymbol{O\_K}_{1,2},\ \ldots,\ \boldsymbol{O\_K}_{1,nz},\ \ldots,\ \boldsymbol{O\_K}_{1,knz1}\},\{\boldsymbol{O\_K}_{2,1},\boldsymbol{O\_K}_{2,2},$$

$\ldots,\ \boldsymbol{O\_K}_{2,nz},\ \ldots,\ \boldsymbol{O\_K}_{2,knz2}\},\ \ldots,\ \{\boldsymbol{O\_K}_{t,1},\boldsymbol{O\_K}_{t,2},\ \ldots,\ \boldsymbol{O\_K}_{t,nz},\ \ldots,\ \boldsymbol{O\_K}_{t,knzt}\},\ \ldots,\ \{\boldsymbol{O\_K}_{l,1},$ $\boldsymbol{O\_K}_{l,2},\ \ldots,\ \boldsymbol{O\_K}_{l,nz},\ \ldots,\ \boldsymbol{O\_K}_{l,knzl}\}\};$ $\boldsymbol{O\_K}_{t,nz}=\{\bigcup_{i=1}^{ks\_k} TM\_K_{t,nz,i}\}=\{TM\_K_{t,nz,1},\ TM\_K_{t,nz,2},\ \ldots,$

$TM\_K_{t,nz,i},\ \ldots,\ TM\_K_{t,nz,ks\_k}\}$; відповідно множина $\boldsymbol{O\_K}$ приймає наступний вид: $$\boldsymbol{O\_K}=\{\bigcup_{t=1}^{l}\bigcup_{nz=1}^{knz_t}\bigcup_{i=1}^{ks\_k} TM\_K_{t,nz,i}\}=\{\{TM\_K_{t,1,1},\ TM\_K_{t,1,2},\ \ldots,\ TM\_K_{t,1,i},\ \ldots,\ TM\_K_{t,1,ks\_k}\},$$

$\{TM\_K_{t,2,1},\ TM\_K_{t,2,2},\ \ldots,\ TM\_K_{t,2,i},\ \ldots,\ TM\_K_{t,2,ks\_k}\},\ \ldots,\ \{TM\_K_{t,nz,1},\ TM\_K_{t,nz,2},\ \ldots,$ $TM\_K_{t,nz,i},\ \ldots,\ TM\_K_{t,nz,ks\_k}\},\ \ldots,\ \{TM\_K_{t,knz,1},\ TM\_K_{t,knz,2},\ \ldots,\ TM\_K_{t,knz,i},\ \ldots,$ $TM\_K_{t,knz,ks\_k}\}\};$ $nz=\overline{1,knz_t}$; knz_t – кількість навчальних зразків КП t-ого користувача

в БДНЗ, які використовуються в даному сеансі розпізнавання; ***O_K*** – множина навчальних зразків КП з БДНЗ, які використовуються в даному сеансі розпізнавання; *ks_k* – кількість символів в КФ, динаміка вводу яких використовується при АК; $\boldsymbol{O\_K}_{t,nz}$ – *nz*-ий навчальний зразок КП *t*-ого користувача, який є множиною ознак КП користувача; здійснюється умова, що $\boldsymbol{O\_K} \subset \boldsymbol{At}$;

$\boldsymbol{ON\_K}=\{\bigcup_{i=1}^{ks\_k} TM\_K_i\} = \{TM\_K_1, TM\_K_2, \ldots, TM\_K_i, \ldots, TM\_K_{ks\_k}\}$; ***ON_K*** – зразок КП, якій подається на ІНМ для розпізнавання; здійснюється умова, що $\boldsymbol{ON\_K} \subset \boldsymbol{At}$;

$OCH\_K_{t,in}$ – відсоток помилок, які робить *t*-ий користувач, під час вводу КФ *in*-набору;

$A\_OCH\_K_{in}$ – амплітуда (розкид) відсотка помилок у користувачів між собою, під час вводу КФ *in*-набору;

$M_{t,i}$ – математичне очікування (середнє значення параметра) *i*-ого параметра у *t*-ого користувача;

$S_{t,i}$ – дисперсія (розкид параметра) *i*-ого параметра у *t*-ого користувача;

H_i – спеціальна функція від $M_{t,i}$ та $S_{t,i}$;

$Sr_{t,i}$ – середньоарифметичне значення *i*-ого параметра у *t*-ого користувача.

Відповідно до розробленої моделі даних, функція *R_K* має вигляд:

$$R\_K = \begin{cases} 1, \text{ при } (US\_K_X \in \boldsymbol{USL\_K}) \wedge (US\_K_X = USL\_K_d); \\ 0, \text{ при } ((US\_K_X \in \boldsymbol{USL\_K}) \wedge (US\_K_X \neq USL\_K_d)) \vee (US\_K_X \in \boldsymbol{USB\_K}). \end{cases}$$

2.2.Метод первинної обробки зразків клавіатурного почерку

В даному методі (метод ПОЗКП) виконується первинна обробка зразків КП користувачів, що накопичуються в БДНЗ. *Цей метод складається з двох етапів* [99-116]**.**

Етап 1. Вибір ключової фрази та характеристик, що будуть аналізуватися під час АК ІС. На цьому етапі виконується аналіз зразків з пробної БДНЗ для вибору набору КФ та символів КФ, динаміка вводу яких буде аналізуватися під час АК ІС. КФ, в даній роботі, – це слово або фраза, динаміка вводу символів з якої, буде аналізуватися під час АК ІС.

Даний етап складається з двох кроків.

Крок 1. Вибір КФ на основі аналізу накопичених зразків РП з пробної бази даних. На даному кроці, на основі аналізу НЗ введення різних наборів КФ, за критеріями мінімуму амплітуди розподілу відсотка помилок у користувачів, під час вводу КФ, обирається робочий набор КФ. Робочий набір КФ – це набір КФ, динаміка вводу яких буде аналізуватися під час АК в даній ІС. На основі аналізу результатів експериментів, що проведені в даній роботі, можна сказати, що від відсотка помилок, що роблять користувачі, навіть якщо НЗ, у яких зроблені ці помилки, не зберігаються в БДНЗ, досить сильно залежить ІПР системою даного користувача. Помилкою клавіатурного вводу (ПКВ), в даній роботі, вважається введення неправильного символу, або використання, для введення, часу більшого, ніж вказано в налаштуваннях системи розпізнавання [99-116]. Відсоток помилок, які робить користувач обчислюється за наступною формулою:

$$OCH\_K_{t,in} = \frac{100 \cdot \sum_{nz=1}^{knz} OCH\_K_{t,in,nz}}{ks\_k_{in} \cdot knz_{t,in} + \sum_{nz=1}^{knz} OCH\_K_{t,in,nz}};$$

Крім того, на ІПР системою всіх користувачів досить сильно впливає амплітуда (розподіл) відсотка помилок у користувачів між собою. Цей параметр обчислюється за наступною формулою:

$$A\_OCH\_K_{in} = OCH\_K\_MAX_{in} - OCH\_K\_MIN_{in};$$

де $OCH\_K\_MAX_{in}$ - максимальний відсоток помилок, що виникають при вводі слів *in*-го набору, серед усіх користувачів; $OCH\_K\_MIN_{in}$ - мінімальний відсоток помилок, що виникають при вводі слів *in*-го набору, серед усіх користувачів.

В даній роботі робочим обирається набір $\boldsymbol{NSL}_{mao}=\{\bigcup_{ns=1}^{ksn} SL_{mao,ns}\}$, для якого A_OCH_K всіх користувачів ІС мінімальне.

Крок 2. Вибір характеристик, що будуть аналізуватися при АК ІС. На даному кроці, за допомогою пошуку мінімуму значення спеціальної функції від математичного очікування та дисперсії, в КФ робочого набору обираються символи,

динаміка вводу яких є характерною для групи користувачів. Для визначення характеристик, найбільш придатних для розпізнавання, в даній роботі, спочатку розраховуються математичне очікування (середнє значення параметра) $M_{t,i}$ та дисперсія (розподіл параметра) $S_{t,i}$ для значень $TM\_K_{t,nz,i}$ для кожного символу КФ [99-116]. $M_{t,i}$ та $S_{t,i}$ обчислюються за наступними формулами:

$$M_{t,i} = \frac{\sum_{nz=1}^{knz_t} TM\_K_{t,nz,i}}{knz_t}; \quad S_{t,i} = \sqrt{\sum_{nz=1}^{knz_t} (TM\_K_{t,nz,i} - M_{t,i})^2 / (knz_t - 1)}.$$

На основі значень $M_{t,i}$ та $S_{t,i}$, для вибору кращих для розпізнавання характеристик, розраховується спеціальна функція від цих параметрів, використовуючи наступні формули [99-116]:

$$H_i = \frac{\sum_{r=1}^{l-1} \sum_{c=r+1}^{l} \frac{(S_{r,i} + S_{c,i})}{|M_{r,i} - M_{c,i}|}}{komb}; \quad komb = \frac{l!}{2!*(l-2)!} = \frac{l!}{2*(l-2)!};$$

де *komb* - кількість можливих комбінацій з *l* по дві (кількість можливих пар для порівняння); знак «!» означає факторіал числа.

Головною вимогою для характеристик, що аналізуються, є: H_i<1, але чим H_i менше, тим краще буде розпізнавання за цією ознакою [99-116].

В даній роботі, на основі параметру H_i, обирається динаміку вводу *ks_k* яких саме символів з КФ з робочого набору $\boldsymbol{NSL}_{mao}$, аналізувати під час АК даної ІС, тобто обираються елементи множини ознак КП ***AT_K***, в кількості *ks_k*, для яких мінімальні значення параметру H_i,та виконується умова, що для всіх обраних ознак H_i<1.

Етап 2. Попередній відбір НЗ, які будуть використовуватись для розпізнавання. На цьому етапі виконується видалення з БДНЗ, зразків з грубими помилками за допомогою порівняння ознаки $TM\_K_{t,nz,i}$ з її середньоарифметичним значенням $Sr_{t,i}$ для *t*-ого користувача. $Sr_{t,i}$ обчислюється за наступною формулою:

$$Sr_{t,i} = \frac{\sum_{nz=1}^{knz_t} TM\_K_{t,nz,i}}{knz_t}.$$

Якщо для всіх характеристик $TM\_K_{t,nz,i}$ навчального зразка $\boldsymbol{O\_K}_{t,nz}$ виконується

умова $\left|TM\_K_{t,nz,i} - Sr_{t,i}\right| \le k * Sr_{t,i}$, тоді цей НЗ $\boldsymbol{O\_K}_{t,nz}$ повинен залишитися в БДНЗ, в іншому випадку цей НЗ необхідно видалити з БДНЗ. Коефіцієнт *k* обирається в залежності від необхідної точності розпізнавання, чим менший цей коефіцієнт, тим більш точні будуть накопичені НЗ, але тим менше їх залишиться після відбору [99-116].

2.3.Метод автентифікації користувачів інформаційних систем за їх клавіатурним почерком

У розробленому методі (метод АККП) для АК ІС аналізується множина ознак його КП ***AT_K***, яка описана в моделі даних. Ця множина складається зі значень часових інтервалів між натисканням двох сусідніх символів КФ (TM_K_i). При цьому задачу автентифікації можна звести до задачі класифікації або розпізнавання образів.

Є чотири основні різновиди технологій розпізнавання за КП [99-116]:

1. За динамікою набору якоїсь постійною ключовою фразою.

2. За динамікою набору вільного тексту – фрази, яка постійно змінюється.

3. За динамікою набору одного випадково обраного слова з заздалегідь підібраного набору слів, при цьому можуть аналізуватися всі отримані характеристики, але часові інтервали між натисканням двох сусідніх символів порівнюються з відповідними характеристиками тільки такого ж слова.

4. За динамікою набору одного випадково обраного слова з заздалегідь підібраного набору слів, але у всіх цих словах повинен бути однаковий фрагмент і аналізуються часові інтервали між натисканням сусідніх символів саме з цього фрагмента, хоча порівнюватися вони можуть з відповідними ознаками цього фрагмента з будь-якого слова даного набору.

Технології першого різновиду забезпечують не досить високий рівень безпеки ІС. Для реалізації технологій другого різновиду необхідно задіяти досить значний об’єм ресурсів. Тому в даній роботі використовувалися третій та четвертий різновид технологій розпізнавання.

Розроблений метод АККП складається з дев'ятьох етапів.

Етап 1. Попереднє формування множини ознак КП користувача ІС.

На даному етапі роботи формується множина ознак КП користувача

$\boldsymbol{AT\_K}=\{\bigcup_{i=1}^{ks\_k+1} AT\_K_i\}$, яка потім використовується під час розпізнавання та для аналізу при виборі набору слів, для подальшого розпізнавання [99-116]. Ознакою в даній роботі є часовий інтервал між натисканням клавіш двох сусідніх символів КФ TM_K_i.

Етап 2. Налаштування параметрів, які є найбільш критичними при АК ІС за їх КП. Ефективність роботи системи залежить від правильних налаштувань. В даній роботі налаштовуються такі параметри: необхідна кількість НЗ; кількість слів в наборі; кількість символів, динаміка введення яких аналізується; значення коефіцієнту (*k*), який використовується на Етапі 2 методу ПОЗКП;

Етап 3. Накопичення пробної БДНЗ. При виборі КФ слід враховувати сферу діяльності, в якій застосовується дана система АК, тобто обирати слова, які часто використовуються в роботі користувачів ІС і стиль введення їх буде найбільш характерним. Особливо важливо використовувати в якості КФ інформацію, введення якої працівники часто повторюють, у випадку, якщо автентифікація виконується для проведення прихованого моніторингу. Для вибору найбільш оптимальної КФ в роботі була накопичена пробна БДНЗ (невеликого об'єму) по динаміці вводу *kns* наборів з *ksn* слів, з яких кожен раз, під час АК, випадковим чином буде обиратися одне слово для вводу; при цьому, в кожному слові є послідовність символів, яка повторюється у всіх словах відповідного набору [99-116].

Етап 4. Вибір КФ на основі аналізу накопичених зразків РП з пробної бази даних. Доцільність цього етапу пояснюється наступними факторами. Правильність розпізнавання досить сильно залежить від правильності вибору КФ. При виборі КФ слід враховувати сферу діяльності, в якій застосовується дана система АК, тобто обирати слова, які часто використовуються в роботі користувачів ІС і стиль введення їх буде найбільш характерним. Особливо важливо використовувати в якості КФ інформацію, введення якої працівники часто повторюють, у випадку, якщо автентифікація виконується для проведення прихованого моніторингу. Для виконання цієї задачі виконуються Крок 1, Етапу 1, методу ПОЗКП [99-116].

Етап 5. Накопичення БДНЗ. Після вибору робочого набору КФ $\boldsymbol{NSL}_{mao}$, виконується накопичення БДНЗ об'єму, необхідного для роботи системи, тобто

виконується реєстрація користувачів ІС. Якщо користувач намагається пройти АК, а для в БДНЗ недостатньо НЗ, тоді його не допускають до етапу автентифікації, а направляють на даний етап накопичення БДНЗ для реєстрації його в ІС [99-116]. Під час формування БДНЗ, користувачам необхідно набрати деяку кількість разів (декілька сотень разів) набір з 10 слів. У всіх цих словах останні 6 літер однакові. При наборі заміряється час необхідний для набору кожного символу, тобто формується НЗ $\boldsymbol{O\_K}_{nz}$ з множини ознак $\boldsymbol{AT\_K}=\{\bigcup_{i=1}^{ks\_k+1} AT\_K_i\}$.

Етап 6. Вибір характеристик, що будуть аналізуватися при АК ІС. Правильність розпізнавання досить сильно залежить від правильності вибору параметрів, які аналізуються, тобто від того часові проміжки $TM\_K_{t,nz,i}$ між вводом яких саме символів аналізуються. Для виконання цієї задачі виконуються Крок 2, Етапу 1, методу ПОЗКП.

Етап 7. Попередній відбір НЗ, які будуть використовуватись для розпізнавання. Для підвищення якості розпізнавання, необхідно виконати відбір навчальних даних, тобто видалити з БДНЗ зразки з грубим відхиленням хоча б однієї з ознак. В даній роботі під грубим відхиленням малося на увазі відхилення значення ознаки зразка $TM\_K_{t,nz,i}$ від середньоарифметичного значення $Sr_{t,i}$ значень цієї ж ознаки у всіх інших навчальних зразках почерку цього користувача, на відсоток більший, ніж вказано в налаштуваннях системи розпізнавання. Для виконання цієї задачі виконуються Етап 2, методу ПОЗКП [99-116].

Етап 8. Виконання, за необхідністю, відбору за словами навчальних зразків з БДНЗ. На цьому етапі визначається який саме різновид технології розпізнавання за КП використовується. Тобто, якщо аналізується часові інтервали між введенням сусідніх символів не однакового фрагмента всіх слів набору, а часові інтервали між введенням сусідніх символів всього слова, тоді необхідно з множини навчальних зразків КП $\boldsymbol{O\_K}$, вибрати тільки зразки вводу того слова, яке вводить в даному сеансі автентифікації, користувач, що розпізнається US_K_X [99-116].

Етап 9. Виконання автентифікації користувачів ІС за їх КП. Задачу АК, тобто розпізнавання образів, в даному випадку, можна звести до задачі класифікації

образів. В якості механізму класифікації образів, в даному випадку, була обрана ІНМ. Для розпізнавання на ІНМ подається зразок ***ON_K***. Обрана мережа складається з наступних чотирьох шарів: вхідний шар, шар зразків, шар підсумовування, вихідний шар. Виходячи з поставленої задачі, кількісно архітектура мережі була визначена наступним чином [99-116,126]:

1. Число вхідних елементів дорівнює числу ознак. В якості ознаки використовуємо час набору відповідного символу слова TM_K_i. Таким чином, якщо ми аналізуємо час набору шести символів, то вхідний шар буде складатися з шести елементів.

2. Число елементів шару зразків дорівнює числу навчальних зразків. У нашому випадку, навчальним зразком будуть дані про час набору одним користувачем символів одного слова за одну спробу. Отже, кількість елементів у шарі зразків дорівнює кількості користувачів, для яких будується мережа, помноженій на кількість слів введених кожним користувачем.

3. Число елементів шару додавання дорівнює числу класів. При рішенні задачі автентифікації необхідно визначити конкретного користувача системи. Тому класом у нас є користувач системи. Отже, кількість користувачів у системі визначає кількість класів *l*.

4. Вихідний шар має завжди один елемент.

Активність елементу шару зразків обраховується за наступною формулою:

$$AK\_K_{nz} = \exp\left(\frac{\sum_{i=1}^{ks\_k} ON\_K_i \cdot O\_K_{i,nz} - 1}{\sigma^2}\right),$$

де σ – ширина функції активності.

Для використання цієї формули, вектор вхідних даних повинен бути нормалізовано.

Вихідний елемент являє собою дискримінатор граничної величини, який вказує елемент шару додавання з максимальним значенням, тобто вказує до якого класу належить невідомий екземпляр, або іншими словами, визначає користувача, який намагається отримати доступ до ІС. Якщо це ім’я дорівнює імені, що пред’явлено

під час АК, тоді система дозволяє доступ до ІС [99-116].

Для збільшення ІПР, в множині ***AT_K*** можна фіксувати, для подальшого використанні при АК, не тільки часові інтервали між натисканням двох сусідніх символів КФ, а й часові інтервали між натисканням та відпусканням відповідних символів. Крім того, враховуючи той факт, що у всіх людей різний ступінь уваги, тому кількість зроблених помилок, частота їх повторювання і слова, при вводі яких виник збій – це досить індивідуальні характеристики, які теж можна використовувати для АК для збільшення ІПР.

2.4.Алгоритмічна реалізація методу первинної обробки зразків клавіатурного почерку та методу автентифікації користувачів інформаційних систем за їх клавіатурним почерком

Розглянемо докладніше розроблені методи, доцільність виконання деяких його етепів та алгоритми, які були розроблені на їх основі розроблених методів [99-116].

Для зручності написання вподальшому програмного забезпечення, на основі розроблених алгоритмів, позначення деяких характеристик в описі алгоритму і в описі методу відрізняються.

Можна виділити як мінімум чотири різновиди автентифікації за КП:

1. За динамікою набору якоїсь постійної ключової фрази. Для зручності подальшого викладу позначимо цей вид автентифікації, як автентифікація за динамікою набору постійної ключової фрази (автентифікація за ДНПКФ).

2. За динамікою набору вільного тексту – фрази, яка постійно змінюється. Таку автентифікацію позначимо, як автентифікація за динамікою набору вільного тексту (автентифікація за ДНВТ).

3. За динамікою набору одного випадково обраного слова з заздалегідь підібраного набору слів, при цьому можуть аналізуватися всі отримані характеристики, але часові інтервали між натисканням двох сусідніх символів порівнюються з відповідними характеристиками тільки такого ж слова. Таку автентифікацію можна позначити, як автентифікація за динамікою набору слова з набору (автентифікація за ДНСЗН).

4. За динамікою набору одного випадково обраного слова з заздалегідь підібраного набору слів, але у всіх цих словах повинен бути однаковий фрагмент і

аналізуються часові інтервали між натисканням сусідніх символів саме з цього фрагмента, хоча порівнюватися вони можуть з відповідними ознаками цього фрагмента з будь-якого слова даного набору. Цей вид автентифікації позначимо, як автентифікація за динамікою набору повторюваного фрагмента слова з набору (автентифікація за ДНПФСЗН).

Також існують і інші різновиди автентифікації за КП. Може використовуватися характеристика «похідна» від КП. Наприклад, розпізнавання може виконуватися за рисунком почерку, тобто для розпізнавання може використовуватися послідовність значень, що представляють собою різницю між сусідніми часовими інтервалами. Ця досить індивідуальна характеристика показує відносні уповільнення чи прискорення під час набору ключової фрази або окремих слів.

Треба визнати, що автентифікація за ДНВТ істотно відрізняється від інших різновидів автентифікації за КП. У даному випадку в базі даних навчальних зразків повинна зберігатися інформація не про набір конкретних слів чи фраз, а інформація про набір всіх можливих комбінацій символів у відповідній мові (а може й у декількох мовах), тобто в кожному зразку для кожного користувача повинні зберігатися часові інтервали між натисканням усіх можливих комбінацій символів. Відповідно, при виконанні автентифікації в кожному зразку треба знайти часовий інтервал для введеної з клавіатури пари символів (для яких визначився часовий інтервал) і це число і буде згодом використовуватися як ознака навчального зразка, яка аналізується. З огляду на значну відмінність цього способу розпізнавання, далі в роботі автентифікація за ДНВТ розглядатися не буде.

Як вже було сказано, у даній роботі для розпізнавання використовувався набір з десяти слів, у яких останні шість символів повторювалися. Всі слова в заздалегідь обраному наборі закінчувалися на «ізація» (наприклад, «телефонізація»). Такий набір можна використовувати в однаковій мірі для кожного з останніх двох різновидів автентифікації (автентифікація за ДНСЗН та за ДНПФСЗН). Крім того, будь-яке слово з такого набору можна використовувати і для автентифікації за ДНПКФ, як один з варіантів (постійний) ключової фрази. А для автентифікації за ДНВТ можна використовувати будь-який текст. Тому можна сказати, що обраний набір можна вико-

ристовувати для кожного з описаних у даній роботі різновидів автентифікації.

Будь-яка система захисту, яка виконує автентифікацію користувачів, використовуючи для цього їх біометричні параметри, повинна містити в собі базу даних навчальних зразків аналізуємих параметрів всіх користувачів, яким дозволений доступ до комп'ютерної системи, що захищається. В залежності від методу біометричної автентифікації, який використовується (від біометричної характеристики людини, яка використовується для аналізу) мінімально допустимий розмір бази даних навчальних зразків буде різним. Наприклад, якщо метод для розпізнавання використовує відбиток пальця, тоді в базі даних для кожного користувача має бути по декілька зразків (для декількох пальців і з урахуванням того, що палець на сканер може бути поміщений під різним кутом). Якщо ж метод для розпізнавання використовує КП, як у даній роботі, тоді розмір бази буде в багато разів більший. Це пояснюється тим, що відбиток пальця є статичним параметром і протягом життя людини не міняється, а КП – це динамічний параметр і в залежності від різних факторів (від досвіду роботи за комп'ютером, настрою, фізичного стану і т.д.), хоч і незначно, але може змінюватися. Крім того, для того, щоб реалізувати автентифікацію за КП за допомогою ІНМ база навчальних зразків повинна бути досить великою. Зазвичай вона досягає декілька сотень, а іноді, і тисяч зразків для кожного користувача. При налаштовуванні конкретної системи автентифікації кількість навчальних зразків для кожного користувача, які аналізуються, має вказати адміністратор чи інший працівник, який відповідає за безпеку даної комп'ютерної системи. Але як визначити необхідну кількість навчальних зразків для кожного користувача (z_t, де t – номер користувача), яка забезпечить необхідний рівень безпеки даної комп'ютерної системи? Точну відповідь дати неможливо. Значення цього параметра (z_t) у кожному конкретному випадку буде відрізнятися. Його потрібно підбирати. Він буде залежати від користувачів, які розпізнаються та відібраних для розпізнавання параметрів. У даній роботі було зроблене припущення, що чим більше в базі навчальних зразків для кожного користувача, тим більш ефективно буде працювати система розпізнавання, тобто ІПР буде вище. Для перевірки істинності цього припущення було проведено ряд експериментів, результати, яких наведені в розділі 4 даної роботи. Але при виборі кількості навча-

льних зразків для кожного користувача необхідно враховувати, що при значному збільшенні кількості навчальних зразків виникають деякі проблеми. При збільшенні цього параметра (z_t) системи автентифікації не тільки збільшується якість розпізнавання, але збільшується обсяг інформації, яка обробляється та зберігається а, отже, збільшуються часові витрати для виконання процесу автентифікації та збільшується обсяг необхідної пам'яті (постійної та оперативної). Тому при виборі кількості навчальних зразків, які використовуються для розпізнавання користувачів, треба визначити те оптимальне значення цього параметра, при якому буде забезпечуватися необхідний рівень безпеки і при цьому рівень витрачених ресурсів не буде недопустимо великим.

Можна сказати, що етап накопичення бази даних навчальних зразків параметрів, які аналізуються, для всіх користувачів, що мають право на доступ до комп'ютерної системи, що захищається, досить відповідальний та тривалий. Цей етап повинен здійснюватися під контролем адміністратора, тому що накопичення викривлених (навмисне чи ненавмисно) навчальних даних приведе до неправильного розпізнавання, що зведе до нуля ефективність застосування даної системи захисту. Накопичення навчальних даних полягає в тому, що кожен користувач набирає на клавіатурі деяку кількість разів запропоновану йому ключову фразу чи набір слів у випадковій послідовності (порядку), як у даній роботі. При цьому програма, що виконує збір і наступну обробку навчальних даних, вимірює і зберігає в спеціально створену базу даних часові інтервали між набором кожної пари сусідніх символів ключової фрази. Тобто, при кожному новому введенні ключової фрази в базі даних зберігається новий зразок аналізованих параметрів користувача, що у даний момент уводить цю ключову фразу. Довжина зразка дорівнює кількості символів у ключовій фразі, тобто скільки символів у ключовій фразі стільки параметрів, що аналізуються (для першого символу вимірюється час між початком набору та введенням першого символу, тому цей параметр може бути викривлений). Варто відзначити, що навіть якщо до моменту початку накопичення бази даних навчальних зразків вже відомо, що в процесі розпізнавання будуть використовуватися часові інтервали між натисканням тільки деяких символів ключової фрази (не усіх), проте, бажано вимірювати та потім зберігати в базі даних

навчальних зразків часові інтервали між натисканням усіх пар сусідніх символів ключової фрази, тому що в майбутньому з різних причинам можуть знадобитися ці дані. Наприклад, після аналізу великої кількості накопичених зразків з'ясується, що при попередньому аналізі цих біометричних характеристик були визначені не найоптимальніші для розпізнавання ознаки (часові інтервали між натисканням не тих символів).

Також, бажано щоб накопичення навчальних даних виконувалося не за один день, а протягом хоча б тижня, тобто щоб кожен користувач щодня протягом тижня періодично вводив запропоновану ключову фразу. Зважаючи на те, що з досвідом роботи на клавіатурі динаміка КП може мінятися (хоч і незначно), тому для більшої достовірності навчальних даних, і, отже, для підвищення ефективності використання даної системи автентифікації, накопичення навчальних даних бажано продовжити і після закінчення етапу накопичення. Тобто, згодом аналізовані параметри, які отримані при введенні ключової фрази, для проходження автентифікації, мають використовуватися не тільки для розпізнавання користувача в даний момент часу, але й зберігатися в базу даних в якості навчальних даних (якщо автентифікація дала позитивний результат). Або базу навчальних даних можна періодично поповнювати (обновляти), наприклад два рази на рік.

Як вже було сказано, при наборі ключової фрази чи слова під час накопичення бази даних навчальних зразків, з якихось причин, користувач може натиснути випадково не ту клавішу, тобто зробити помилку чи просто незвично довго замислитися, все це приводить до появи викривлених даних у базі даних навчальних зразків, яка накопичується, а, отже, до зменшення ІПР користувачів. Тому якщо при введенні КФ чи слова під час накопичення навчальних даних була зроблена помилка між введенням двох сусідніх символів слова чи відбулася недопустимо велика пауза, тоді дану спробу введення краще припинити, цей зразок не зберігати в базі даних навчальних зразків, а користувачу, який тестується, запропонувати ввести це слово заново. Крім того, у всіх людей різний рівень уважності, тому кількість зроблених помилок, частота їхнього повторення та слова, при введенні яких відбувся збій – це досить індивідуальні характеристики людини, і це згодом можна використовувати

для розпізнавання користувачів, тому ці дані також бажано зберігати в базі даних навчальних зразків. Проаналізувавши результати проведених у даній роботі експериментів, що проілюстровані в підрозділі 4.3, можна сказати, що від відсотка помилок, що робляться користувачем, (*Och*), навіть якщо навчальні зразки, у яких зроблені ці помилки, не зберігаються в базі, досить сильно залежить ІПР системою даного користувача.

Відсоток помилок, які робить користувач обчислюється за наступною формулою:

$$Och = \frac{100 * k_o}{kol + k_o};$$

де k_o - кількість натискання помилкових клавіш;

kol - кількість натискання вірних клавіш.

Крім того, на ІПР системою всіх користувачів досить сильно впливає амплітуда (розкид) відсотка помилок у користувачів між собою. Цей параметр обчислюється за наступною формулою:

$$A_{Och} = Och_{\max} - Och_{\min};$$

де $Och_{\max}$ - максимальний відсоток помилок серед усіх користувачів; $Och_{\min}$ - мінімальний відсоток помилок серед усіх користувачів.

Ці факти також треба враховувати при створенні та налаштовуванні системи автентифікації у кожному конкретному випадку.

Якщо при майбутньому аналізі планується усереднювати накопичені навчальні дані (алгоритм і доцільність виконання цих дій будуть описані пізніше), то при їхньому накопиченні бажано вводити ключову фразу (у випадку автентифікації за ДНПКФ) або кожне слово з набору (у випадках автентифікації за ДНСЗН та за ДНПФСЗН) декілька разів підряд. Кількість повторень (K_{pov}) повинна встановлюватися до початку процесу накопичення навчальних даних (хоча згодом може коректуватися) адміністратором або іншою людиною, що відповідає за безпеку даної комп'ютерної системи. У даній роботі було обрано K_{pov}=10.

Підсумувавши сказане, можна скласти алгоритм накопичення навчальних зразків з запам'ятовуванням зроблених помилок (алгоритм ННЗЗЗЗП). Але при

цьому треба враховувати, що в залежності від того який різновид автентифікації за КП виконується (виконується автентифікація за ДНПКФ, за ДНВТ, за ДНСЗН, за ДНПФСЗН), будуть відрізнитися й алгоритми ННЗЗЗЗП.

У випадках, коли виконується автентифікація за ДНСЗН або за ДНПФСЗН алгоритм ННЗЗЗЗП (для накопичення даних про введення заданої кількості разів (K_{pov}) підряд кожного слова з одного набору слів одним користувачем) буде мати наступний вигляд:

1. Прочитати з відповідної таблиці параметрів налаштовувань бази даних і записати у відповідні змінні та масиви наступну інформацію: номер активного (обраного для накопичення по ньому навчальних зразків) набору слів – у змінну N_{sl}, кількість слів в активному наборі – у змінну s, кількість повторень кожного слова активного набору при введенні (тестуванні) – у змінну K_{pov}, усі слова активного набору – у масив з ім'ям *Slovo*.

2. Попросити користувача, який тестується, ввести своє прізвище, ім’я, по батькові та поточну дату.

3. Після введення цієї інформації користувачу необхідно натиснути кнопку, що вказує на початок введення слів з набору. Якщо відповідна кнопка натиснута, тоді перейти до пункту 4.

4. Якщо введена вся необхідна інформація (зазначена в пункті 2), тоді створити новий навчальний зразок і зберегти цю інформацію та інформацію, отриману в пункті 1 у даному зразку, крім того, визначити поточний (системний) час та, також, зберегти його в даному зразку (ця інформація згодом пригодиться для аналізу). Після цього перейти до пункту 5. Інакше, якщо була введена не вся необхідна інформація, тоді видати на екран повідомлення про це та перейти до пункту 3.

5. Обнулити змінну p_s: p_s=0, де p_s - кількість слів з масиву з ім'ям *Slovo*, для яких введені зразки в даному тесті.

6. За допомогою датчика випадкових чисел визначити номер (k_s) слова для введення.

7. Записати 1 у змінну n_{pov}: n_{pov}=1, де n_{pov} - номер повторення.

8. Записати в масив з ім'ям *Slovo_n* k_s-е слово з масиву з ім'ям *Slovo*.

9. Обнулити змінну *Och*: *Och*=0, де *Och* - кількість зроблених у цій спробі помилок.

10. Обнулити змінну *T*: *T*=0, де *T* - час, витрачений для набору даного слова в даній спробі.

11. Вивести на екран обране k_s-е слово з масиву з ім'ям *Slovo*, номер повторення (n_{pov}) та необхідну кількість повторень (K_{pov}) із пропозицією користувачу ввести це слово.

12. Визначити та записати у відповідний зразок бази даних навчальних зразків для кожної пари символів час між їх натисканням (*i* змінювати від 1 до *n*). Для цього:

12.1. Спочатку, після виводу слова на екран визначити і записати в змінну *T2* поточний час.

12.2. Активізувати таймер.

12.3. Якщо спрацював таймер, тобто пройшло максимально допустимий для введення одного символу проміжок часу, тоді видати повідомлення про це та повторити дану спробу введення даного слова заново, для цього перейти до пункту 10.

12.4. Коли натиснута якась клавіша, то перевірити, чи правильний символ введений. Якщо введений помилковий символ, тоді збільшити *Och* на 1: *Och*=*Och*+1 та перейти до пункту 10. Якщо введений правильний символ, тоді виміряти поточний час та записати його в змінну *T1*; знайти *T3*=*T1*-*T2* (тобто визначити шуканий часовий інтервал); записати *T3* у відповідний зразок бази даних навчальних зразків; обчислити *T*=*T*+*T3*; переписати значення з змінної *T1* у змінну *T2*: *T2*=*T1*. Якщо символ, який вводився, не останній, тоді очікувати введення наступного символу, для цього перейти до підпункту 12.2, якщо символ, який вводився останній, тоді значення змінних *T* та *Och* записати у відповідний зразок бази даних навчальних зразків та перейти до пункту 13.

13. Збільшити n_{pov} на 1: $n_{pov}=n_{pov}+1$.

14. Якщо $n_{pov} > K_{pov}$, тоді перейти до пункту 15, інакше перейти до пункту 9.

15. Збільшити p_s на 1: $p_s=p_s+1$.

16. Якщо p_s=s, тоді перейти до пункту 17, інакше вибрати в масиві з ім'ям *Slovo* нове слово для тесту, для цього:

16.1. За допомогою датчика випадкових чисел визначити номер (k_s) слова для введення.

16.2. Перевірити, чи немає в масиві з ім'ям *Slovo_n* k_s-го слова з масиву з ім'ям *Slovo*. Якщо таке слово там є, тоді для забезпечення рівномірного накопичення навчальних зразків для всіх слів набору, необхідно вибрати нове слово, для цього перейти до підпункту 16.1. Якщо такого слова немає в масиві з ім'ям *Slovo_n*, тоді перейти до пункту 7.

17. Видати на екран повідомлення про закінчення проходження тесту.

Як вже було сказано, це алгоритм для накопичення навчальних зразків по введенню одним користувачем одного набору слів, при цьому кожне слово повторюється підряд K_{pov} раз. Для накопичення більшої кількості навчальних зразків для кожного слова з набору необхідно, відповідно, більшу кількість разів повторити введення заданого набору слів (та збереження характеристик введення), тобто відповідну кількість разів виконати даний алгоритм. Ці дії необхідно виконувати для всіх користувачів, що працюють у даний час, чи передбачених згодом.

У випадку, коли виконується автентифікація за ДНПКФ алгоритм ННЗЗЗЗП (для накопичення даних по введенню заданої кількості разів (K_{pov}) ключової фрази одним користувачем) зміниться: по-перше, у пункті 1 не буде вводитися кількість слів у наборі та замість введення слів у масив з ім'ям *Slovo*, буде вводитися ключова фраза в змінну *Slovo*, при цьому скрізь в алгоритмі всі дії будуть виконуватися не зі словами, а з однією фразою; по-друге, в алгоритмі зникнуть пункти 5, 6, 8, 15, 16. Інших істотних відмінностей не буде.

Правильність розпізнавання досить сильно залежить від правильності вибору параметрів, які аналізуються, тобто від правильності вибору ключової фрази та від того, час між натисканням яких символів аналізувати. Тому що, зазвичай аналізуються не всі отримані параметри, а тільки деякі з них, інші ж використовуються для зручності сприйняття користувачем запропонованої фрази. Ключова фраза та параметри, які в ній аналізуються, підбираються в залежності від багатьох параметрів в

процесі експериментів. Наприклад, важливим може виявитися те, в якій галузі працює організація, в якій використовується ця система захисту. Також в якості ключової фрази може використовуватися назва організації, назва відділу, чи яка-небудь інша ключова для даного колективу інформація. Особливо важливо використовувати в якості ключової фрази інформацію, введення якої працівники часто повторюють у випадку, якщо автентифікація виконується для проведення прихованого моніторингу, тобто якщо користувачі не повинні знати, коли їх контролюють. В інших випадках в якості ключової фрази може використовуватися будь-який текст, динаміка набору якого індивідуальна для всіх користувачів даної комп'ютерної системи. Підсумувавши сказане, можна зробити висновок, що правильний вибір ключової фрази має досить великий вплив на якість виконання автентифікації.

Тому до накопичення досить великої бази даних навчальних зразків необхідно для декількох варіантів ключової фрази накопичити невеликий обсяг навчальних зразків («пробні» бази даних навчальних зразків для декількох варіантів ключових фраз) і проаналізувати отримані дані. Потім вибрати з варіантів ключової фрази, що вводяться, ту, динаміка набору якої найбільш точно класифікує користувачів даної комп'ютерної системи. І якщо така точність класифікації забезпечує необхідний рівень безпеки, тоді обрати цей варіант ключової фрази, як постійний та по цій фразі накопичувати необхідну базу даних навчальних зразків і потім цю фразу використовувати в якості ключової для автентифікації. Якщо ж не одна з запропонованих фраз не забезпечує необхідну точність розпізнавання, тоді необхідно вибрати інші варіанти ключових фраз, накопичити необхідний невеликий обсяг даних по динаміці їх набору та зробити їх аналіз. І так доти, доки не буде знайдена фраза, динаміка набору якої може забезпечити необхідний рівень безпеки даної комп'ютерної системи.

При цьому виникає питання про те, який критерій вибору характеристик придатних для класифікації користувачів. Для вибору придатних для класифікації характеристик можна скористатися такими математичними поняттями, як математичне очікування (середнє значення параметра) і дисперсія (розкид параметра).

Математичне очікування *i*-ої аналізованої характеристики конкретного користувача можна обчислити за наступною формулою:

$$M_{it} = \frac{\sum_{j=1}^{z_t} u_{ijt}}{z_t};$$

де u_{ijt} - часовий інтервал, необхідний для натискання *i*-го аналізованого символу ключової фрази з *j*-го зразка *t*-го користувача.

При цьому u_{ijt} є елементом масиву навчальних зразків *U*:

$$U = \{u_{111}, u_{112}, u_{113}, ..., u_{ijt}, ..., u_{nz_l l}\};\ 1 \le i \le n;\ 1 \le j \le z_t;\ 1 \le t \le l;$$

де n - кількість ознак, що аналізуються; z_t - кількість навчальних зразків для *t*-го користувача; l - кількість користувачів, про яких є інформація в базі даних навчальних зразків.

Для підрахунку дисперсії *i*-ої аналізованої характеристики конкретного користувача можна скористатися наступною формулою:

$$S_{it} = \sqrt{\sum_{j=1}^{z_t} (u_{ijt} - M_{it})^2 / (z_t - 1)}.$$

Для вибору кращих для розпізнавання характеристик можна використовувати декілька критеріїв. Один з них – це значення дисперсії – чим дисперсія менша, тим характеристика краще, тобто тим менший розкид значень у цієї характеристики. Іншим критерієм може служити співвідношення двох розглянутих характеристик. Це співвідношення може розраховуватися за формулою:

$$H_i = \frac{\sum_{r=1}^{l-1} \sum_{c=r+1}^{l} \frac{(S_{ir} + S_{ic})}{|M_{ir} - M_{ic}|}}{komb};$$

де *komb* - кількість можливих комбінацій з *l* по дві (кількість можливих пар для порівняння), що обчислюється за наступною формулою:

$$komb = \frac{l!}{2!*(l-2)!} = \frac{l!}{2*(l-2)!};$$

де знак «!» означає факторіал числа.

Головною вимогою для характеристик, що аналізуються, є: $H_i<1$, але чим H_i менше, тим краще буде розпізнавання за цією ознакою.

Виходячи зі сказаного, можна скласти алгоритм визначення якості аналізованих параметрів (алгоритм ВЯАП), або іншими словами алгоритм визначення придатності конкретних характеристик, для використання їх в якості параметрів, які аналізуються для проведення автентифікації. Але, як вже було сказано, в якості ключової фрази можна використовувати не тільки постійно ту саму ключову фразу (автентифікація за ДНПКФ), а якесь одне слово з заздалегідь сформованого набору слів (варіант автентифікації за ДНВТ тут не будемо розглядати). Яке саме слово використовувати в даному сеансі автентифікації визначається випадковим чином. У цьому випадку є два варіанти вибору ознак, що аналізуються. Або в якості характеристик, що аналізуються, обираються часові інтервали між натисканням символів спільного для всіх слів фрагмента (усіх або обраних) – автентифікація за ДНПФСЗН, або ознаками, що аналізуються, можуть бути часові інтервали між натисканням будь-яких сусідніх символів слова (усіх або обраних), але при цьому вони повинні порівнюватися з відповідними ознаками навчальних зразків тільки такого ж слова - автентифікація за ДНСЗН. У першому випадку не має особливого значення з навчального зразка, якого саме слова обраного набору вводиться символ, головне щоб цей символ був з фрагменту, який повторюється у всіх словах цього набору. В другому ж випадку навпаки, не має значення часовий інтервал між натисканням яких саме символів слова аналізується, головне щоб ця характеристика, що аналізується, порівнювалася з відповідною ознакою з навчального зразка тільки такого ж слова. Відповідно й алгоритми ВЯАП у цих двох випадках будуть відрізнятися один від одного (хоча і не дуже сильно).

У першому випадку, коли аналізуються часові інтервали між натисканням сусідніх символів спільної частини для всіх слів набору (автентифікація за ДНПФСЗН), алгоритм ВЯАП складається з наступних етапів:

1. Виключити з бази даних навчальних зразків записи з грубими помилками. Алгоритм виконання цієї операції буде розглянутий пізніше.

2. Проаналізувати дані в базі навчальних зразків для всіх користувачів. В резу-

льтаті аналізу визначити слова, для яких накопичені експериментальні дані для користувачів, які будуть автентифікуватися в майбутньому. Вибрати з них ті, котрі необхідно протестувати (усі або вибірково) та записати ці слова в масив з ім'ям *Slovo*.

3. Визначити n_{all} - кількість ознак, які аналізуються в даному алгоритмі, що дорівнює довжині фрагменту, який повторюється у всіх словах масиву з ім'ям *Slovo*.

4. Для кожного з користувачів (*t* змінювати від 1 до *l*) прочитати в масив *U* дані з бази даних навчальних зразків та обчислити за відповідними формулами математичне очікування M_{it} та дисперсію S_{it} кожної ознаки з фрагмента, який повторюється у всіх словах масиву з ім'ям *Slovo* (*i* змінювати від 1 до n_{all}).

5. Для кожної ознаки з обраного фрагмента (*i* змінювати від 1 до n_{all}) розрахувати за формулою значення критерію H_i.

6. Відсортувати ознаки за значенням знайденої дисперсії S_{it} та вивести на екран обраний спільний фрагмент слова та результати сортування для подальшого аналізу адміністратором.

7. Відсортувати ознаки за значенням H_i та вивести на екран обраний спільний фрагмент слова та результати сортування для подальшого аналізу адміністратором.

8. Проаналізувати всі отримані результати про якість ознак та, у залежності від необхідного рівня безпеки, котрий повинна забезпечувати система захисту, що створюється, по сукупності двох отриманих характеристик (S_{it} та H_i) визначити чи є в фрагменті, який повторюється у всіх обраних словах, необхідна кількість символів (*n*), які придатні для використання їх для розпізнаванні. Та серед цих символів знайти найбільш якісні, для рішення поставленої задачі, ознаки.

В другому випадку, коли порівнюються відповідні ознаки тільки однакових слів (автентифікація за ДНСЗН), алгоритм ВЯАП зміниться та буде складатися з наступних етапів:

1. Виключити з бази даних навчальних зразків записи з грубими помилками. Алгоритм виконання цієї операції, як вже було сказано, буде розглянутий пізніше.

2. Проаналізувати дані в базі навчальних зразків для всіх користувачів. В результаті аналізу визначити спільні для всіх користувачів слова, для яких

накопичені експериментальні дані. Записати ці слова в масив з ім'ям *Slovo*, а їхню кількість в змінну *s*.

3. Записати 1 в змінну k_s: k_s=1, де k_s - номер слова в масиві з ім'ям *Slovo*.

4. Вибрати з масиву з ім'ям *Slovo* слово з номером k_s.

5. Визначити n_{all} - кількість ознак, які аналізуються в даному алгоритмі, що дорівнює довжині k_s-го слова.

6. Для кожного з користувачів (*t* змінювати від 1 до *l*) прочитати в масив *U* дані для k_s-го слова (*i* змінювати від 1 до n_{all}) з бази даних навчальних зразків та підрахувати, відповідно, математичне очікування M_{it} та дисперсію S_{it} кожної ознаки k_s-го слова (*i* змінювати від 1 до n_{all}).

7. Для кожної ознаки з k_s-го слова (*i* змінювати від 1 до n_{all}) розрахувати за формулою значення критерію H_i.

8. Відсортувати ознаки з k_s-го слова за значенням знайденої дисперсії S_{it} та вивести на екран k_s-те слово та результати сортування для подальшого аналізу адміністратором.

9. Відсортувати ознаки k_s-го слова за значенням H_i та вивести на екран k_s-те слово та результати сортування для подальшого аналізу адміністратором.

10. Збільшити k_s на 1: k_s= k_s+1.

11. Якщо k_s>s, тоді перейти до пункту 12, інакше перейти до пункту 4.

12. Проаналізувати всі отримані результати про якість ознак з усіх розглянутих слів та, у залежності від необхідного рівня безпеки, котрий повинна забезпечувати система захисту, що створюється, по сукупності двох отриманих характеристик (S_{it} та H_i) визначити в яких з розглянутих слів є необхідна кількість символів (*n*), які придатні для використання їх для розпізнаванні. Та в цих словах знайти найбільш якісні, для рішення поставленої задачі, ознаки.

Якщо ж в якості ключової фрази передбачається використовувати не одне з слів (яке обирається щоразу випадковим чином) підібраного по якимось параметрам набору, а постійно якусь одну фразу (автентифікація за ДНПКФ) та в базі навчальних зразків (пробної) накопичена інформація з декількох варіантів такої фрази, тоді

алгоритм ВЯАП буде подібний алгоритму ВЯАП, котрий використовується у випадку, коли для автентифікації використовується одне випадково обране слова з заздалегідь підібраного набору та аналізуються часові інтервали між натисканням усіх сусідніх символів, що порівнюються з відповідними ознаками тільки такого ж слова. Але в розглянутому зараз випадку записуватися в масив з ім'ям *Slovo* будуть не слова, а фрази, тобто в кожний елемент цього масиву буде записуватися не одне слово, а один з варіантів передбачуваної ключової фрази. Відповідно й оброблятися потім будуть часові інтервали між натисканням сусідніх символів не одного слова, а фрази. Інші принципи роботи даного алгоритму не змінюються.

Який з розглянутих варіантів використовувати, повинен вирішувати той, хто розробляє цю систему захисту та відповідно адміністратор (або яка-небудь інша людина, яка відповідальна за безпеку конкретної комп'ютерної системи) комп'ютерної системи при виборі та налаштовуванні конкретної системи захисту.

Але у всіх цих варіантах алгоритму ВЯАП мається на увазі, що вже відома необхідна кількість символів (*n*) для забезпечення даною системою автентифікації необхідного рівня безпеки комп'ютерної системи. В кожному конкретному випадку, в залежності від конкретних користувачів, від обраного різновиду автентифікації (автентифікація за ДНПКФ, за ДНВТ, за ДНСЗН, за ДНПФСЗН), від якості обраних ознак, значення цього параметра буде відрізнятися та повинно підбиратися адміністратором або іншим працівником, який відповідає за безпеку даної комп'ютерної системи. Тому, з огляду на те, що визначення якості аналізованих параметрів повинне виконуватися як мінімум двічі: після накопичення «пробної» бази даних навчальних зразків та після накопичення повної бази даних навчальних зразків (також цей етап може виконуватися для коректування роботи системи автентифікаціїі згодом, при значному збільшенні кількості навчальних зразків в базі даних з появою нових користувачів даної комп'ютерної системи), то, можна сказати, що при виконанні алгоритму ВЯАП після накопичення «пробної» бази даних навчальних зразків значення змінної *n* краще вибрати максимально можливим (тобто всі символи аналізованого фрагмента), а при виконанні алгоритму ВЯАП після накопичення повної бази даних навчальних зразків значення змінної *n* вже

повинне бути підібране, на основі навчальних зразків, які є. У даній роботі було зроблене припущення, що чим більше ознак, що аналізуються, тим вище імовірність правильної автентифікації. Для перевірки даного припущення було проведено ряд експериментів, результати, яких приведені в розділі 4 даної роботи. Однак при виборі кількості аналізованих параметрів (n) треба враховувати те, що при збільшенні n не тільки підвищується якість розпізнавання, але й збільшується обсяг інформації, яка зберігається та обробляється для розпізнавання, а, отже, збільшується обсяг необхідної пам'яті (оперативної та постійної) та збільшуються часові витрати для виконання процесу автентифікації. Тобто треба зіставляти якість автентифікації, яка досягається з витраченими ресурсами.

Як вже було сказано, одним з етапів алгоритму визначення якості параметрів, що аналізуються, та подальшого налаштовування системи автентифікації є виключення з бази даних навчальних зразків записів з грубими помилками однієї з ознак. Дійсно для підвищення якості розпізнавання необхідно (бажано) провести відбір навчальних даних. Ця необхідність пояснюється тим, що КП є динамічною характеристикою, тобто цей параметр не має стовідсоткову стабільність. Тому що на відміну від статичних характеристик (відбиток пальця, сітківка ока, райдужна оболонка ока і т.д.), КП піддається впливу різних факторам (настрій, втома, навколишнє оточення і т.д.). Будь-яка людина може при наборі тексту відволіктися на якісь зовнішні обставини (вона може задуматися, у неї може щось заболіти, їй може щось чи хтось перешкодити) і через це помилитися або зробити незвично велику (чи навпаки маленьку) паузу. І якщо при цьому характеристики її набору фіксувалися в базі даних навчальних зразків, тоді при наступному аналізі цих даних буде помітна значна нестабільність параметрів, що аналізуються, особливо якщо помилок та часових відхилень у даного користувача було досить багато. Але якщо невелика нестабільність характерна для КП і з цією проблемою ІНМ досить непогано справляються, а саме цей механізм використовується в даній роботі для автентифікації користувачів, то великий розкид значень будь-якого аналізованого параметра призводить до зниження точності розпізнавання і, отже, до зниження ІПР користувачів. Тому, можна сказати, що виключення навчальних даних з грубою

помилкою хоча б однієї з ознак, що аналізуються, приведе до певного підвищення якості розпізнавання. Можна використовувати різні алгоритми визначення того, які навчальні зразки варто виключати. У даній роботі проаналізований алгоритм, описаний у [49], який далі будемо називати алгоритмом виключення навчальних зразків з грубими помилками за допомогою сортування та виключення крайніх значень (алгоритм ВНЗСВКЗ) та запропонований свій алгоритм, що будемо називати алгоритмом виключення навчальних зразків з грубими помилками за допомогою порівняння ознаки з його середнім значенням (алгоритм ВНЗПОСЗ). На підставі результатів проведених експериментів, результати яких будуть викладені в розділі 4, можна зробити висновок, що запропонований і використаний у даній роботі алгоритм ВНЗПОСЗ у більшості випадків є більш ефективним засобом для підвищення якості розпізнавання користувачів комп'ютерної системи. Розглянемо ці два алгоритми докладніше.

Нагадаємо, що:

$$U = \{u_{111}, u_{112}, u_{113}, ..., u_{ijt}, ..., u_{nz_l l}\};\ \ 1 \le i \le n;\ 1 \le j \le z_t;\ 1 \le t \le l;$$

де U- масив навчальних зразків; u_{ijt} - часовий інтервал, необхідний для натискання i-го аналізованого символу ключової фрази з j-го зразка t-го користувача; n - кількість ознак, що аналізуються (обраних по алгоритму ВЯАП); z_t - кількість навчальних зразків для t-го користувача; l - кількість користувачів, для яких є інформація в базі даних навчальних зразків.

Як і під час визначення якості аналізованих параметрів, алгоритм виключення записів з грубими помилками буде відрізнятися при автентифікації за ДНПФСЗН (без відбору за словами) та при автентифікації за ДНСЗН (з відбором за словами). Алгоритм, що використовується у першому випадку, розглянемо більш докладно, а потім укажемо, чим буде відрізнятися алгоритм у другому випадку.

Спочатку розглянемо алгоритм ВНЗСВКЗ (з виправленням деяких неточностей та адаптувавши його під дану задачу). У використаному джерелі приводиться алгоритм виключення навчальних зразків з грубими помилками по одній з ознак і для одного користувача. У нашому ж випадку цей алгоритм буде складатися з

наступних етапів:

1. Переписати всі навчальні зразки з головної таблиці бази даних у проміжну таблицю, пронумерувавши в ній всі навчальні зразки. Структура проміжної таблиці така ж, як у головної, але з додаванням одного додаткового стовпця для номера навчального зразка. Цей етап необхідний для збереження первісної послідовності навчальних зразків. У [49] цього етапу нема, але для дотримання рівних умов при проведенні експериментів для порівняння ефективності від застосування даного алгоритму й алгоритму ВНЗПОСЗ, у даній роботі цей етап включений в алгоритм ВНЗСВКЗ.

2. Ініціалізувати змінну циклу t: t=1.

3. Прочитати з проміжної таблиці бази даних навчальних зразків усі дані для t-ого користувача.

4. Присвоїти 1 змінній nn: nn=1. Переписати зі змінної z_t значення в змінну nk: $nk=z_t$. Цього пункту в [49] нема, але, з огляду на те, що в нашому випадку на грубі помилки перевіряється не одна ознака, як це показано в [49], а послідовно декілька ознак, то для збереження значення змінної z_t та для зручності реалізації даного алгоритму доданий цей пункт.

5. Ініціалізувати змінну циклу i: i=1.

6. Відсортувати навчальні зразки для t-ого користувача за зростанням i-ї ознаки.

7. Виконати наступні переприсвоєння: $j_n=nn$, $L_n=j_n+1$, $L_f=nk$. У [49] цей пункт трохи відрізняється від даного представлення. Це розходження викликане додаванням у нашому випадку пункту 4.

8. Обчислити за наступними формулами:

$$M = \frac{\sum_{j=L_n}^{L_f} u_{ijt}}{z_t - 1};$$

$$S = \sqrt{\sum_{j=L_n}^{L_f} (u_{ijt} - M)^2 / (z_t - 2)};$$

$$T = \left| (u_{ijt} - M) \right| / S.$$

У цьому пункті в [49] при обчисленні змінної T не брався модуль від чисельника дробі, але при такій неточності (помилці) програма буде працювати неправильно (будуть виключатися не всі навчальні зразки з грубими помилками), тому для правильної роботи програми в нашому випадку додана операція взяття модуля від чисельника дробі.

9. По таблиці t-розподілу Ст'юдента для p=0.95 та числа ступенів свободи nk – 2 визначити величину Zn. З огляду на те, що в нашому випадку використовується велика кількість навчальних зразків для кожного користувача (більш ніж 120), тому для постійного значення змінної p змінна Zn завжди буде дорівнювати постійному числу (Zn=1.960).

10. Якщо $T>Zn$, тоді, в [49] пропонується, відкинути даний навчальний зразок, а зразки, що залишилися, перенумерувати, змінивши при цьому значення змінної, у якій зберігається кількість навчальних зразків. У нашому випадку пропонується ці зразки згодом у цьому алгоритмі просто не розглядати та, відповідно, не записувати їх у базу даних навчальних зразків для використання їх при наступній автентифікації, тому пропонується наступне: якщо $j_n=L_n-1$, тоді збільшити nn на 1 ($nn=nn+1$) та перейти до пункту 7, якщо $j_n=L_f+1$, тоді зменшити nk на 1 ($nk=nk-1$) та перейти до пункту 7.

Інакше якщо $T\leq Zn$ та якщо $j_n\neq nk$, тоді виконати наступні переприсвоєння: $j_n=nk$, $L_n=nn$, $L_f=j_n-1$ та перейти до пункту 8, інакше перейти до пункту 11. У цьому пункті даного алгоритму також є розбіжності з [49]: по-перше, у [49] пропонується перейти до операцій, що знаходяться, в даному випадку, в пункті 7, а це приведе до неправильної роботи програми, тому ця неточність (помилка) виправлена та перехід здійснюється до пункту 8; інші розбіжності, викликані адаптацією даного алгоритму під нашу задачу.

11. Збільшити i на 1: $i=i+1$.

12. Якщо $i>n$, тоді перейти до пункту 13, інакше перейти до пункту 6.

13. Записати всі навчальні зразки з номерами від nn до nk (включно) для t-го користувача у відповідну таблицю бази даних навчальних зразків для використання їх при подальшій автентифікації, розташувавши їх у тій послідовності, у якій вони

знаходилися в головній таблиці бази даних.

14. Збільшити t на 1: $t=t+1$.

15. Якщо $t>l$, тоді закінчити виключення навчальних зразків з грубими помилками, інакше перейти до пункту 3.

А тепер розглянемо алгоритм ВНЗПОСЗ. Він складається з наступних етапів:

1. Ініціалізувати змінну циклу t: $t=1$.

2. Прочитати з вихідної бази даних навчальних зразків усі дані для t-ого користувача.

3. Ініціалізувати змінну циклу i: $i=1$.

4. Обчислити середнє арифметичне значення i-ї аналізованої ознаки t-го користувача за наступною формулою:

$$Sr_{it} = \frac{\sum_{j=1}^{z_t} u_{ijt}}{z_t}.$$

5. Збільшити i на 1: $i=i+1$.

6. Якщо $i>n$, тоді перейти до пункту 7, інакше перейти до пункту 4.

7. Ініціалізувати змінну циклу j: $j=1$.

8. Обнулити змінну p: $p=0$, де p – кількість аналізованих ознак у j-ому зразку, без грубих помилок.

9. Ініціалізувати змінну циклу i: $i=1$.

10. Якщо $\left|u_{ijt} - Sr_{it}\right| \le k * Sr_{it}$, тоді $p=p+1$, де k -коефіцієнт, який обирається в залежності від необхідної точності розпізнавання. Чим менший цей коефіцієнт, тим більш точні будуть накопичені навчальні дані, але тим менше їх залишиться після відбору.

11. Збільшити i на 1: $i=i+1$.

12. Якщо $i>n$, тоді перейти до пункту 13, інакше перейти до пункту 10.

13. Якщо $p=n$, тоді записати цей j-й навчальний зразок у відповідну таблицю бази даних навчальних зразків для подальшого використання при автентифікації, інакше відкинути його.

14. Збільшити j на 1: $j=j+1$.

15. Якщо $j>z_t$, тоді перейти до пункту 16, інакше перейти до пункту 8.

16. Збільшити t на 1: $t=t+1$.

17. Якщо $t>l$, тоді закінчити виключення навчальних зразків з грубими помилками, інакше перейти до пункту 2.

Якщо в базі даних зберігаються навчальні зразки не для одного слова (фрази), а для декількох (за винятком автентифікації за ДНПФСЗН, при якій хоча й використовуються різні слова, але аналізується тільки фрагмент, який повторюється), тоді обраний один з цих двох алгоритмів (ВНЗСВКЗ або ВНЗПОСЗ) треба виконати для кожного слова (фрази), попередньо визначивши довжину даного слова чи фрази, у випадку якщо планується аналізувати всі ознаки, або попередньо визначивши кількість ознак, що аналізуються (можливо, їх номера). Ця довжина або кількість і буде, потім, кількістю тестуємих ознак, у випадку якщо виключаються дані з грубими помилками з пробної бази даних, або кількістю (n) ознак, що аналізуються, при виконанні процесу автентифікації.

Проаналізувавши алгоритм ВНЗСВКЗ та алгоритм ВНЗПОСЗ можна сказати, що другий алгоритм є не тільки більш ефективним у більшості випадків, але і більш простим у використанні. Для підтвердження цього було проведено ряд експериментів, результати яких будуть викладені в розділі 4. Ці два алгоритми тестувались у рівних умовах, тобто використовувалися ті ж самі навчальні зразки, з тими ж самими параметрами налаштовування та тестування виконувалося на тому ж самому комп'ютері. Значення коефіцієнта k, який використовувався в алгоритмі ВНЗПОСЗ було обрано рівним 1.

В результаті аналізу написаних на базі цих алгоритмів програм та аналізу результатів проведених експериментів можна сказати, що алгоритм ВНЗПОСЗ:

1. Простіший.
2. Витрачає менш часових та обчислювальних ресурсів.
3. Є в більшості випадків більш ефективним.

Перші дві переваги алгоритму ВНЗПОСЗ пояснюються тим, що:

1. У цьому алгоритмі виконується відбір відразу за всіма ознаками. Що аналізуються, на відміну від алгоритму ВНЗСВКЗ, у якому багато дій виконуються

по декілька разів.

2. На відміну від алгоритму ВНЗПОСЗ, в алгоритмі ВНЗСВКЗ для кожної ознаки виконується сортування, що займає досить багато часу та оперативної пам'яті, особливо якщо масив, який сортується, є великим.

Велика ефективність в більшості випадків застосування алгоритму ВНЗПОСЗ виражається в наступному:

1. Аналіз якості аналізованих ознак після виконання виключення з бази даних навчальних зразків з грубими помилками за алгоритмом ВНЗПОСЗ показав, що всі параметри мають приблизно однаковий, досить непоганий, рівень якості, а після виключення по алгоритму ВНЗСВКЗ рівень якості для різних аналізованих ознак сильно розрізняється між собою (у деяких дуже гарний, а в деяких дуже поганий).

2. При відносно великій кількості аналізованих ознак алгоритм ВНЗПОСЗ забезпечує кращу якість розпізнавання, а при малій кількості (недостатній для якісного розпізнавання) аналізованих ознак алгоритм ВНЗСВКЗ виявляється більш ефективнішим.

Усе це підтверджує те, що використання алгоритму ВНЗПОСЗ у даному випадку виявилося переважніше.

Подібний відбір бажано проводити не тільки при обробці результуючої (накопиченої в достатньому для розпізнавання обсязі) бази даних навчальних зразків, але і при обробці «пробної» бази даних навчальних зразків на етапі підбора оптимальної ключової фрази. Однак на цьому етапі, можливо, можна виконувати більш грубий відбір навчальних даних, тобто k може мати більше значення, чим на етапі розпізнавання. Вплив цього коефіцієнта на точність автентифікації визначається за допомогою експериментів. Для цього в даній роботі було проведено ряд таких експериментів, результати яких будуть показані в розділі 4.

Наступним підготовчим етапом перед виконанням безпосередньо процесу автентифікації є процес налаштовування даної системи автентифікації. Це досить важливий етап роботи даної системи автентифікації. При правильному налаштовуванні на цьому етапі конкретних параметрів, адміністратор або інша людина, яка відповідальна за забезпечення безпеки даної комп'ютерної системи, може значно збільшити точність автентифікації користувачів, при цьому

збільшивши рівень безпеки комп'ютерної системи, що захищається. Необхідність налаштовування таких параметрів, як кількість зразків, що аналізуються, для кожного користувача (z_t) та кількість ознак, що аналізуються, (n) вже було пояснено раніше в цьому підрозділі. Тепер проаналізуємо, від яких ще параметрів залежить точність розпізнавання. Наскільки серйозно вони впливають на ІПР користувачів, було проаналізовано за допомогою ряду проведених експериментів, результати яких будуть представлені в розділі 4.

Одним з параметрів, які налаштовуються, є параметр, що вказує на виконання усереднення навчальних зразків (*usr*). Можлива доцільність усереднення навчальних даних пояснюється наступними причинами. Як вже було сказано, користувач при наборі тексту може помилитися або відволіктися. Тому навіть для найуважніших користувачів в базі даних будуть помилкові навчальні зразки. Ті зразки, у яких хоч для однієї ознаки, що аналізується, присутня груба помилка звичайно повинні виключатися за допомогою одного з відповідних алгоритмів (алгоритм ВНЗСВКЗ або алгоритм ВНЗПОСЗ), а для зменшення негативного ефекту від невеликих помилок навчальні зразки усереднюють. Тобто під час побудови ІНМ будуть використовуватися не окремо кожен навчальний зразок, а усереднені значення кожної ознаки з групи усереднених зразків. По скільки навчальних зразків усереднювати (чому дорівнює параметр K_{usr}) необхідно вказати на етапі налаштовування програми, при установці ознаки усереднення навчальних зразків у включений стан (*usr*=істина). Наприклад, якщо K_{usr}=10, тоді з бази даних читається по десять навчальних зразків, потім відповідні ознаки кожного навчального зразка додаються (окремо всі перші ознаки, окремо всі другі ознаки і т.д.) та діляться на десять, після чого ці усереднені ознаки будуть використовуватися при проведенні автентифікації, в якості ознак одного усередненого навчального зразка. При виконанні такого усереднення зменшується похибок в даних, що аналізуються (навчальних зразках), але при цьому для виконання автентифікації залишається менше навчальних зразків (у K_{usr} раз). Тому виникає питання, що важливіше для правильності розпізнавання: незначне збільшення точності навчальних зразків за рахунок зменшення їхньої кількості або збільшення кількості навчальних зразків за

рахунок збереження в них невеликих похибок. Для відповіді на це питання в даній роботі було проведено ряд експериментів, результати яких приведені в розділі 4.

На етапі налаштовування також можна вказувати чи проводити при виконанні автентифікації відбір за словами (параметр *ots*), тобто для автентифікації, при побудові імовірнісної нейтронної мережі використовувати всі які є навчальні зразки або тільки зразки для слова, що вводиться в цій момент. Включення цього параметра (*ots*=істина) може знадобитися в наступних випадках:

1. Для виконання автентифікації за ДНСЗН. Тому що при такому різновиді автентифікації слово для введення вибирається випадковим чином із заздалегідь створеного набору, тому для аналізу з бази даних треба вибрати навчальні зразки тільки для введеного слова.

2. Для виключення похибки в значеннях першої ознаки під час автентифікації за ДНПФСЗН. Це викликано наступною причиною. При такому різновиді антентифікації слово для введення обирається випадковим чином із заздалегідь створеного набору, а для розпізнавання використовуються часові інтервали між натисканням двох сусідніх символів тільки в фрагменті, який повторюється у всіх словах набору. При цьому, символ, що знаходиться перед першим символом фрагмента, що повторюється, у всіх словах набору різний, тому перша ознака може бути з похибкою. Для виключення цього негативного фактора іноді виконують відбір за словами.

У першому випадку виконання відбору за словами є обов'язковою умовою даного різновиду автентифікації. В другому випадку відбір за словами зменшує похибку навчальних даних, але при цьому значно (у *s* разів) зменшує кількість навчальних зразків, що аналізуються. У цьому випадку автентифікація за ДНПФСЗН зводиться до автентифікації за ДНСЗН, але при цьому аналізується менша, чим під час автентифікації за ДНСЗН кількість символів. Тому, можливо, і нема сенсу проводити відбір за словами під час автентифікації за ДНПФСЗН. Для того, щоб перевірити це припущення в даній роботі було проведено ряд експериментів, результати яких представлені в розділі 4.

Після виконання підбора ключової фрази, накопичення бази навчальних зразків, виключення зразків із занадто грубими помилками та проведення

необхідного налаштовування параметрів системи може виконуватися сам процес автентифікації користувачів.

Розглянемо алгоритм процесу автентифікації користувачів ІС за КП за допомогою ІНМ (алгоритм ПАКПІНМ). Цей алгоритм складається з наступних етапів:

1. Запропонувати користувачу набрати на клавіатурі комп'ютера ключову фразу (якщо виконується автентифікація за ДНПКФ) або слово з заздалегідь обраного набору (якщо виконується автентифікація за ДНСЗН або за ДНПФСЗН).

2. При введенні користувачем ключової фрази (або слова) заміряти параметри, що аналізуються, (час між натисканням сусідніх символів ключової фрази), тобто створити невідомий зразок *X*, значення ознак якого будуть значеннями елементів вхідного шару. При цьому невідомий зразок має вигляд:

$$X = \{x_1, x_2, x_3, ..., x_i, ..., x_n\};\ \ 1 \le i \le n;$$

де x_i – *i*-а аналізована ознака в невідомому зразку.

3. Для того, щоб згодом для визначення активності шару зразків можна було б використовувати більш просту формулу, усі вхідні вектори повинні бути нормалізовані. Тому необхідно нормалізувати невідомий зразок. Для цього:

3.1.Обчислити:

$$S_x = \sum_{i=1}^{n} x_i^2.$$

3.2.Ініціалізувати змінну циклу *i*: *i*=1.

3.3.Обчислити:

$$x_{n_i} = \frac{x_i}{\sqrt{S_x}}.$$

3.4.Збільшити *i* на 1: *i*=*i*+1.

3.5.Якщо *i*>*n*, тоді перейти до пункту 4, інакше перейти до підпункту 3.3.

4. Сформувати масив навчальних зразків. Для цього:

4.1.Прочитати ті навчальні зразки, по яким будується в даному випадку ІНМ (це залежить від різновиду автентифікації, що виконується).

4.2.Якщо на етапі налаштовування було встановлено *ots*=істина (тобто необхідно виконати відбір за словами), тоді залишити серед прочитаних у підпункті

4.1 навчальних зразках тільки ті, котрі відповідають введеному в даному сеансі автентифікації слову (фразі), інакше перейти до підпункту 4.3.

4.3.Якщо на етапі налаштовування було встановлено *usr*=істина (тобто необхідно виконати усереднення навчальних зразків по K_{usr}), тоді з тих навчальних зразків, що були накопичені в підпункті 4.1 і, можливо, відібрані в підпункті 4.2 сформувати масив навчальних зразків, при цьому, кожен навчальний зразок цього масиву повинен складатися з ознак рівних середньоарифметичним значенням відповідних ознак K_{usr} навчальних зразків, які йдуть підряд (тобто замість K_{usr} навчальних зразків створюється один), інакше перейти до пункту 5.

5. З причин, зазначених в пункті 3, необхідно нормалізувати навчальні зразки. Для цього:

5.1. Ініціалізувати змінну циклу *t*: *t*=1.

5.2. Ініціалізувати змінну циклу *j*: *j*=1.

5.3. Обнулити змінну $S_{u_{jt}}$: $S_{u_{jt}} = 0$.

5.4. Обчислити:

$$S_{u_{jt}} = \sum_{i=1}^{n} u_{ijt}^2.$$

5.5. Ініціалізувати змінну циклу *i*: *i*=1.

5.6. Обчислити ознаки (w_{ijt}) нормалізованих навчальних зразків, що згодом будуть використовуватися в якості вагових значень зв'язків, що входять в елементи шару зразків:

$$w_{ijt} = \frac{u_{ijt}}{\sqrt{S_{u_{jt}}}}.$$

5.7. Збільшити *i* на 1: *i*=*i*+1.

5.8. Якщо *i*>*n*, тоді перейти до підпункту 5.9, інакше перейти до підпункту 5.6.

5.9. Збільшити *j* на 1: *j*=*j*+1.

5.10.Якщо $j>z_t$, тоді перейти до підпункту 5.11, інакше перейти до підпункту 5.3.

5.11.Збільшити *t* на 1: *t*=*t*+1.

5.12.Якщо *t*>*l*, тоді перейти до пункту 6, інакше перейти до підпункту 5.2.

6. Розрахувати ефективність кожного елемента шару зразків. Для цього:

6.1.Ініціалізувати змінну циклу *t*: *t*=1.

6.2.Ініціалізувати змінну циклу *j*: *j*=1.

6.3.Зважаючи на те, що усі вхідні вектори нормалізовані, ефективність кожного елемента шару зразків можна обчислити за формулою, яка у нашому випадку прийме наступний вигляд:

$$O_{jt} = \exp\left(\frac{\sum_{i=1}^{n} x_{n_i} w_{ijt} - 1}{\sigma^2}\right).$$

6.4.Збільшити *j* на 1: *j*=*j*+1.

6.5.Якщо $j>z_t$, тоді перейти до підпункту 6.6, інакше перейти до підпункту 6.3.

6.6.Збільшити *t* на 1: *t*=*t*+1.

6.7.Якщо *t*>*l*, тоді перейти до підпункту 7, інакше перейти до підпункту 6.2.

7. Виконати функції елементів шару додавання, тобто приплюсувати вихідні значення елементів шару зразків (активності) кожного класу окремо. Для цього:

7.1.Ініціалізувати змінну циклу *t*: *t*=1.

7.2.Обчислити:

$$S_{O_t} = \sum_{j=1}^{z_t} O_{jt}.$$

7.3.Збільшити *t* на 1: *t*=*t*+1.

7.4.Якщо *t*>*l*, тоді перейти до пункту 8, інакше перейти до підпункту 7.2.

8. Знайти елемент шару додавання з максимальною активністю, тобто знайти користувача, який найбільш імовірно є хазяїном пред'явлених біометричних характеристик. Для цього знайти в масиві S_{O_t} максимальний елемент $\max_{S_{O_t}}$.

9. Якщо цей найбільш імовірний користувач є хазяїном пред'явленого ідентифікатора, тоді робиться висновок, що автентифікований користувач дійсно той за кого себе видає.

У цьому алгоритмі можливі варіації. Наприклад, у пункті 8 можна шукати не один елемент шару додавання з максимальною активністю, а декілька (наприклад, 3)

і якщо при цьому в пункті 9 виявляється, що пред'явлений ідентифікатор належить не найбільш імовірному користувачу, але одному з відібраних у пункті 8, тоді автентифікованому користувачу не відмовити в доступі, а запропонувати йому пройти автентифікацію повторно цим же або іншим способом. Крім того, для економії ресурсів (часу та пам'яті) пункти 6 і 7 можна виконувати не послідовно, а паралельно, тобто при пошуку ефективність кожного елемента шару зразків відразу ж додавати її до суми вихідних значень елементів шару зразків відповідного класу. Це і було зроблено в даній роботі.

Тепер побудуємо загальний алгоритм роботи системи автентифікації за КП за допомогою ІНМ (алгоритм САКПІНМ). Цей алгоритм складається з наступних кроків:

1. Довільний вибір декількох варіантів ключової фрази.

2. Накопичення «пробної» бази даних навчальних зразків для кожного варіанта за алгоритмом ННЗЗЗЗП.

3. Виключення з кожної «пробної» бази даних навчальних зразків записи з грубими відхиленнями значення хоча б однієї з характеристик від її середнього значення (грубий відбір) за алгоритмом ВНЗПОСЗ.

4. Аналіз характеристик з «пробних» баз даних навчальних зразків для визначення ключової фрази з найкращими для класифікації за цією фразою характеристиками за алгоритмом ВЯАП.

5. Перевірка чи зможе обрана ключова фраза забезпечити потрібну точність класифікації. Якщо ні, тоді повернення до першого етапу. Якщо так, тоді перехід до наступного етапу.

6. Накопичення необхідного обсягу бази даних навчальних зразків для обраної ключової фрази за алгоритмом ННЗЗЗЗП.

7. Виключення з результуючої бази даних навчальних зразків запису з грубими відхиленнями значення хоча б однієї з характеристик від її середнього значення (допустиме відхилення значення аналізованої характеристики від її середнього значення залежить від необхідної точності класифікації, тобто від необхідного рівня безпеки комп'ютерної системи, що захищається) за алгоритмом ВНЗПОСЗ.

8. Вибір у результуючій базі даних навчальних зразків найкращих характеристик, які і будуть згодом використовуватися для автентифікації, за алгоритмом ВЯАП.

9. Налаштовування параметрів системи автентифікації, від яких залежить якість автентифікації.

10. Виконання автентифікації користувачів комп'ютерної системи за алгоритмом ПАКПІНМ.

Усі кроки, крім останнього, виконуються один раз під час установки даної системи автентифікації та згодом, якщо буде необхідно її підлаштувати.

2.5. Висновки до другого розділу

В другому розділі запропоновано метод первинної обробки зразків КП, який за рахунок аналізу спектральних характеристик почерку користувача, дозволяє виключити хибні зразки почерку, які є нехарактерними і викликані випадковими помилками користувача при наборі тексту і відрізняється від еталонного методу (метод Расторгуєва С.П.) [49] тим, що забезпечує більш високу (в 2-3 рази) та рівномірну якість характеристик почерку та, завдяки цьому, збільшується ІПР користувачів на 10-20%. Після чого вдосконалено метод АК ІС за їх КП, який за рахунок використання для розпізнавання ІНМ та виконання первинної обробки зразків КП, збільшує ІПР користувачів ІС, в порівнянні з еталонним методом (метод ВШНМ), на 4-8%. Розроблений метод АК за їх КП має сенс використовувати не тільки, як самостійний метод автентифікації, а й як один з етапів багатофакторної автентифікації. Крім того, даний метод розпізнавання можна використовувати не тільки для АК ІС, а й для моніторингу функціонального стану працівників критичних професій та під час прийому на роботу, для аналізу поточного стану таких характеристик людини, як уважність, сконцентрованість, акуратність.

РОЗДІЛ 3. МЕТОД АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ ІНФОРМАЦІЙНИХ ЗА ЇХ РУКОПИСНИМ ПОЧЕРКОМ

3.1. Постановка задачі автентифікації за рукописним почерком

Даний розділ присвячено розробці методу АК ІС за їх РП (АКРП) та методу первинної обробки зразків РП користувачів ІС (ПОЗРП), необхідного для виконання АК ІС за їх РП. В якості механізму розпізнавання використовується ІНМ. Для передачі зразка РП користувача в комп'ютер, використовувався графічний планшет (ГП), як один з варіантів пристрою з сенсорним екраном, який застосовується для динамічного передавання характеристик РП користувача. Тобто в даному розділі будується функція R – функція розпізнавання людей за паролем, який вводиться за допомогою ГП [116-123].

Для розробки методів АКРП та ПОЗРП використовується наступна модель даних:

$$\boldsymbol{US\_R}=\{\bigcup_{p=1}^{m} US\_R_p\}=\{US\_R_1,\ US\_R_2,\ \ldots,\ US\_R_p,\ \ldots,\ US\_R_x\ \ldots,\ US\_R_m\};\quad p=\overline{1,m};\ m$$

– кількість членів множини ***US_R***; ***US_R*** – множина користувачів, які можуть спробувати отримати доступ до системи, що захищається;

$$\boldsymbol{USL\_R}=\{\bigcup_{t=1}^{l} USL\_R_t\}=\{USL\_R_1,\ USL\_R_2,\ \ldots,\ USL\_R_t,\ \ldots,\ USL\_R_d,\ \ldots,\ USL\_R_l\};$$

$t=\overline{1,l};\ \ l$ – кількість легальних користувачів; ***USL_R*** – множина легальних користувачів даної системи; здійснюється умова, що $\boldsymbol{USL\_R}\subset\boldsymbol{US\_R}$;

US_R_X – користувач ІС, який проходить автентифікацію; здійснюється умова, що $US\_R_X\in\boldsymbol{US\_R}$;

USL_R_d – легальний користувач, за якого видає себе авторизована сторона, що автентифікується; здійснюється умова, що $((USL\_R_d\in\boldsymbol{US\_R})\wedge(USL\_R_d\in\in\boldsymbol{USL\_R}))$;

$$\boldsymbol{USB\_R}=\{\bigcup_{tb=1}^{lb} USB\_R_{tb}\}=\{USL\_R_1,\ USL\_R_2,\ \ldots,\ USL\_R_{tb},\ \ldots,\ USL\_R_{lb}\};\quad tb=\overline{1,lb};\ lb$$

– кількість порушників даної системи; ***USB_R*** – множина порушників даної системи, тобто користувачів, які в ній не зареєстровані, але намагаються отримати

до неї доступ; здійснюється умова, що $((\boldsymbol{USB\_R} \subset \boldsymbol{US\_R}) \wedge (\boldsymbol{USB\_R} \not\subset \boldsymbol{USL\_R}))$;

$\boldsymbol{T}=\{\bigcup_{az=1}^{v} T_{az}\}=\{T_1,\ T_2,\ \ldots,\ T_{az},\ \ldots,\ T_v\};\ az=\overline{1,v};\ v$ – кількість точок в зображенні ключової фрази (КФ); $\boldsymbol{T}$ – множина всіх точок зображення;

$\boldsymbol{TS}=\{\bigcup_{c=1}^{ks} \boldsymbol{TS}_c\}=\{\boldsymbol{TS}_1,\ \boldsymbol{TS}_2,\ldots,\ \boldsymbol{TS}_c,\ \ldots,\ \boldsymbol{TS}_{ks}\};\ \boldsymbol{TS}_c=\{\bigcup_{a=1}^{vc} TS_{c,a}\}=\{TS_{c,1},TS_{c,2},...,TS_{c,a},...,TS_{c,vc}\}$; відповідно множина $\boldsymbol{TS}$ приймає наступний вигляд: $\boldsymbol{TS}=\left\{\bigcup_{c=1}^{ks}\bigcup_{a=1}^{vc} TS_{c,a}\right\}=$ $\{\{TS_{1,1},TS_{1,2},...,TS_{1,a},...,TS_{1,vc}\},\ \{TS_{2,1},TS_{2,2},...,TS_{2,a},...,TS_{2,vc}\},...,\ \{TS_{c,1},TS_{c,2},...,TS_{c,a},...,TS_{c,vc}\},...,\{TS_{ks,1},TS_{ks,2},...,TS_{ks,a},...,TS_{ks,vc}\}\};\ c=\overline{1,ks};\ ks$ – кількість символів в КФ; $\boldsymbol{TS}$ – множина точок зображень символів КФ; $a=\overline{1,vc};\ vc$ – кількість точок в c-ому символі; $\boldsymbol{TS_c}$ – множина точок c-го символу; здійснюється умова, що $\boldsymbol{TS} \subset \boldsymbol{T}$;

$\boldsymbol{KTS}_c=\{\bigcup_{g=1}^{w} KTS_{c,g}\}=\{KTS_{c,1},KTS_{c,2},...,KTS_{c,g},...,KTS_{c,w}\};\ g=\overline{1,w};\ w$ – кількість КТ в c-ому символі; $\boldsymbol{KTS}_c$ – множина контрольних точок (КТ);

$\boldsymbol{AT\_R}=\{\bigcup_{i=1}^{n} AT\_R_i\}=\{AT\_R_1,\ AT\_R_2,\ \ldots,\ AT\_R_i,\ \ldots,\ AT\_R_n\};\ i=\overline{1,n};\ n$ – кількість ознак РП користувача; $\boldsymbol{AT\_R}$ – множина ознак РП користувача;

$\boldsymbol{AT1\_R}=\{\bigcup_{c=1}^{ks}\bigcup_{a1=1}^{3\cdot vc} ATX1\_R_{c,a1}\}=\{XT_{1,1},\ YT_{1,1},\ TPT_{1,1},\ XT_{c,1},\ YT_{c,1},\ TPT_{c,1},\ \ldots,\ XT_{c,a},\ YT_{c,a},\ TPT_{c,a},\ \ldots,\ XT_{ks,vc},\ YT_{ks,vc},\ TPT_{ks,vc}\}$; $\boldsymbol{AT1\_R}$ – множина ознак РП, які використовуються в першому раунді АК; $\boldsymbol{XT}$, $\boldsymbol{YT}$, $\boldsymbol{TPT}$ – вектори значень координат X і Y та типу точок зображення, відповідно; $\boldsymbol{XT} \subset \boldsymbol{AT1\_R}$; $\boldsymbol{YT} \subset \boldsymbol{AT1\_R}$; $\boldsymbol{TPT} \subset \boldsymbol{AT1\_R}$;

$\boldsymbol{AT2\_R}=\{\bigcup_{c=1}^{ks}\bigcup_{a2=1}^{5\cdot vc+5} ATX2\_R_{c,a2}\}=\{TPT_{1,1},\ PT_{1,1},\ UT_{1,1},\ TMT_{1,1},\ VT_{1,1},\ TPT_{c,1},\ PT_{c,1},\ UT_{c,1},\ TMT_{c,1},\ VT_{c,1},\ldots,\ TPT_{c,a},\ PT_{c,a},\ UT_{c,a},\ TMT_{c,a},\ VT_{c,a},\ldots,\ TPT_{ks,vc},\ PT_{ks,vc},\ UT_{ks,vc},\ TMT_{ks,vc},\ VT_{ks,vc},\ PL_c,\ KT_c,\ KU_c,\ CH_c,\ KP_c\}$; $\boldsymbol{AT2}$ – множина ознак РП, які використовуються в другому раунді АК; $\boldsymbol{PT}$– вектор значень тиску, з яким користувач натискає ручкою (чи іншим подібним пристроєм) на сенсорний екран під час створення точок; $\boldsymbol{UT}$

– вектор значень кута зміни напрямку написання при створенні точок; ***TMT*** – вектор значень часу, який пройшов від початку написання символу до створення конкретної точки; ***VT*** – вектор значень швидкості переміщення ручки від поперед.ньої точки в дану точку; ***PL*** – вектор значень площ зображень символів; ***KT*** – вектор значень кількостей точок символів, що аналізуються під час розпізнавання; ***KU*** – вектор значень кутів нахилу символів; ***CH*** – вектор значень частот точок символів, що зафіксовано системою; ***KP*** – вектор значень кількостей повторів точок в зображені символів (точок, що створені підряд, з однаковими обома координатами); $\boldsymbol{TPT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{PT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{UT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{TMT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{VT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{PL} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{KT} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{KU} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{CH} \subset \boldsymbol{AT2\_R}$; $\boldsymbol{KP} \subset \boldsymbol{AT2\_R}$;

$$\boldsymbol{O1\_R}_c=\{\bigcup_{s1=1}^{3\cdot w} OX1\_R_{s1}\}=\{XKT_1,\ YKT_1,\ TPKT_1,\ XTK_2,\ YTK_2,\ TPKT_2,\ \ldots,\ XKT_{c,g},\ YKT_{c,g},\ TPKT_{c,g},\ \ldots,\ XKT_{c,w},\ YKT_{c,w},\ TPKT_{c,w}\};$$

$\boldsymbol{O1\_R}_c$ – множина навчальних зразків написання *c*-го символу, з БДНЗ, який подається на ІНМ в першому раунді АК; $\boldsymbol{O1\_R}_c \subset \boldsymbol{AT1\_R}$;

$$\boldsymbol{O2\_R}_c=\{\bigcup_{s2=1}^{5\cdot w+5} OX2\_R_{s2}\}=\{TPKT_1,\ PKT_1,\ UKT_1,\ TMKT_1,\ VKT_1,\ TPKT_2,\ PKT_2,\ UKT_2,\ TMKT_2,\ VKT_2,\ldots,\ TPKT_{c,g},\ PKT_{c,g},\ UKT_{c,g},\ TMKT_{c,g},\ VKT_{c,g},\ldots,\ TPKT_{c,w},\ PKT_{c,w},\ UKT_{c,w},\ TMKT_{c,w},\ VKT_{c,w},\ PL_c,\ KT_c,\ KU_c,\ CH_c,\ KP_c\};$$

$\boldsymbol{O2\_R}_c$ – множина навчальних зразків стилю написання *c*-го символу, з БДНЗ, який подається на ІНМ в другому раунді АК; $\boldsymbol{O2\_R}_c \subset \boldsymbol{At2\_R}$;

$$\boldsymbol{ON1\_R}=\{\bigcup_{s1=1}^{3\cdot w} ONX1\_R_{s1}\}=\{XKT_1,\ YKT_1,\ TPKT_1,\ XTK_2,\ YTK_2,\ TPKT_2,\ \ldots,\ XKT_{c,g},\ YKT_{c,g},\ TPKT_{c,g},\ \ldots,\ XKT_{c,w},\ YKT_{c,w},\ TPKT_{c,w}\};$$

ON1_R – зразок РП, якій подається на ІНМ для розпізнавання в першому раунді АК; $\boldsymbol{ON1\_R} \subset \boldsymbol{At1\_R}$;

$$\boldsymbol{ON2\_R}=\{\bigcup_{s2=1}^{5\cdot w+5} ONX2\_R_{s2}\}=\{TPKT_1,\ PKT_1,\ UKT_1,\ TMKT_1,\ VKT_1,\ TPKT_2,\ PKT_2,\ UKT_2,\ TMKT_2,\ VKT_2,\ \ldots,\ TPKT_{c,g},\ PKT_{c,g},\ UKT_{c,g},\ TMKT_{c,g},\ VKT_{c,g},\ \ldots,\ TPKT_{c,w},\ PKT_{c,w},\ UKT_{c,w},\ TMKT_{c,w},\ VKT_{c,w},\ PL_c,\ KT_c,\ KU_c,\ CH_c,\ KP_c\};$$

ON2_R – зразок РП, якій

подається на ІНМ для розпізнавання в другому раунді АК; $\boldsymbol{ON2\_R} \subset \boldsymbol{At2\_R}$;

ED_T – це параметр, який означає максимальну довжину лінії (в точках), яку необхідно вважати випадковою;

$MAXXT_1$, $MAXXT_{ks}$, $MAXXT_c$, $MAXYT_1$, $MAXYT_{ks}$, $MAXYT_c$ – максимальні значення векторів значень ***XT*** та ***YT*** 1-го, *ks*-го та *c*-го символу КФ;

$MINXT_1$, $MINXT_{ks}$, $MINXT_c$, $MINYT_1$, $MINYT_{ks}$, $MINYT_c$ – мінімальні значення векторів значень ***XT*** та ***YT*** 1-го, *ks*-го та *c*-го символу КФ;

XMAX і *YMAX* – максимально можлива кількість точок робочої області обраного розміру за осями *X* та *Y*;

UG – кут, на який необхідно повернути зображення кожного символу, для нормалізації кута нахилу їх осей координат;

M_c – коефіцієнт по символьного масштабування с-го символу.

В дисертаційній роботі, для підвищення якості розпізнавання користувачів, запропоновано розділити процес АК на два раунди [116-123]:

1. розпізнавання КФ, що написана;

2. розпізнавання стилю написання КФ.

Відповідно функція *R* має вигляд:

$$R\_R = \begin{cases} 1, \text{ при } (US\_R_X \in \boldsymbol{USL\_R}) \wedge (US\_R_X = USL\_R_d); \\ 0, \text{ при } ((US\_R_X \in \boldsymbol{USL\_R}) \wedge (US\_R_X \neq USL\_R_d)) \vee (US\_R_X \in \boldsymbol{USB\_R}). \end{cases}$$

Якщо КФ є секретом, тобто імовірність її несанкціонованого отримання мінімальна, тоді для методу автентифікації достатнім є виконання тільки першого раунду розпізнавання.

В роботі розроблено метод реалізації першого раунду розпізнавання за РП та запропоновані рекомендації для реалізації другого раунду розпізнавання за РП. В даній роботі був розроблений метод ПОЗРП, необхідний для виконання АК ІС за їх РП [116-123].

3.2.Метод первинної обробки зразків рукописного почерку

В даному методі (метод ПОЗРП) виконується видалення або виправлення помилок в зразках РП користувачів ІС. Необхідність цієї обробки викликана специфікою використання ГП для АК за РП. Для РП властива деяка нестабільність. ІНМ досить

непогано справляється з цією проблемою, але якщо в зразку присутні дані, які є випадковими відхиленнями (помилками), які є несуттєвими для автентифікації і будуть тільки погіршувати якість розпізнавання, тоді їх необхідно або видалити зі зразку, або виправити, або видалити весь зразок з БДНЗ (в залежності від типу помилки). Проаналізувавши накопичені навчальні зразки (НЗ), в даній роботі було виділені вісім типів помилок, п'ять з яких видаляються, а три виправляються [116-123].

Метод ПОЗРП складається з наступних двох етапів, виконання яких необхідно на різних стадіях роботи методу АКРП [116-123]:

Етап 1.Попередній відбір даних, які будуть використовуватись для розпізнавання.

Етап 2.Корекція даних, які будуть використовуватись для розпізнавання зразків РП користувачів ІС.

Етап 1. Попередній відбір даних, які будуть використовуватись для розпізнавання. Цей відбір полягає у видаленні помилок перших п'яти типів [116-123].

В даній роботі видалення цих помилок (рис.6) виконується на різних стадіях виконання АК ІС за їх РП і складається з п'яти кроків.

Крок 1. Видалення помилок 1-го типу. Помилка 1-го типу – це послідовність точок з нульовим тиском (крім першої подібної точки з кожної послідовності). Виникають якщо користувач провів ручкою над ГП на невеликій відстані, не доторкнувшись до нього. Ці помилки усуваються за рахунок зберігання в зразку РП лише тих пакетів з нульовим тиском, які є першими в послідовності таких пакетів. Виконується на етапі формування БДНЗ або зразка, що розпізнається [116-123].

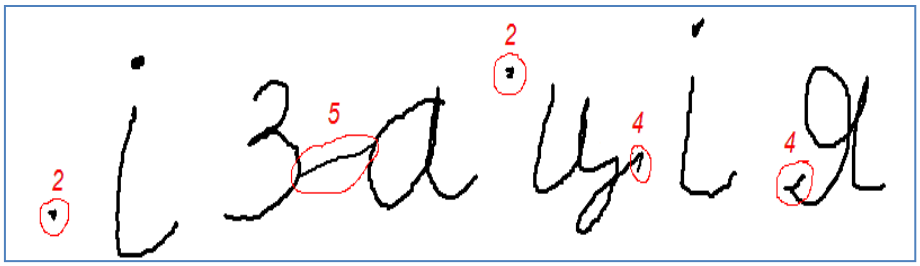


Рис.6. Приклади помилкових даних, які потребують видалення

Крок 2. Видалення помилок 2-го типу. Помилка 2-го типу – це випадкові точки (невелика кількість). Виникають якщо користувач випадково доторкнеться ручкою до ГП. Точки (в діапазоні від точки (*az*+1) по наступну точку з нульовим тиском) визнаються випадковими і видаляються із зразка, що аналізується, якщо виконується умова: $(PT_{az}=0)\wedge((PT_{az+1}=0)\vee(PT_{az+2}=0)\vee(PT_{az+3}=0)\vee\ldots\vee$

$\vee(PT_{az+ED\_T+1}=0))$, де ED_T налаштовується для кожної ІС і вказується в налаштуваннях системи [116-123].

Крок 3. Видалення помилок 3-го типу. Помилка 3-го типу – це повтори, тобто послідовність точок, що йдуть підряд, у яких значення координат по обом осям (X і Y) не змінилися (крім випадку, коли у одній з точок нульовий тиск). Виникають якщо пакет даних передався в комп'ютер в зв'язку з зміною не координат по якоїсь з осей (X і Y), а іншого параметру [116-123]. Дані по точці (az +1) повинні видалятися із зразка, що аналізується, якщо виконується наступна умова: $(XT_{az}=XT_{az+1}) \wedge (YT_{az}=YT_{az+1}) \wedge (PT_{az}\neq 0) \wedge (PT_{az+1}\neq 0)$.

Крок 4. Видалення помилок 4-го типу. Помилка 4-го типу – це випадкові невеликі загини (зазвичай, з гострим кутом) на початку ліній. Виникають або з вини інертності ГП, або через тремтіння руки користувача. Для визначення закінчення подібної помилки перевірявся напрям зміни координат ручки по осям X і Y і, якщо напрям хоча б по одній осі змінився – це означає, що помилка (загин) закінчився і далі вже йде сам символ. Відповідно, az -ая точка є останньою точкою загину, якщо виконується наступна умова: $(\text{sgn}(XT_{az+2}-XT_{az+1}) \neq \text{sgn}(XT_{az+1}-XT_{az}))\vee \vee(\text{sgn}(YT_{az+2}-YT_{az+1})\neq \text{sgn}(YT_{az+1}-YT_{az}))$. Довжину загину, який є помилкою (а не частиною символу, або відмінністю РП користувача), необхідно настроювати [116-123].

Крок 5. Видалення помилок 5-го типу. Помилка 5-го типу – неякісний зразок, який відкидається через неможливість розбивки зображення КФ на задану кількість зображень символів. Виникають якщо або вводиться невірна КФ, або якщо користувач має малий досвіт роботи з ГП, або якщо користувач які-небудь символи написав не окремо, а разом. При наявності такої помилки, зображення КФ не можна розділити на задану кількість зображень окремих символів, тому такі зразки вилучаються на етапі розділення зображення КФ на зображення окремих символів КФ [116-123].

Деякі з цих типів помилок є ними тільки в першому раунді АК, а в другому раунді є характерними особливостями стилю письма користувача. Наприклад, помилки 4-го типу можуть мути особливостями стилю письма, а помилка 5-го типу є помилками і в першому, і в другому раунді автентифікації.

Етап 2. Корекція даних, які будуть використовуватись для розпізнавання зразків РП користувачів ІС. Ця корекція полягає у виправленні помилок 6-8 типів, які викликані неправильним розміщенням написаної КФ на робочій області ГП .

На основі проведеного аналізу накопичених НЗ, було прийнято рішення о доцільності проведення посимвольної корекції даних в зразках РП, тобто виправлення помилок 6-8 типів (рис.7) [116-123]. Перед першим і другим раундами АК корекція різна. В першому раунді розпізнавання необхідність корекції обумовлена потребою привести зразки написання всіх символів під один шаблон. В другому раунді АК, з трьох типів корекції, рекомендується виконання корекції тільки 2-го типу. В даній роботі виконувались три типи корекції даних, під час яких вектори ***XT***, ***YT*** перетворюються в вектори ***XTP***, ***YTP***. Перетворення відбувається на трьох кроках.

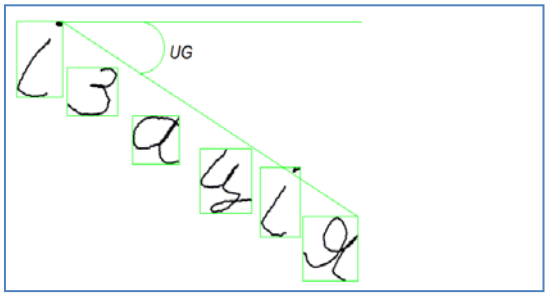


Рис.7. Приклад зображення до виконання корекції

Крок 1. Виправлення помилок 6-го типу. Помилка 6-го типу – це різний кут нахилу, відносно осей робочої області ГП, зображення КФ у різних користувачів. Для виправлення таких помилок виконується корекція 1-го типу – посимвольний поворот зображень символів для нормалізації кута нахилу їх осей координат (рис.8) [116-123].

Рис. 8. Результат посимвольного повороту зображення

Значення *UG* обчислюється за наступною формулою:

$$UG = arctg(\frac{MAXYT_{ks} - MAXYT_1}{MAXXT_{ks} - MAXXT_1}).$$

Поворот виконується за наступними формулами:

$$SX = \frac{MAXXT_c - MINXT_c}{2};\ SY = \frac{MAXYT_c - MINYT_c}{2};$$

$$XTP_{c,a} = round((XT_{c,a} - SX + MINXT_c)\cdot\cos(UG) + (YT_{c,a} - SY + MINYT_c)\cdot\sin(UG))$$
$$+round(SX + MINXT_c);$$

$$YTP_{c,a} = round((YT_{c,a} - SY + MINYT_c)\cdot\cos(UG) - (XT_{c,a} - SX + MINXT_c)\cdot\sin(UG))$$
$$+round(SY + MINYT_c).$$

Крок 2. Виправлення помилок 7-го типу. Помилка 7-го типу – це різне місце розташування, на робочій області ГП, зображення КФ у різних користувачів. Для виправлення таких помилок виконується корекція 2-го типу – посимвольний зсув зображень кожного символу в центр робочої області обраного розміру (рис.9) [116-123]. Зсув виконується за наступними формулами:

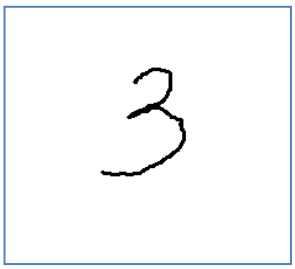


Рис. 9. Результат посимвольного зсуву зображення

$$XTP_{c,a} = XTP_{c,a} + round(\frac{XMAX - (MAXXT_c - MINXT_c)}{2}) - MINXT_c;$$

$$YTP_{c,a} = YTP_{c,a} + round(\frac{YMAX - (MAXYT_c - MINYT_c)}{2}) - MINYT_c.$$

Крок 3. Виправлення помилок 8-го типу. Помилка 8-го типу – це різний розмір зображення КФ, на робочій області ГП, у різних користувачів. Для виправлення таких помилок виконується корекція 3-го типу – посимвольне пропорційне масштабування (розтягування/стискання) зображень кожного символу на всю робочу область обраного розміру (рис. 10) [116-123].

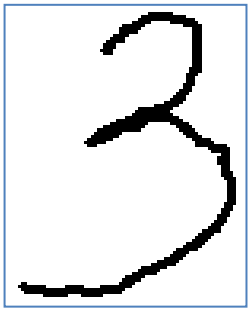


Рис. 10. Результат посимвольного пропорційного масштабування зображення

Коефіцієнт масштабування обраховувався за наступною формулою:

$$M_c = \min(\min(\frac{0 - (SX + MINXT_c)}{MINXT_c - (SX + MINXT_c)}, \frac{XMAX - (SX + MINXT_c)}{MAXXT_c - (SX + MINXT_c)}),$$
$$\min(\frac{0 - (SY + MINYT_c)}{MINYT_c - (SY + MINYT_c)}, \frac{YMAX - (SY + MINYT_c)}{MAXYT_c - (SY + MINYT_c)})).$$

Масштабування виконується за наступними формулами:

$$XTP_{c,a} = round((XTP_{c,a} - (SX + MINXT_c))\cdot M_c + (SX + MINXT_c));$$

$$YTP_{c,a} = round((YTP_{c,a} - (SY + MINYT_c))\cdot M_c + (SY + MINYT_c)).$$

3.3.Метод автентифікації користувачів інформаційних систем за їх рукописним почерком

Розроблений метод АКРП складається з десяти етапів [116-123]**.**

Етап 1. Попереднє формування множини ознак РП користувача ІС. Для АК ІС аналізується множина ознак його РП $\boldsymbol{AT\_R}=\{\bigcup_{i=1}^{n} At\_R_i\}$, яка описана в моделі даних. Ця множина складається з векторів ознак, значення яких передаються з ГП – це ***XT***, ***YT***, ***PT*** та з векторів ознак, які обраховуються на наступних етапах даного методу і є похідними від тих, що передані з ГП – це ***TPT***, ***UT***, ***TMT***, ***VT***, ***PL***, ***KT***, ***KU***, ***CH***, ***KP***. В кожному з перерахованих раундів АК, множина ознак ***AT_R***, яка використовується для розпізнавання зразків, різна. В першому раунді АК, для розпізнавання КФ в роботі використовується множина ***AT1_R***; $\boldsymbol{AT1\_R} \subset \boldsymbol{AT\_R}$.В другому раунді, для розпізнавання стилю написання КФ, використовується множина ***AT2_R***, в яку входять вектори ознак не тільки точок зображення, а й параметри загальні для усього символу; $\boldsymbol{AT2\_R} \subset \boldsymbol{AT\_R}$.В залежності від співвідношення надійності системи, що вимагається, до припустимого об’єму ресурсів, що задіяні, можна використовувати всі перераховані ознаки, або деякі з них [116-123].

Етап 2. Налаштування параметрів, які є найбільш критичними при АК ІС за їх РП. Налаштування найбільш критичних параметрів має дуже велике значення для збільшення ІПР користувачів ІС за їх РП. Визначення переліку цих параметрів здійснювалося за допомогою аналізу результатів проведених експериментів. Цими параметрами є: необхідна кількість символів КФ; мінімальна необхідна кількість навчальних зразків в БДНЗ для кожного користувача; помилки яких типів необхідно видаляти і при яких умовах (довжина послідовності точок, які повинні вважатися випадковими та довжина лінії до її загину, яка повинна вважатися помилкою); які типи корекції даних проводити для другого раунду АК (для першого корекція всіх трьох типів обов’язкова); з яких типів точок формувати множину КТ; довжина частини лінії, на якій треба обрати тільки одну КТ 2-го типу [116-123].

Етап 3. Формування БДНЗ користувачів ІС. В залежності від заданого режиму роботи, отриманий зразок РП користувача ІС, буде або зареєстрований в

БДНЗ, або відправлений на автентифікацію. Якщо в БДНЗ недостатньо НЗ для користувача, що розпізнається, тоді його не допускають до етапу автентифікації, а направляють на етап формування БДНЗ. Однією з особливостей принципу роботи ГП, яка враховувалася при розробці зазначеної системи АК, є той факт, що пакет даних з параметрами точки передається в комп'ютер, якщо змінилось значення хоча б одного параметру. Внаслідок цього в систему передаються зайві пакети даних, які необхідно видалити. Для видалення помилок одного з типів на даному етапі виконується Крок 1, Етапу 1, методу ПОЗРП. Крім параметрів точок зображення КФ, в даній роботі, в БДНЗ фіксуються "логін" користувача, дата та час створення точки. До користувачів висувається вимога, що символи КФ повинні бути написані не разом, а окремо один від одного [116-123].

Для отримання даних по точкам зображення з ГП, враховувались характеристики ГП, що використовувався для динамічної передачі зображення в комп'ютер; використовувався інтерфейс для роботи з ним (Wintab) та єдина модель біометричних технологій розпізнавання (BioAPI).

Етап 4. Видалення помилок 2,3,4 типів. Для виконання цієї задачі виконуються Крок 2 –Крок4, Етапу 1, методу ПОЗРП.

Етап 5. Умовне розділення зображення КФ на зображення окремих символів. Доцільність цього етапу аргументується наступними причинами [116-123]:

1. Легше накопичити БДНЗ для N символів, ніж для $\sum_{ks=1}^{mks} ks^N$ можливих КФ (або ks^N, якщо ks відоме), де N – кількість елементів множини можливих символів КФ; ks – довжина КФ; mks – максимально можлива довжина КФ.

2. Об'єкт, що класифікується, легше перевірити чи належність він до одного з N класів, ніж до одного з $\sum_{ks=1}^{mks} ks^N$ класів.

3. Значно зменшується об'єм ресурсів, що використовуються під час розпізнавання, за рахунок зменшення довжини зразка РП. Тобто, враховуючи те, що зображення одного символу в середньому складається з 100 точок, то якщо розділення не виконувати, тоді зразок РП буде складатися приблизно з $300 \cdot ks$ характеристик в пер-

шому раунді автентифікації та з 505·*ks* характеристик в другому раунді, а якщо розділення виконувати, тоді відповідно приблизно з 300 та 505 характеристик.

Розділення зображення КФ на зображення окремих символів виконувалось на основі аналізу значення тиску (PT_{az}) в пакетах даних о точках, що передаються з ГП, а саме – точка, в якої PT_{az} =0, використовувалась, як розмежував між зображеннями символів. Значення *ks* задається при налаштуванні системи АК ІС за їх РП [116-123].

Якщо після розділення, кількість отриманих зображень виявляється менша, ніж *ks*, тоді це помилка 5 типу і для її видалення виконується Крок 5, Етапу 1, методу ПОЗРП.

Якщо після розділення, кількість отриманих зображень виявляється більша, ніж *ks*, тобто деякі символи складаються з декількох частин, тоді ці частини з'єднувались, за рахунок знаходження сусідніх зображень, між границями яких мінімальна відстань.

Після розділення зображення КФ на зображення окремих символів КФ, з множини ***T*** буде сформована множина ***TS***.

Етап 6. Корекція даних, які будуть використовуватись для розпізнавання зразків РП користувачів ІС. Для виконання цієї задачі виконуються Етап 2, методу ПОЗРП.

Етап 7. Формування множини контрольних точок. Необхідність цього етапу пояснюється тим, що $c_v \approx 100$, відповідно множина ***AT1_R*** буде складатися з $3 \cdot ks \cdot c_v$ елементів (в нашому випадку ≈1800), а множина ***AT2_R*** відповідно з $5 \cdot ks \cdot c_v$ елементів (в нашому випадку ≈3000), у зв'язку з чим для виконання розпізнавання, витрачаються занадто великі ресурси. В зв'язку з цим в даній роботі аналізуються характеристики не всіх точок, а тільки найбільш значимих для кожного символу – контрольних точок (КТ). При цьому для кожного символу, з множини $\boldsymbol{TS}_c$ виділяється множина $\boldsymbol{KTS}_c$, яка і буде використовуватися для розпізнавання. Відповідно

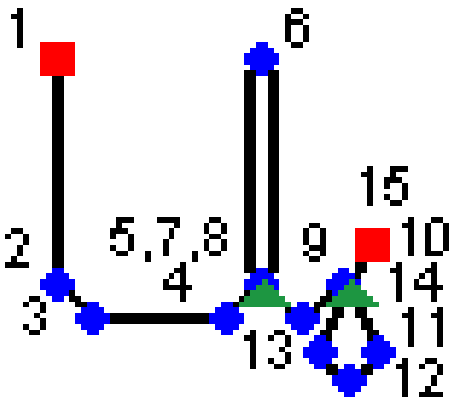


Рис. 11. Приклад розстановки КТ

XTP→XKT, ***YTP→YKT***, ***TPT→TPKT***, ***PT→PKT***, ***UT→UKT***, ***TMT→TMKT***, ***VT→VKT***. Від правильності розстановки КТ значно залежить ІПР об'єктів. В даній роботі виділяються наступні 3 типи КТ [116-123]:

1. Початкові та кінцеві точки кожної лінії – це точки торкання ручкою ГП та точки відриву ручки від ГП. На рис. 11 ці точки показані квадратами (точки 1 та 15) [116-123]. Для визначення таких точок використовуються збережені пакети даних с нульовим тиском ($PT_{c,a}$=0). Умова того, що *a*-я точка є початковою або кінцевою КТ: $((PT_{c,a} \neq 0) \wedge ((a = 1) \vee (a = vc) \vee (PT_{c,a-1} = 0) \vee (PT_{c,a+1} = 0)))$.

2. Кутові точки ліній – це точки, які знаходяться на вигині лінії. На рис. 11 ці точки показані колами (точки 2,3,4,5,6,7,9,10,11,12,13) [116-123]. Вигином лінії будемо називати зміну напряму лінії, яку можна визначити за зміною знака зміни координат по одній з осей (або по обом). Є декілька варіантів розміщення точок, при яких утворюються вигини ліній (рис. 12-14). Умова того, що *a*-я точка є кутовою КТ: $((a \neq 1) \wedge (a \neq vc) \wedge (PT_{c,a} \neq 0) \wedge (PT_{c,a-1} \neq 0) \wedge (PT_{c,a+1} \neq 0) \wedge (c_a = c_{a-1} = c_{a+1}) \wedge ((\text{sgn}(XT_{c,a} - XT_{c,a-1}) \neq \text{sgn}(XT_{c,a+1} - XT_{c,a})) \vee (\text{sgn}(YT_{c,a} - YT_{c,a-1}) \neq \text{sgn}(YT_{c,a+1} - -YT_{c,a}))))$; де sgn – математична функція, яка визначає знак виразу. При визначенні кутових КТ, за вказаним критерієм, виникає певна кількість хибних кутових КТ. Для усунення від процесу розпізнавання таких точок, в розробленій системі АК ІС за їх РП створена функція відбору найбільш значимих кутових КТ. На кожному відрізку лінії довжиною *D* (підбирається) визначається тільки одна, найбільш значима, кутова КТ, а інші відкидаються. При відборі аналізується траєкторія руху ручки по ГП під час переміщення між двома сусідніми кутовими точками зображення символу. Якщо

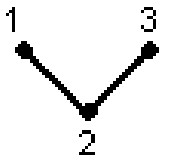


Рис.12. Кутова КТ першого типу

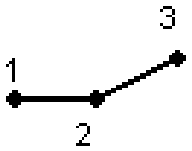


Рис.13. Кутова КТ другого типу

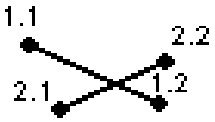
Рис.14.Кутова КТ третього типу

виконується наступна умова: $\sum_{cd=cdn}^{cdk} \sqrt{(XTP_{c,cd} - XTP_{c,cd+1})^2 + (YTP_{c,cd} - YTP_{c,cd+1})^2} < D$; (де *cdn* і *cdk* – номера, під якими зберігаються в множині $\boldsymbol{TS}_c$, відповідно дві точки, які згодом визначились як сусідні кутові КТ), тоді одну з двох точок необхідно видалити з множини КТ. В якості критерію відбору того, яку з двох кутових КТ треба залишити, в даній роботі використовується кут вигину лінії (*KUT*) в точці, – точка, в якій *KUT* менше, але не дорівнює нулю, повинна залишатися.

3. Точки перетинання ліній – це точки, які знаходяться на перетинанні ліній. На рис. 11 ці точки показані трикутниками (точки 8 та 14) [116-123]. Пошук ускладнює той факт, що точки зображення знаходяться не деякій відстані друг від друга (відстань між точками, зазвичай, більше 1 пікселю), тому в даній роботі були виділені три типи КТ перетинання (рис. 15-17). В першому випадку умовою того, що *a*-я точка є точкою перетинання ліній, є: $(TS_{c,j} \in \boldsymbol{TS}_c) \wedge (TS_{c,j} = \overline{TS_{c,1}, TS_{c,a-1}}) \wedge \quad (PT_{c,a} \neq 0)$ $\wedge (XTP_{c,j} = XTP_{c,a}) \wedge (YTP_{c,j} = YTP_{c,a}) \wedge \ (PT_{c,j} \neq 0)$. В другому випадку умовою того, що *a*-я точка є точкою перетинання ліній, є: $(TS_{c,j} \in \boldsymbol{TS}_c)$ $\wedge (TS_{c,j+1} \in \boldsymbol{TS}_c) \ \wedge (TS_{c,j} = \overline{TS_{c,1}, TS_{c,vc-1}}) \ \wedge (PT_{c,a} \neq$ $\neq 0) \wedge \quad (TS_{c,a} \in [TS_{c,j} TS_{c,j+1}]) \quad \wedge (PT_{c,j} \neq 0)$ $\wedge (PT_{c,j+1} \neq 0) \wedge (TS_{c,a} \neq TS_{c,j}) \wedge \ (TS_{c,a} \neq TS_{c,j+1})$. В третьому випадку умовою того, що *p*-я точка є точкою перетинання ліній, є: $(TS_{c,a} \in \boldsymbol{TS}_c) \wedge$ $\wedge (TS_{c,a+1} \in \boldsymbol{TS}_c) \quad \wedge (TS_{c,j} \in \boldsymbol{TS}_c) \wedge (TS_{c,j+1} \in$ $\in \boldsymbol{TS}_c) \wedge (TS_{c,j} = \overline{TS_{c,2}, TS_{c,a-2}}) \ \wedge (TS_{c,p} \notin \boldsymbol{TS}_c) \ \wedge (TS_{c,p} \in [TS_{c,a} TS_{c,a-1}]) \ \wedge (TS_{c,p} \in$ $\in [TS_{c,j} TS_{c,j-1}]) \ \wedge (PT_{c,a} \neq 0) \wedge (PT_{c,a-1} \neq \ 0) \wedge (PT_{c,j} \neq 0) \wedge (PT_{c,j-1} \neq 0) \wedge \ (c_j = c_{j-1} = c_a =$

Рис.15. КТ перетинання першого типу

б)
Рис.16. КТ перетинання другого типу



в)
Рис.17. КТ перетинання третього типу

$= c_{a-1}$). Для зменшення ресурсів, які використовувалися, не погіршив при цьому якість розпізнавання, в розробленому методі АК ІС за їх РП, забезпечувалось дотримання того, щоб одна і та сама точка не зберігалась більше ніж один раз в множині КТ. При правильному розміщенні КТ в кожному символі їх кількість буде приблизно однаковою, що є необхідною умовою для подальшого розпізнавання.

Етап 8. Виконання першого раунду АК ІС за їх РП, а саме розпізнавання КФ, що написана.

Процес розпізнавання в першому та другому раундах АК за їх РП різні. Задачу АК, тобто розпізнавання образів, в даному випадку, можна звести до задачі класифікації образів. В якості механізму класифікації образів, в даному випадку, була обрана ІНМ.

Як вже було сказано раніше, в першому раунді АК розпізнається кожний символ окремо і, відповідно, зразок написання кожного символу складається з множини ознак ***AT1_R*** для кожного символу з множини $\boldsymbol{KTS}_c$ Відповідно для розпізнавання на ІНМ подається зразок ***ON1_R***. Архітектура ІНМ обумовлена тим, що ця мережа повинна розпізнавати кожний написаний на ГП символ. Обрана мережа складається з наступних чотирьох шарів: вхідний шар, шар зразків, шар підсумовування, вихідний шар. Кількісно архітектуру мережі, виходячи з задачі, яка розв'язується, можна визначити наступним чином [116-123,126]:

1. Число вхідних елементів дорівнює числу ознак. В якості ознак, в даному випадку, використовуються по три параметра кожної зафіксованої КТ, відповідно, кількість вхідних елементів дорівнює $3 \cdot w$.

2. Число елементів шару зразків дорівнює числу НЗ. Кількість НЗ дорівнює сумі кількостей НЗ написання всіх символів (для яких є зразки написання в БДНЗ) з w-ою кількістю КТ (тому що для розпізнавання використовуються тільки ті НЗ написання символів, в яких зафіксована така ж кількість КТ, як і в зразку, що розпізнається). Відповідно, кількість елементів шару зразків дорівнює $\sum_{cu=1}^{ks} kol\_u_{cu}$, де kol_u – масив, елементи якого дорівнюють кількостям НЗ для кожного з ks символів.

3. Число елементів шару підсумовування дорівнює числу класів. При

розпізнавання написаних на ГП символів класом є символ з *w*-ою кількістю КТ, для яких є НЗ в БДНЗ. Відповідно, кількість класів дорівнює *ks*.

4. Вихідний шар завжди складається з одного елементу. Він визначає клас (в даному випадку символ), до якого з найбільшою імовірністю належить невідомий зразок.

Активність елементу шару зразків обраховується за наступною формулою:

$$AK1\_R_{nz} = \exp\left(\frac{\sum_{s1=1}^{3\cdot w} ONX1\_R_{s1} \cdot OX1\_R_{s1,nz} - 1}{\sigma^2}\right),$$

де σ – ширина функції активності.

Для використання цієї формули, вектор вхідних даних повинен бути нормалізовано.

Вихідний елемент являє собою дискримінатор граничної величини, який вказує елемент шару додавання з максимальним значенням, тобто вказує до якого класу належить невідомий екземпляр, або іншими словами, визначає який символ написано. Якщо цей символ правильним, тоді приймається рішення про успішне проходження першого раунду автентифікації [116-123].

Якщо всі символи КФ розпізнані правильно, тоді приймається рушення, що КФ написана правильна. Як аргументовано далі, в залежності від необхідного рівня безпеки, рішення про успішне розпізнавання можна приймати не обов’язково при вірному розпізнаванні всіх символів КФ, а наприклад, при розпізнаванні 5 з 6 символів. В даній роботі, у випадку коли правильно розпізнана вся КФ, приймалось рішення про успішне проходження АК ІС за їх РП [116-123].

Етап 9. Первинна обробка зразків РП користувачів ІС, за допомогою розробленого методу, яка необхідна для виконання другого раунду АК ІС за їх РП. Необхідні на цьому етапі дії описані в розробленому методі ПОЗРП.

Етап 10. Виконання другого раунду АК ІС за їх РП, а саме розпізнавання стилю написання КФ.

В другому раунді АК, в залежності від вимог до системи, в якості параметрів, що аналізуються, можна використовувати всі, або деякі ознаки множини ознак

AT2_R (наведених раніше параметрів КТ і параметрів окремих символів). Тобто для розпізнавання на ІНМ подається зразок ***ON2_R***. Кількісно архітектура мережі в другому раунді АК відрізняється від мережі в першому раунді і виглядає наступним чином [116-123]:

1. Число вхідних елементів дорівнює кількості ознак і відповідно дорівнює $5 \cdot w+5$.

2. Число елементів шару зразків дорівнює $\sum_{t=1}^{l} kol\_us_t$ (*kol_us* масив, елементи якого дорівнюють кількостям НЗ стилю написання КФ для l користувачів ІС).

3. Число елементів шару підсумовування дорівнює числу класів і відповідно дорівнює l.

4. Вихідний шар завжди складається з одного елементу. Він визначає клас (в даному випадку користувача), якому з найбільшою імовірністю належить невідомий зразок.

Активність елементу шару зразків обраховується за наступною формулою:

$$AK2\_R_{nz} = \exp\left(\frac{\sum_{s1=1}^{5 \cdot w+5} ONX2\_R_{s1} \cdot OX2\_R_{s1,nz} - 1}{\sigma^2} \right).$$

Для використання цієї формули, вектор вхідних даних повинен бути нормалізовано.

Вихідний елемент являє собою дискримінатор граничної величини, який вказує елемент шару додавання з максимальним значенням, тобто вказує до якого класу належить невідомий екземпляр, або іншими словами, визначає користувача, який намагається отримати доступ до ІС. Якщо це ім’я дорівнює імені, що пред’явлено під час АК, тоді система дозволяє доступ до ІС [116-123].

3.4.Алгоритмічна реалізація методу первинної обробки зразків рукописного почерку

Розглянемо деяки алгоритми, які були розроблені на їх основі запропонованих методів [116-123]. Для зручності написання вподальшому програмного забезпечення, на основі розроблених алгоритмів, позначення деяких характеристик в описі алгоритму і в описі методу відрізняються.

Алгоритми виключення із зразку деяких помилкових даних

Алгоритм УУЛ (алгоритм видалення ділянки лінії) має наступний вигляд:

1. Зменшити змінну *v* на кількість видаляємих елементів (*c*12-*c*10): *v*=*v*-*c*12+*c*10.

2. Видалити із безліча даних про передаваємі точки *Pac* аналізуємого зразку помилкові (зайві) точки. Для цього:

2.1.Записати в змінну *c*11 значення вираження (*c*10+1): *c*11= *c*10+1.

2.2.Якщо *c*11≤*v*, тоді перейти до пункту 2.3, інакше перейти до пункту 3.

2.3.Переписати із (*c*11+*c*12-*c*10)-го елемента безлічі *Pac* всі данні в *c*11-ий елемент безлічі *Pac*: $Pac_{c11} = Pac_{c11+c12-c10}$.

2.4.Збільшити змінну *c*11 на 1: *c*11= *c*11+1 і перейти до пункту 2.2.

3. Повернутися в алгоритм ИИОНХДНТ, до пункту, наступний за пунктом, який викликав алгоритм УУЛ.

Алгоритм ИИОНХДНТ (алгоритм виключення із зразку не характерних для нього точок) має наступний вигляд :

1. Перевірити чи встановлений в включений стан параметр, який вказує на необхідність виключити помилки третього типу (*chb_povt*=true). Якщо встановлено, то перейти до пункту 2, інакше перейти до пункту 3. Необхідно відмінити, що необовязково відкидувати помилки третього типу на даному етапі, це можна зробити при формуванні масиву контрольних точок чи при виборі найбільш значущих із них.

2. Проаналізувати чи є в оброблюваному зразку (в безлічі точок Т) помилки третього типу, тобто поспіль йдуть точки з однаковими обома координатами, але з різними будь-якими іншими параметрами точки і якщо такі повтори присутні в даному зразку, тоді із кожної серії повторів залишити по одній точці. Для цього необхідно виконати наступні дії:

2.1.Записати 1 в змінну *a*: *a*=1, де *a* – номер аналізуємої точки.

2.2.Якщо *a*<*v*, тоді перейти до пункту 2.3, інакше перейти до пункту 3.

2.3.Перевірити, чи має (*a*+1)-ая точка такі ж значення обох координат, як і *a*-ая точка і при цьому чи мають обидві ці точки не нульовий тиск,тобто перевірити на істинність умови. Якщо дані умови виконуються, тоді перейти до пункту 2.4, інакше перейти до пункту 2.8.

2.4.Записати в змінну *c*11 значення виразу (*a*+2): *c*11=*a*+2.

2.5.Перевірити, чи є в даном зразку *c*11-ая точка, чи має ця точка не нульовий тиск і чи має вона таке ж значення обох координат, як і *a*-ая точка тобто перевірити на істинність наступні умови:

$$\begin{cases} c11 \le v; \\ X_a = X_{c11}; \\ Y_a = Y_{c11}; \\ P_{c11} \ne 0. \end{cases}$$

Якщо дані умови виконуються , тоді збільшити змінну *c*11 на 1 (*c*11= *c*11+1) і перейти до пункту 2.5, інакше перейти до пункту 2.6.

2.6.Записати в змінну *c*12 значення виразу (*c*11-1): *c*12= *c*11-1.

2.7.Видалити із більшості даних про точки, які передаються *Pac* аналізуючого зразку точки, які є повторами *a*-ой точки. Для цього в змінну *c*10 записати значення змінної *a*: *c*10=*a*, після чого виконати алгоритм УУЛ.

2.8.Збільшити змінну *a* на 1 (*a*=*a*+1) і перейти до пункту 2.2.

3. Перевірити чи встановлено у включений стан параметр, який вказує на необхідність виключити помилки другого типу (*chb_t*=true). Якщо встановлено, то перейти до пункту 4, інакше перейти до пункту 5.

4. Проаналізувати чи є в оброблюваному зразку (в безлічі точок *T*) помилки другого типу, тобто лінія (одна чи декілька), яка складається із точок, чиє кількість не перевищує значення змінної ed_t і якщо такі є, тоді видалити їх. Для цього необхідно виконати настпуні дії:

4.1.Записати 1 в змінну *a*: *a*=1, де *a* – номер аналізуючої точки.

4.2.Якщо *a*<*v*, тоді перейти до пункту 4.3, інакше перейти до пункту 5.

4.3.Перевірити, чи має *a*-ая точка нульове значення тиску або ж є першою точкою зображення слова, тобто перевірити на істинність таких умов: якщо дана умова виконується, тобто (*a*+1)-ая точка буде першою точкой лінії чи першою точкой зображення слова, тоді перейти до пункту 4.4, інакше перейти до пункту 4.14.

4.4.Перевірити на істинність наступну умову: $P_a = 0$. Якщо умова істинна, тоді в змінну *c*10 записати значення змінної *a*: *c*10=*a*, інакше в змінну *c*10 записати вираз (*a*-1): *c*10=*a*-1. Потім незалежно від результату виконаної перевірки перейти до пункту 4.5

4.5.Записати в змінну *c*12 значення змінної *c*10: *c*12= *c*10.

4.6.Записати в змінну *c*11 значення виразу (*c*10+1): *c*11= *c*10+1.

4.7.Якщо *c*11≤(*c*10+1+*ed_t*), тоді перейти до пункту 4.8, інакше перейти до пункту 4.11.

4.8.Перевірити, чи є в даному зразку *c*11-ая точка, тобто перевірити на істинність наступну умову: $c11 \le v$. Якщо умова виконується, тоді перейти до пункту 4.9, інакше записати в змінну *c*12 значення виразу (*c*11-1): *c*12=*c*11-1 і перейти до пункту 4.11.

4.9.Перевірити, чи має *c*11-ая точка нульове значення тиску, тобто перевірити на істинність наступну умову: $P_{c11} = 0$. Якщо данна умова виконується, тобто (*c*11-1)-ая точка була останньою точкой лінії, тоді записати в змінну *c*12 значення змінної *c*11: *c*12=*c*11. Потім незалежно від результату виконаної перевірки перейти до пункту 4.10.

4.10. Збільшити змінну *c*11 на 1 (*c*11= *c*11+1) і перейти до пункту 4.7.

4.11. Перевірити чи була знайдена на проміжку між (c10 + 1) -ої і (c10 + 1 + ed_t) -ої точками (включаючи межі інтервалу) точка з нульовим тиском, тобто перевірити, чи закінчилася лінія, розпочата в (c10 + 1) -ої точці. Для цього перевірити на істинність таку умову: Якщо умова виконується, тоді перейти до пункту 4.12, інакше перейти до пункту 14.Видалити із безлічі даних про точки , які передаються *Pac* аналізуючого зразку серію випадкових точок, яка починається з (*c*10+1)-ой точки. Для цього виконати алгоритм УУЛ.

4.12. Перевірити чи є *c*10-ая точка не останньою точкою даного зразку, тобто перевірити на істинність наступну умову: *c*10<*v*. Якщо умова виконується, тоді перейти до пункту 4.5, інакше перейти до пункту 5.

4.13. Збільшити змінну *a* на 1 (*a*=*a*+1) і перейти до пункту 4.2.

5. Перевірити чи встановлений у ввімкненому стані параметр (chb_x), який вказує на необхідність виключати помилки четвертого типу и при цьому визначити чи більше він нуля, значення параметру (ed_kol_t), який вказує на максимальну

кількість точок в лінії, котра може визначатися помилкою четвертого типу, тобто перевірити на дійсність наступні умови:

$$\begin{cases} chb\_x = true; \\ ed\_kol\_t > 0. \end{cases}$$

Якщо умова виконується, тоді перейти до пункту 6, в іншому випадку перейти до пункту 7.

6. Проаналізувати чи є в оброблюваному зразку помилки четвертого типу, які відповідають відповідному критерію, тобто на початку ліній (однієї або декількох) випадкові загиби які складаються із точок, кількість яких не перевищує задане значення, тобто перевірити направлення зміни координат перших ed_kol_t точок кожної лінії, і якщо на цьому проміжку виявиться зміна направлення координат хоча б по однії із осей (*X* и *Y*), тоді зафіксувати помилку четвертого типу та видалити її. Для цього необхідно виконати наступні дії:

6.1.Записати 1 в змінну *a*: *a*=1, де *a* – номер аналізуючої точки.

6.2.Якщо *a*<(*v*-2), тоді перейти до пункту 6.3, інакше закінчити виконання даного алгоритму.

6.3.Перевірити, чи має точка *a* нульове значення тиску, тобто перевірити чи правдива наступна умова: $P_a = 0$. Якщо дана умова виконується, тобто (*a*+1)-а точка буде першою точкою лінії, тоді перейти до пункту 6.4, інакше перейти до пункту 6.12.

6.4.Записати в змінну *c*12 значення змінної *a*: *c*12=*a*.

6.5.Записати в змінну *c*11 значення виразу (*a*+1): *c*11=*a*+1.

6.6.Якщо $c11 \le (a+1+ed\_kol\_t)$, тоді перейти до пункту 6.7, інакше перейти до пункту 6.10.

6.7.Перевірити, чи є в даному зразку (*c*11+2)-а точка і при цьому чи мають точки *c*11-ая, (*c*11+1)-а і (*c*11+2)-а не нульове значення тиску, тобто перевірити на дійсність наступні умови:

$$\begin{cases} c11 \le (v-2); \\ P_{c11} \ne 0; \\ P_{c11+1} \ne 0; \\ P_{c11+2} \ne 0. \end{cases}$$

Якщо умови виконуються, тоді перейти до пункту 6.8, в іншому випадку перейти до пункту 6.10.

6.8.Перевірити чи змінюється направлення координат хоча б по одній з осей (*X* и *Y*) між точками (*c*11+2) і (*c*11+1) в порівнянні з напрямком зміни координат по відповідним осям між точками (*c*11+1) і *c*11, тобто перевірити на правдивість наступну умову (3.5), котра в даному випадку має такий вигляд:

$$\text{sgn}(X_{c11+2} - X_{c11+1}) \neq \text{sgn}(X_{c11+1} - X_{c11}) \text{ или } \text{sgn}(Y_{c11+2} - Y_{c11+1}) \neq \text{sgn}(Y_{c11+1} - Y_{c11}).$$

Якщо умова виконється, тоді записати змінну *c*12 значення змінної *c*11: *c*12=*c*11. Потім незалежно від результату виконаної перевірки перейти до пункту 6.9.

6.9.Збільшити змінну *c*11 на 1 (*c*11= *c*11+1) і перейти до пункту 6.6.

6.10. Перевірити чи була виявлена в проміжку між (*a*+1)-ою та (*a*+1+ed_kol_t)-ою точками (включаючи межі інтервалу) зміна направлення напрямку координат хоча б по одній із осей (*X* и *Y*), тобто випадковий загиб на початку лінії. Для цього перевірити на правдивість наспутні умови: $c12 \neq a$. Якщо умова виконується, тоді перейти до пункту 6.11, в іншому випадку до пункту 6.12.

6.11. Видалити більшість даних про передаваємі точки *Pac* аналізуючого зразку серію точок (яка починається з (*a*+1)-ї точки), які утворюють помилку четвертого типу. Для цього в змінну *c*10 записати значення змінної *a*: *c*10=*a*, після чого виконати алгоритм УУЛ.

6.12. Збільшити змінну *a* на 1 (*a*=*a*+1) і перейти до пункту 6.2.

7. Перевірити чи встановлено у включений стан параметр (chb_x), який вказує на необхідність виключити помилки четвертого типу і при цьому чи більше нуля, значення параметру (ed_x), який вказує на максимальну довжину частини ліній, яка може визнаватися помилкою четвертого типу, тобто перевірити на правдивість наступні умови:

$$\begin{cases} chb\_x = true; \\ ed\_x > 0. \end{cases}$$

Якщо умови виконуються, тоді перейти до пункту 8, інакше закінчити виконання даного алгоритму.

8. Проаналізувати чи є в оброблюваному зразку (в безлічі точок *T*) помилки четвертого типу, які відповідають вибраному критерію, тобто на початку ліній (однієї чи декількох) випадкові загиби, які не перевищують задану довжину, тобто перевірити напрямок зміни координат перших точок, які знаходяться на початку кожної лінії, на ділянці довжина, якого дорівнює значенню параметру ed_x, і якщо на цій ділянці виявиться зміна напрямку зміни координат хоча б по одній із осей (*X* и *Y*), тоді зафіксувати помилку четвертого типу, яку необхідно видалити. Для цього необхідно виконати наступні дії:

8.1.Записати 1 в змінну *a*: *a*=1, де *a* – номер аналізуючої точки.

8.2.Якщо *a*<(*v*-2), тоді перейти до пункту 8.3, інакше закінчити виконання даного алгоритму.

8.3.Перевірити, чи має *a*-а точка нульове значення тиску, тобто перевірити на правдивість наступну умову: $P_a = 0$. Якщо дана умова виконується, тобто (*a*+1)-а точка буде першою точкой лінії, тоді перейти до пункту 8.4, інакше перейти до пункту 8.14.

8.4.Записати в змінну *c*12 значення змінної *a*: *c*12=*a*.

8.5.Записати 0 в змінну *c_pr*: *c_pr* =0 (где *c_pr* – сума довжин відрізків , із яких складається досліджуєма ділянка лінії).

8.6.Записати в змінну *c*11 значення виразу (*a*+1): *c*11=*a*+1.

8.7.Перевірити, чи не перевищує значення змінної *c_pr* значення заданого параметру ed_x, тобто перевірити на правдивість наступну умову: $c\_pr \le ed\_x$. Якщо умова виконується , тоді перейти до пункту 8.8, інакше перейти до пункту 8.12.

8.8.Перевірити , чи є в даному зразку (*c*11+2)-а точка і при цьому чи мають точки *c*11-а, (*c*11+1)-а і (*c*11+2)-а не нульові значення тиску, тобто перевірити на правдивість наступні умови:

$$\begin{cases} c11 \le (v-2); \\ P_{c11} \ne 0; \\ P_{c11+1} \ne 0; \\ P_{c11+2} \ne 0. \end{cases}$$

Якщо умови виконуються, тоді перейти до пункту 8.9, інакше перейти до пункту 8.12.

8.9.Перевірити, чи змінюється напрямок зміни координат хоча б по одній із осей (*X* и *Y*) між точками (*c*11+2) і (*c*11+1) в порівнянні з напрямком зміни координат по відповідним осям між точками (*c*11+1) і *c*11, тобто перевірити на правдивість умови (3.5), яке в даном випадку приймає наступний вигляд:

$$\text{sgn}(X_{c11+2} - X_{c11+1}) \neq \text{sgn}(X_{c11+1} - X_{c11}) \text{ или } \text{sgn}(Y_{c11+2} - Y_{c11+1}) \neq \text{sgn}(Y_{c11+1} - Y_{c11}).$$

Якщо умова виконується, тоді записати в змінну *c*12 значення змінної *c*11: *c*12=*c*11. Потім незалежно від результату виконаної перевірки перейти до пункту 8.10.

8.10. Збільшити змінну *c_pr* на довжину відрізку між *c*11-о і (*c*11+1)-о точками, тобто виконати наступну дію:

$$c\_pr = c\_pr + \sqrt{(X_{c11+1} - X_{c11})^2 + (Y_{c11+1} - Y_{c11})^2}.$$

8.11. Збільшити змінну *c*11 на 1 (*c*11= *c*11+1) і перейти до пункту 8.7.

8.12. Перевірити чи була виявлена на ділянці лінії довжиною, зо дорівнює значенню змінної ed_x зміна напряму зміни координат хоча б по одній із осей (*X* и *Y*), тобто випадковий загиб на початку лінії. Для цього перевірити на правдивість наступну умову: $c12 \neq a$. Яущо умова виконується, тоді перейти до пункту 8.13, інакше перейти до пункту 8.14.

8.13. Видалити із безлічі даних про передаваємі точки *Pac* аналізуючого зразку серію точок (яка починається з (*a*+1)- точки), які утворюють помилку четвертого типу. Для цього в змінну *c*10 записати значення змінної *a*: *c*10=*a*, після чого виконати алгоритм УУЛ.

8.14. Збільшити змінну *a* на 1 (*a*=*a*+1) і перейти до пункту 8.2.

Алгоритми проведення необхідного корегування аналізуємих даних

Алгоритм ПТ (алгоритм пошуку точки) має наступний вигляд:

1. Перевірити чи є точка з номером *c*14 в даном зразку, тобто перевірити наступну умову: $c14 \leq v$. Якщо умова виконується, тоді перейти до пункту 2, інакше повернутися в алгоритм, викликавший алгоритм ПТ, в пункт, наступний за пунктом-викликом.

2. Перевірити наступну умову: $ID_{c14} \geq c19$, тобто порівняти номер c14-ої точки при створенні зразку зі значенням змінної c19. Якщо умова виконується, тоді повернутися в алгоритм, викликавший алгоритм ПТ, в пункт, наступний за пунктом-викликом, інакше до пункту 3.

3. Збільшити змінну c14 на 1 (c14= c14+1) і перейти до пункту 1.

Алгоритм ОГС (алгоритм визначення кордонів символів) має наступний вигляд:

1. Проініціалізувати змінні, в яких будуть зберігатися значення кордонів c-го символу,тобто записати в ці змінні відповідні координати X і Y першої точки c-го символу з ненульовим тиском. Першою точкою з ненульовим тиском може бути не обовязково перша точка зображення, але і друга чи, через різні збої, інша наступна точка. Випадок, коли такої точки взагалі не має, в даном алгоритмі не розглядається, тому що розроблена система подібні зразки взагалі не фіксує. Вказані факти пояснюються особливостями стилю написання різних людей і специфікою работи графічних планшетів. В результаті, для ініціалізування вказаних змінних треба:

1.1.Записати в змінну c19 значення змінної c13 (її значення передається із алгоритму КД): c19=c13, де c19 – номер аналізуємої точки під яким вона була збережена в зразку.

1.2.Перевірити чи належить c19-ая точка c-му символу. Для цього перевірити чи менше значення змінної c19 номера точки с нульовим тиском (TND_c), яка відокремлює зображення c-го символу від (c+1)-го символа, тобто перевірити наступну умову: $c19 \leq TND_c$. Якщо умова виконується, тоді перейти до пункту 1.3, інакше перейти до пункту 2.

1.3.Знайти c19-ую точку в масиві (безлічі) точок T. Для цього виконати алгоритм ПТ.

1.4.Перевірити, не рівняється чи нулю, тиск в c14-ой точці, тобто перевірити наступну умову: $P_{c14} \neq 0$. Якщо умова виконується, тоді спочатку в змінні $MinX_c$, $MaxX_c$, $MinY_c$, $MaxY_c$, записати відповідні координати X і Y c14-ої точки c-го символу, тобто виконати наступні присвоєння:

$$MinX_c = X_{c14}; MaxX_c = X_{c14}; MinY_c = Y_{c14}; MaxY_c = Y_{c14}; c15 = c14;$$

потім перейти до пункту 2. Якщо вказана умова не виконується, тоді перейти до пункту 1.5.

1.5.Збільшити змінну $c19$ на 1 ($c19= c19+1$) і перейти до пункту 1.2.

2. Визначити кордони c-го символу, тобто створити масиви, в яких для c-го символу визначити значення $MinX$, $MaxX$, $MinY$, $MaxY$. Для цього необхідно зробити наступне:

2.1.Записати в змінну $c19$ значення змінної $c13$: $c19=c13$, де $c19$ – номер аналізуючої точки під яким вона була збережена в зразку.

2.2.Перевірити чи належить $c19$-ая точка c-му символу. Для цього перевірити чи менше значення змінної $c19$ номера точки з нульовим тиском (TND_c), яка відділяє зображення c-го символу від (c+1)-го символу, тобто перевірити наступну умову: $c19 \le TND_c$. Якщо умова виконується, тоді перейти до пункту 2.3, інакше перейти до пункту 3.

2.3.Знайти $c19$-ую точку в масиві (безлічі) точок T. Для цього виконати алгоритм ПТ.

2.4.Перевірити, чи не рівняється нулю тиск в $c14$-ой точці, тобто перевірити наступну умову: $P_{c14} \neq 0$. Якщо умова виконується, тоді виконати наступне:

якщо $MinX_c > X_{c14}, mo \;\; MinX_c = X_{c14}$; якщо $MaxX_c < X_{c14}, mo\; MaxX_c = X_{c14}$;

якщо $MinY_c > Y_{c14}, mo\; MinY_c = Y_{c14}$; якщо $MaxY_c < Y_{c14}, mo\; MaxY_c = Y_{c14}$.

Потім незалежно від результату перевірки умов перейти до пункту 2.5.

2.5.Збільшити змінну $c19$ на 1 ($c19= c19+1$) і перейти до пункту 2.2.

3. Повернутися в алгоритм КД, в пункт, наступний за пунктом, викликавшим алгоритм ОГС.

Алгоритм КД (алгоритм корегування даних) має вигляд:

1. Перевірити чи встановлено у включенний стан параметр (*chb_pov*=true), який вказує на необхідність виконання корекції першого типу (ця функція необхідна на першому етапі автентифікації, аргументація необхідності викладена раніше в цьому ж підрозділі). Якщо встановлено, тоді перейти до пункту 2, інакше перейти до пункту 3.

2. Виконати корекцію першого типу. Для цього виконати наступне:

2.1.Записати в змінну $c13$ номер першої точки, під яким вона була збережена в зразку: $c13=ID_1$.

2.2.Записати 1 в змінну $c14$: $c14=1$.

2.3.Записати 1 в змінну c: $c=1$, де c – номер аналізуючого символу.

2.4.Перевірити, чи є в данному зразку c-ий символ, тобто перевірити наступну умову: $c \le sv$. Якщо умова виконується, тоді перейти до пункту 2.5, інакше перейти до пункту 2.8.

2.5.Виділити кордони c-ого символу. Для цього виконати алгоритм ОГС.

2.6.Записати в змінну $c13$ номер точки з нульовим тиском (TND_c), яка віддаляє зображення c-го символу від (c+1)-го символу, тобто виконати наступне присвоєння: $c13 = TND_c$.

2.7.Збільшити змінну c на 1 ($c= c+1$) і перейти до пункту 2.4.

2.8.Визначити (ug) як правильно повернути зображення символів слова – пароля, щоб нормалізувати нахил їх осей координат.

2.9.Записати в змінну $c19$ номер першої точки, під яким вона була збережена в зразку: $c19=ID_1$.

2.10. Записати 1 в змінну $c14$: $c14=1$.

2.11. Записать 1 в змінну c: $c=1$, де c – номер аналізуючого символу.

2.12. Перевірити, чи є в даному зразку *c-й* символ, тобто перевірити наступну умову: $c \le sv$. Якщо умова виконується, тоді перейти до пункту 2.13, інакше перейти до пункту 3.

2.13. Записати в змінну $c14$ значення змінної $c15$: $c14 = c15$.

2.14. Перевірити чи належить $c19$-а точка c-му символу. Для цього перевірити меньше значення змінної $c19$ номера точки с нульовим тиском (TND_c), котра розмежовує зображення c-го символа від зображення (c+1)-го символа, тобто перевірити наступну умову: $c19 \le TND_c$. Якщо умова виконується, тоді перейти до пункту 2.15, або перейти до пункту 2.20.

2.15. Знайти $c19$-у точку в кількості точок T. Для цього виконати алгоритм ПТ.

2.16. Записати в буферну змінну $c16$ значення X_{c14} ($c16=X_{c14}$), а в буферну

змінну $c17$ значення Y_{c14} ($c17=Y_{c14}$). Ці дії необхідні, тому що в формулах обчислення нових значень координат кожної точки задіяні старі значення по обох осях координат, а при обчисленні значення по другій осі, по першій осі вже нові значення. Тому старі значення необхідно зберегти в буферних змінних.

2.17. Обчислити координати *c14-oї* точки повернутого зображення с-го символу за формулами, які з урахуванням використання буферних змінних *c16 і c17*, а також того, що нові значення записуються на місце старих і того, що в якості індексу (у координат точок) використовується не номер, під яким точка була збережена в зразку, а номер по порядку, приймуть такий вигляд:

$$X_{c14} = round((c16 - \frac{MaxX_c - MinX_c}{2} + MinX_c) * \cos(\frac{ug * \pi}{180}) +$$

$$+ (c17 - \frac{MaxY_c - MinY_c}{2} + MinY_c) * \sin(\frac{ug * \pi}{180})) + round(\frac{MaxX_c - MinX_c}{2} + MinX_c);$$

$$Y_{c14} = round((c17 - \frac{MaxY_c - MinY_c}{2} + MinY_c) * \cos(\frac{ug * \pi}{180}) -$$

$$- (c16 - \frac{MaxX_c - MinX_c}{2} + MinX_c) * \sin(\frac{ug * \pi}{180})) + round(\frac{MaxY_c - MinY_c}{2} + MinY_c).$$

2.18. Перевірити наступну умову: $ID_{c14} \neq c19$. Якщо дія виконується, тоді необхідно виконати наступну умову: $c19 = ID_{c14} + 1$, інакше виконати наступну дію: $c19 = c19 + 1$.

2.19. Перейти до пункту 2.14.

2.20. Записати в змінну с19 номер точки з нульовим тиском (TNDc), яка відокремлює зображення с-го символу від *(с + 1)-го* символу, тобто виконати наступне привласнення:

2.21. Збільшити змінну *с* на *1* (с= с+1) і перейти до пункту 2.12.

3. Перевірити чи встановлений у включений стан параметр *(chb_sdv = true)*, який вказує на необхідність виконання корекції другого типу (ця функція необхідна на першому етапі автентифікації і можлива на другому етапі, аргументація необхідності викладена раніше в цьому ж підрозділі). Якщо встановлено, тоді перейти до пункту 4, інакше перейти до пункту 5.

4. Виконати корекцію другого типу. Для цього виконати наступне:

4.1 Записати в змінну c13 номер першої точки, під яким вона була збережена в зразку: c13 = ID1.

4.2 Записати 1 в змінну c14: c14 = 1.

4.3 Записати 1 в змінну c: c = 1, де з - номер аналізуючого символу.

4.4 Перевірити, чи є в даному зразку з-ий символ, тобто перевірити наступне умова: Якщо умова виконується, тоді перейти до пункту 4.5, інакше перейти до пункту 5.

4.5 Визначити кордону з-ого символу. Для цього виконати алгоритм ОГС.

4.6 Визначити для кожного символу зміщення по координатним осях X і Y (SDXc, SDYc), на яке необхідно зрушити його зображення, щоб зображення кожного символу розташовувалося в центрі робочої області обраного розміру.

4.7 Записати в змінну c19 значення змінної c13: c19 = c13, де c19 - номер аналізуючої точки під яким вона була збережена в зразку.

4.8 Перевірити чи належить c19-ая точка c-му символу. Для цього перевірити менше значення змінної c19 номера точки з нульовим тиском (TNDc), яка відокремлює зображення c-го символу від (c + 1) -го символу, тобто перевірити наступне умова: Якщо умова виконується, тоді перейти до пункту 4.9, інакше перейти до пункту 4.14.

4.9 Записати в змінну c14 значення змінної c15: c14 = c15.

4.10 Знайти c19-ую точку в масиві (безлічі) точок T. Для цього виконати алгоритм ПВ.

4.11 Обчислити координати c14-ої точки зрушено зображення c-го символу, за формулами: $X_{c14} = X_{c14} + SDX_c$; $Y_{c14} = Y_{c14} + SDY_c$;

4.1.Перевірити наступну умову: $ID_{c14} \neq c19$. Якщо умова виконується, тоді виконати наступну дію: $c19 = ID_{c14} + 1$, інакше виконати наступну дію: $c19 = c19 + 1$.

4.2.Перейти до пункту 4.8.

4.3.Записати в змінну $c13$ номер точки з нульовим тиском (TND_c), яка відділяє зображення c-го символу від (c+1)-го символу, тобто виконати наступне присвоєння: $c13 = TND_c$.

4.4.Збільшити змінну c на 1 (c=c+1) і перейти до пункту 4.4.

5. Перевірити чи встановлений у включенні стан параметр (*chb_m*=true), який вказує на необхідність виконання корекції третього типу (ця функція необхідна на першому етапі автентифікації, аргументація необхідності викладена раніше в цьому ж підрозділі). Якщо встановлена, тоді перейти до пункту 6, інакше закінчити виконання даного алгоритму.

6. Виконати корекцію другого типу. Для цього виконати наступне:

6.1.Записати в змінну $c13$ номер першої точки, під яким вона була збережена в зразку: $c13=ID_1$.

6.2.Записати 1 в змінну $c14$: $c14=1$.

6.3.Записати 1 в змінну c: $c=1$, де c – номер аналізуючого символу.

6.4.Перевірити, чи є в даному зразку c-ий символ, тобто перевірити наступну умову: $c \le sv$. Якщо умова виконується, тоді перейти до пункту 6.5, інакше закінчити виконання даного алгоритму.

6.5.Визначити кордони c-ого символу. Для цього виконати алгоритм ОГС.

6.6.Визначити для кожного символу масштаб (M_c), тобто визначити у скільки разів необхідно збільшити чи зменшити зображення кожного символу, щоб кожне з них було розтягнуто на всю робочу область вибраного розміру (принцип визначення викладено раніше в цьому ж підрозділі).

6.7.Записати в змінну $c19$ значення змінної $c13$: $c19=c13$, де $c19$ – номер аналізуючої точки під яким вона була збережена в зразку.

6.8.Перевірити чи належить $c19$-а точка c-му символу. Для цього перевірити и менше значення змінної $c19$ номеру точки з нульовим тиском (TND_c), яка відділяє зображення c-го символу від (c+1)-го символу, тобто перевірити наступну умову: $c19 \le TND_c$. Якщо умова виконується , тоді перейти до пункту 6.9, інакше перейти до пункту 6.14.

6.9.Записати в змінну $c14$ значення змінної $c15$: $c14=c15$.

6.10.Знайти $c19$-у точку в масиві (безлічі) точок T. Для цього виконати алгоритм ПТ.

6.11.Обчислити координати $c14$-ої точки розтягнутого зображення c-го симво-

лу, тобто порахувати формули, які з урахуванням того, що в якості індексу (у координат точок) використовується не номер, під яким точка була збережена в зразку, а номер по порядку на даний момент, приймуть наступний вигляд:

$$X_{c14} = round((X_{c14} - (\frac{(MaxX_c\text{-}MinX_c)}{2} + MinX_c)) * M_c + (\frac{(MaxX_c\text{-}MinX_c)}{2} + MinX_c));$$

$$Y_{c14} = round((Y_{c14} - (\frac{(MaxY_c\text{-}MinY_c)}{2} + MinY_c)) * M_c + (\frac{(MaxY_c\text{-}MinY_c)}{2} + MinY_c)).$$

6.12. Перевірити наступну умову: $ID_{c14} \neq c19$. Якщо умова виконується, тоді виконати наступну дію: $c19 = ID_{c14} + 1$, інакшу виконати наступну дію: $c19 = c19 + 1$.

6.13. Перейти до пункту 6.8.

6.14. Записати в змінну $c13$ номер точки з нульовим тиском (TND_c), яка відділяє зображення c-го символу від (c+1)-го символу, тобто виконати наступне присвоєння: $c13 = TND_c$.

6.15. Збільшити змінну c на 1 (c= c+1) і перейти до пункту 6.4.

3.5. Висновки до третього розділу

В даному розділі запропоновано метод первинної обробки зразків РП, в якому за рахунок автоматизації процесу відбору КТ в зразках РП, чиї характеристики аналізуються, видаленню помилкових точок п’яти типів та проведенню корекції даних трьох типів, досягається збільшення ІПР користувачів ІС на 35-40% та, завдяки зменшенню кількості ознак в зразках, що аналізуються, зменшення затрачуваних ресурсів. Після чого вдосконалено метод АК ІС за їх РП, який за рахунок використання для розпізнавання ІНМ та виконання первинної обробки зразків РП, збільшує ІПР користувачів ІС, в порівнянні з еталонним методом (метод ВБРНМ), на 7-8%. Розроблений метод АК ІС за їх РП має сенс використовувати не тільки, як самостійний метод автентифікації, а й як один з етапів багатофакторної автентифікації. Так як і метод АК за їх КП, метод АК за їх РП можна використовувати не тільки для АК ІС, а й для аналізу поточного стану таких характеристик людини, як уважність, сконцентрованість, акуратність, під час прийому на роботу.

РОЗДІЛ 4. ТЕСТУВАННЯ РОЗРОБЛЕНИХ МЕТОДІВ ТА ВИЗНАЧЕННЯ НАЙБІЛЬШ КРИТИЧНИХ ПАРАМЕТРІВ СИСТЕМ РОЗПІЗНАВАННЯ

4.1.Тестування методу автентифікації користувачів інформаційних за їх клавіатурним почерком

Опис експериментів

Для визначення впливу всіх параметрів був накопичений за алгоритмом ННЗЗЗЗП необхідний обсяг інформації, після чого було проведено ряд експериментів для визначення ефективності застосування даного виду нейронних мереж для вирішення задачі АК ІС, або іншими словами, для визначення ІПР користувачів системи [99-116]. Як вже було сказано, для цього використовувалася спеціально створена програма ІНМАККП. Ця програма (автоматизована система) написана мовою C++ Builder [199]. Для обробки та зберігання даних використовувалася програма для роботи з базами даних Database Desktop та SQL-запити. На основі результатів першого етапу експериментів та візуального аналізу накопиченої інформації про характер роботи користувачів на клавіатурі були також виділені інші фактори, що впливають на ІПР. Їх вплив був досліджений під час наступних етапів проведення експериментів. Всі експерименти можна розділити на три етапи:

1. На першому етапі аналізувалися дані для всіх користувачів, при цьому розглядалися всі навчальні зразки, тобто час набору символів кожного слова. На цьому етапі визначався вплив наступних параметрів на ІПР користувачів:

- Наявність усереднення навчальних даних (*usr*), при якому усереднювалися ознаки із кожних десяти записів. При цьому збільшувалася точність даних, але зменшувалася їх кількість.
- Наявність відбору за словами (*ots*), тобто «невідомі» екземпляри порівнювалися з усіма навчальними даними або тільки з такими ж словами. Якщо відбір не проводився, тоді неточною була тільки перша ознака, а якщо проводився відбір, цих неточностей не було, але кількість навчальних даних зменшувалося в 10 разів, тому що в наборі було 10 слів.
- Кількість ознак (n).
- Кількість навчальних даних для кожного користувача (z_t).

♦ Ширина функції активності (σ).

2. На другому етапі досліджувався вплив на ефективність застосування імовірнісної нейтронної мережі для рішення поставленої задачі таких параметрів, як відсоток помилок при наборі тексту (*Och*) (уважність користувачів при роботі за клавіатурою), амплітуда (A_{Och}) відсотку помилок у користувачів між собою. Крім того, визначався вплив цих факторів на залежність ефективності застосування імовірнісної нейтронної мережі для рішення задачі АК ІС від розглянутих раніше параметрів.

3. На третьому етапі досліджувався вплив на ефективність застосування імовірнісної нейтронної мережі для рішення поставленої задачі такого параметра, як кількість грубих помилок у значеннях ознак в навчальних зразках. Також досліджувалося, який з розглянутих алгоритмів виключення навчальних зразків із грубими помилками кращий. Після вибору кращого алгоритму, визначався максимально допустимий відсоток відхилення ($\left|u_{ijt} - Sr_{it}\right|$) кожної ознаки від її середнього значення (*Sr*). Крім того, визначався вплив цього фактора на залежність ефективності використання імовірнісної нейтронної мережі для рішення задачі АК ІС від розглянутих раніше параметрів.

Результати експериментів першого етапу:

На першому етапі були зроблені наступні припущення [99-116]:

♦ Проведення відбору за словами (*ots*=істина) за рахунок зменшення при цьому кількості навчальних зразків не підвищить ІПР, а, можливо, і зменшить її.

♦ Проведення усереднення навчальних даних (*usr*=істина) за рахунок зменшення при цьому кількості навчальних зразків не підвищить імовірність правильного розпізнавання, а, можливо, і зменшить її.

♦ Чим більше використовується навчальних зразків для розпізнавання, тим вище буде ІПР користувачів ІС.

♦ Чим більше аналізується ознак, тим вище імовірність правильної автентифікації користувачів.

♦ Вдалий підбір ширини функції активності (σ) підвищує якість розпізнавання.

Для перевірки цих припущень було проведено ряд експериментів.

Спочатку досліджувався вплив виконання відбору за словами на якість розпізнавання. Для цього проводилося дві групи експериментів, у яких загальними були наступні умови: навчальних зразків для кожного користувача (z_t) – 1000, ознак (n) – 6, усереднення навчальних зразків (usr) – є. Але в першій групі експериментів відбір за словами не проводився (ots=неправда), а в другій – проводився (ots=істина). При цьому виходить, що кількість навчальних зразків (z_t) зменшується в 10 разів, тому що в наборі 10 слів.

З огляду на великий обсяг інформації, що аналізується, на цьому етапі проведення експериментів, кожен експеримент займав значний проміжок часу, тому експерименти проводилися не для всіх значень аналізованих параметрів, а тільки для декількох значень для кожного параметра. Тому для побудови графіків з необхідною повнотою даних не досить, але, проаналізувавши отримані результати цих двох груп експериментів, можна зробити наступний висновок:

При проведенні відбору за словами (ots=істина) імовірність правильної відповіді приблизно на 2% нижче, ніж у першому, звідси випливає, що краще більше навчальних даних, чим відбір за словами, тому що без відбору за словами «страждає» тільки перша ознака.

Потім досліджувався вплив усереднення навчальних зразків по 10 (usr=істина, K_{usr}=10) на якість розпізнавання користувачів. Для цього проводилося дві групи експериментів, у яких спільними були наступні умови: навчальних зразків для кожного користувача (z_t) – 1000, ознак (n) – 6, відбір за словами не проводився (ots=неправда). Але в першій групі експериментів усереднення відбувається (usr=істина, K_{usr}=10), а в другій – не проводиться (usr=неправда). При цьому виходить, що в результаті в першому випадку кількість навчальних зразків (z_t) у 10 разів менша, ніж у другому.

Проаналізувавши отримані результати експериментів, можна зробити наступний висновок:

У випадках, коли усереднення не виконується, ІПР збільшувалася приблизно на 3-4%, що говорить про те, що кількість навчальних даних має більший вплив на результат, ніж невелике збільшення точності навчальних даних.

У наступних групах експериментів досліджувався вплив кількості навчальних

зразків (z_t) на ІПР користувачів ІС. Для цього було проведено три групи експериментів, у яких загальними були наступні умови: ознак (n) – 6, відбір за словами не проводився (ots=неправда), усереднення навчальних зразків не проводилося (usr=неправда). Але в першій групі експериментів навчальних зразків для кожного користувача було 1000 (z_t=1000), у другій – z_t=500, а в третій – z_t=1500.

Проаналізувавши отримані результати експериментів, можна зробити наступний висновок:

При збільшенні навчальних зразків (z_t) з 1000 до 1500 ІПР збільшувалася на 2%, а при зменшенні з 1000 до 500 ІПР знижувалася на 2%.

Наступною дією досліджувався вплив кількості аналізованих ознак (n) на якість розпізнавання. Для цього проводилися три групи експериментів, у яких спільними були наступні умови: відбір за словами не проводився (ots=неправда), усереднення навчальних зразків не проводилося (usr=неправда), навчальних зразків для кожного користувача було 1000 (z_t=1000). Але в першій групі експериментів кількість ознак дорівнювала 6 (n=6), у другий – n=3 та аналізувалися часові інтервали між натисканням останніх трьох символів, у третій – n=3, але аналізувалися часові інтервали між натисканням передостанніх трьох (4,5,6) символів.

Проаналізувавши отримані результати експериментів можна зробити наступний висновок:

Зміна кількості ознак (n) від 3 до 6 збільшувала імовірність правильної відповіді приблизно на 10%, причому, чим більше користувачів у системі, тим більше поліпшення результату.

Потім у наступних експериментах досліджувався вплив значення ширини функції (σ) на ІПР користувачів ІС за КП. В проведених експериментах значення σ змінювалося від 0.01 до 1 (включно) з кроком 0.01.

Проаналізувавши отримані в цій групі експериментів результати можна зробити наступний висновок:

Вдала зміна ширини функції активності (σ) збільшувала імовірність правильної відповіді тільки приблизно на 2%.

Підсумувавши результати проведених на цьому етапі експериментів можна

зробити наступні висновки:

1. ІНМ можна використовувати для рішення задачі автентифікації у випадках, коли з ІС працює обмежена кількість користувачів.

2. ІПР користувачів залежить від кількості користувачів (l) системи та від налаштовування найбільш критичних параметрів.

3. З приводу критичних параметрів можна зробити наступні висновки:

♦ Для ІПР користувачів кількість навчальних даних (z_t) важливіша, ніж їх первинна обробка, тобто краще більше навчальних даних нехай навіть з невеликою похибкою, ніж більш точні дані, але в меншій кількості.

♦ Ширина функції активності (σ) слабо впливає на якість автентифікації.

♦ Збільшення кількості аналізованих ознак (n) досить сильно впливає на достовірність автентифікації.

♦ Найбільш впливає кількість помилок (Och) при наборі тексту та, при цьому, чим менше амплітуда (A_{Och}) відсотка помилок у користувачів між собою, тим більше ІПР користувачів. Тобто чим більш уважніший користувач при наборі тексту, тим більша ІПР користувачів, крім того, бажано, щоб рівень уважності користувачів був приблизно однаковим.

Результати експериментів другого етапу:

Однією з характеристик роботи користувачів на клавіатурі – є відсоток помилок (Och), тобто відсоток натискання помилкових клавіш. Провівши візуальний аналіз інформації, яка є, про КП користувачів ІС, можна зробити висновок про те, що в різних користувачів значення цього параметра може різко відрізнятися. Звичайно краще, щоб помилок було менше, але при цьому виникає питання – який відсоток помилок можна вважати допустимим, а при якому відсотку помилок користувача не можна допускати до роботи. Крім того, виникає питання, про те який розкид відсотка помилок у користувачів між собою допустимий (A_{och}) [99-116].

Можна зробити наступні припущення:

♦ Чим менший відсоток помилок (Och), тим більша ІПР користувачів.

♦ Чим менший розкид (амплітуда A_{Och}) відсотка помилок у користувачів між собою, тим більша ІПР користувачів ІС.

Для перевірки цих гіпотез, а також для визначення впливу відсотка помилок (*Och*) та амплітуди (A_{Och}) на залежність ефективності застосування ІНМ для рішення задачі АК ІС від розглянутих раніше параметрів було проведено ряд експериментів.

Для проведення цієї серії експериментів був розрахований за формулою відсоток помилок (*Och*) для кожного користувача.

Потім користувачі були розділені на три групи в залежності від кількості помилок (*Och*) при наборі тексту. При цьому треба помітити, що амплітуда відсотка помилок (A_{Och}) у користувачів між собою в цих групах була різна..

У першій групі помилка (*Och*) дорівнювала 1÷6%, у другій групі – 7.5-9.6%, у третій групі – 10-16.4%. Після чого мережа була протестована для кожної з цих груп.

Тобто у перших трьох групах експериментів першого етапу спільними були наступні умови: кількість ознак (n) – 6, навчальних даних (z_t) – 1500, усереднення – немає (*usr*=неправда); а відрізнялася тільки група користувачів, для яких проводилися експерименти. Результати деяких експериментів (у випадку, коли відбір за словами не проводився) показані на рисунках.

Порівнявши результати цих експериментів можна сказати, що:

♦ В першій і другій групі ІПР відрізняється на 1-7%, причому, чим більше навчальних даних (z_t), тим різниця більша, особливо при відносно великій кількості користувачів (l).

♦ Між першою та третьою, та другою та третьою групами ІПР відрізняється на 15-24%.

При цьому, треба відмітити, що амплітуда відсотка помилок (A_{Och}) у першій та другій групах значно відрізняється – цим і пояснюється відносно не дуже велика різниця ІПР для цих двох груп.

З цієї групи експериментів можна зробити наступні висновки:

♦ Чим менший відсоток помилок (*Och*) при наборі тексту, тим більша ІПР користувачів.

♦ Чим менша амплітуда відсотка помилок (A_{Och}) у користувачів між собою, тим більше ІПР користувачів ІС.

Усі наступні експерименти проводилися для групи користувачів з середньою

кількістю помилок. Крім того, відбувалося порівняння результатів цих експериментів з результатами експериментів, проведених на першому етапі.

У наступній групі експериментів досліджувався вплив усереднення навчальних даних по 10 (*usr*) на ефективність застосування даної мережі для рішення нашої задачі. Тобто у наступній групі експериментів спільними були наступні умови: кількість ознак (*n*) – 6, навчальних даних (z_t) – 1500, відсоток помилок (*Och*) – середній; а відмінність полягала в тому, що усереднення (*usr*) або проводилося або ні. Результати деяких експериментів (у випадку, коли відбір за словами проводився – *ots*=істина) показані на рисунку.

Порівнявши результати цих експериментів можна сказати, що у випадку, коли усереднення не відбувалося (*usr*=неправда), ІПР користувачів була вище на 4-6%.

Таким чином, можна зробити висновок, що кількість навчальних даних (z_t) має більший вплив на результат, ніж невелике збільшення точності навчальних даних.

Потім досліджувався вплив виконання відбору за словами (*ots*) на ІПР користувачів. Тобто у наступній групі експериментів загальними були наступні умови: кількість ознак (*n*) – 6, навчальних даних (z_t) – 1500, відсоток помилок (*Och*) – середній; а відмінність полягала в тому, що відбір за словами (*ots*) або проводилося або ні. Результати деяких експериментів (у випадку, коли усереднення проводилося – *usr*=істина) показані на рисунку.

Порівнявши результати цих експериментів можна сказати, що якщо відбір за словами не проводиться (*ots*=неправда), тоді майже у всіх випадках ІПР користувачів вища на 1-3%, причому чим більше навчальних даних (z_t), тим ця різниця менша.

Тому можна зробити висновок, що краще більше навчальних даних нехай навіть з неточностями, ніж проведення відбору за словами (*ots*=істина), тому що без відбору за словами (*ots*=неправда) «страждає» тільки перша ознака, а кількість навчальних даних (z_t), при цьому, у 10 разів більша.

Далі визначався вплив кількості ознак (*n*) на ІПР користувачів. Тобто у цій групі експериментів відрізнялася кількість ознак (*n*), що змінювалося від 3 до 6 (розглядався час набору останніх 6 літер або останніх чи передостанніх 3 літер); а інші умо-

ви були однакові (навчальних даних (z_t) – 1500, відсоток помилок (*Och*) – середній). Результати деяких експериментів (у випадку, коли усереднення не проводилося – *usr*=неправда, відбір за словами не проводився – *ots*=неправда) показані на рисунку. На цьому рисунку замість окремих кривих для випадків, коли розглядався час набору останніх 3 букв та передостанніх показана одна усереднена крива.

Порівнявши результати цих експериментів, можна сказати, що зміна кількості ознак (n) від 3 до 6 збільшувала імовірність правильної відповіді приблизно на 10%, причому, чим більше користувачів у системі, тим більше поліпшення результату.

Звідси можна зробити висновок, що чим більше ознак (n) використовується, тим ІПР більша.

У наступних експериментах досліджувався вплив кількості навчальних даних (z_t) на ІПР користувачів ІС. У цих експериментах різною була кількість навчальних даних (z_t), що дорівнювала 500, 1000, 1500. А інші умови були однаковими (кількість ознак (n) – 6, відсоток помилок (*Och*) – середній). Результати деяких експериментів (у випадку, коли усереднення не проводилося – *usr*=неправда, відбір за словами не проводився – *ots*=неправда) показані на рисунку.

Порівнявши результати цих експериментів можна сказати:

- Між другим та третім варіантами ІПР відрізняться незначно.
- Між першим та третім, та першим та другим ІПР відрізняється на 2-4%.

Звідси можна зробити висновок, що чим більше навчальних даних (n), тим ІПР вища.

У наступних експериментах досліджувався вплив ширини функції активності (σ) на ІПР.

Проаналізувавши результати цих експериментів, можна сказати, що вдала зміна ширини функції активності (σ) збільшувала імовірність правильної відповіді тільки приблизно на 1-2%.

Можна зробити висновок, що цей параметр мало впливає на ІПР. Тому в усіх, раніше розглянутих експериментах, була обрана ширина функції активності (σ), рівна 0.1.

Підсумувавши аналіз проведених експериментів цього етапу, можна зробити

наступні висновки:

1. Підтвердилися, зроблені на першому етапі, основні висновки про залежність ІПР від найбільш критичних параметрів:

♦ Чим більша кількість користувачів (l) у системі, тим менша ІПР.

♦ Для ІПР роботи мережі кількість навчальних даних (z_t) важливіша, ніж їх первинна обробка, тобто краще більше навчальних даних (z_t) нехай навіть з невеликою похибкою, ніж більш точні дані, але в меншій кількості.

♦ Ширина функції активності (σ) слабо впливає на якість автентифікації.

♦ Збільшення кількості аналізованих ознак (n) досить сильно впливає на достовірність автентифікації.

2. Підтвердилися зроблені раніше, на цьому етапі, припущення про залежність ІПР від відсотка помилок (Och) та від розкиду помилок (A_{Och}) у користувачів між собою; таким чином стало очевидно, що найбільший вплив має кількість помилок (Och) при наборі тексту та, при цьому, чим менша амплітуда відсотка помилок (A_{Och}) у користувачів між собою, тим більша ІПР застосування даного підходу. Тобто, чим більш уважніші користувачі при наборі тексту, тим більша ІПР, крім того, бажано, щоб рівень уважності користувачів був приблизно однаковим.

Результати експериментів третього етапу:

Провівши більш детальний аналіз результатів усіх, раніше розглянутих, експериментів, а також візуально проаналізувавши інформацію, яка є, про КП користувачів ІС, можна зробити висновки про те, що для будь-якого користувача існують записи, в яких значення одного або декількох досліджуваних характеристик (час, необхідний для натискання одного символу) значно відрізняється від середнього значення цієї характеристики. Тобто можна сказати, що для будь-якого користувача є помилкові дані (помилки) [99-116]. Це природно, тому що будь-який користувач при наборі тексту може помилятися, або відволікатися або просто втомитися. Також очевидно, що і при тестуванні мережі, і тим більше при реальній АК навчальні зразки з грубими помилками необхідно виключати з розгляду, тому що при цьому надійність проведеної автентифікації повинна зрости. Але виникає питання, про то які відхилення вважати допустимими, а які – грубими помилками.

При цьому можна допустити, що чим менше максимально допустиме відхилення значення ознаки, що аналізується, від її середнього значення, тим ефективність вища.

Для перевірки цієї гіпотези, а також для визначення впливу процентного значення допустимого відхилення на залежність ефективності застосування імовірнісної нейтронної мережі для рішення задачі АК ІС від кількості навчальних даних (z_t) та від кількості користувачів (l) у системі було проведено ряд експериментів.

Крім того, було розглянуто два алгоритми виключення навчальних зразків (записів) з грубими помилками: ВНЗСВКЗ і ВНЗПОСЗ (один еталонний, а другий запропонований).

Тому в проведених експериментах розглядалися варіанти, в яких при тих самих умовах або не проводилося виключення грубих помилок, або проводилося за алгоритмом ВНЗСВКЗ, або за алгоритмом ВНЗПОСЗ.

Спочатку проводилися експерименти, в яких аналізувалися всі дані, потім такі ж експерименти, але, виключивши навчальні зразки з грубими помилками однієї з ознак за алгоритмом ВНЗСВКЗ, після чого проводилися такі ж експерименти, але, виключивши навчальні зразки з грубими помилками однієї з ознак за алгоритмом ВНЗПОСЗ, вибравши k=1. Результати цих експериментів показані на рисунку.

Проаналізувавши результати цієї групи експериментів, можна сказати, що виключення навчальних зразків з грубими помилками і за одним, і за другим алгоритмом підвищує ефективність ІПР застосування даного підходу, але алгоритм ВНЗПОСЗ виявився більш ефективним. Так при використанні алгоритму ВНЗСВКЗ ефективність ІПР збільшується на 2%-7%, а при використанні алгоритму ВНЗПОСЗ ІПР збільшується на 6%-10% у порівнянні з ІПР для випадку, коли виключення навчальних зразків із грубими помилками взагалі не проводиться.

Для ще одного підтвердження того, що алгоритм ВНЗПОСЗ є більш ефективним, було проведено ряд експериментів, в яких перевірялася якість ознак за алгоритмом ВЯАП. Спочатку проводився експеримент за умови, що не проводиться виключення грубих помилок, потім розглядався випадок, коли навчальні зразки з

грубими помилками виключаються за алгоритмом ВНЗПОСЗ (k=1), а потім тестувався варіант, коли навчальні зразки з грубими помилками виключаються за алгоритмом ВНЗСВКЗ. В усіх випадках визначався критерій якості ознак H.

Деякі результати (на прикладі останніх 6 символів слова “телефонізація”) експериментів продемонстровані на рисунку.

Проаналізувавши отримані в цих експериментах результати, можна зробити наступні висновки:

♦ Після виключення навчальних зразків з грубими помилками за будь-яким з розглянутих алгоритмів (ВНЗСВКЗ або ВНЗПОСЗ) якість ознак значно підвищується.

♦ Після виключення навчальних зразків з грубими помилками за алгоритмом ВНЗПОСЗ якість ознак виявляється значно краща, ніж після виключення за алгоритмом ВНЗСВКЗ тобто:

● Критерій H менший.

● Значення критерію H у всіх ознак майже однакове, на відміну від випадку, коли виключення виконується за алгоритмом ВНЗСВКЗ.

Цим і пояснюється одержувана більш висока якість розпізнавання після виключення навчальних зразків за алгоритмом ВНЗПОСЗ.

Тому для подальших експериментів для виключення навчальних зразків з грубими помилками був обраний алгоритм ВНЗПОСЗ.

Спочатку проводилися експерименти, в яких аналізувалися всі дані, а потім – такі ж експерименти, але з виключенням (за алгоритмом ВНЗПОСЗ) тих записів, в яких відхилення значення однієї з ознак від її середнього значення було більше ніж зазначений коефіцієнт, помножений на середнє значення цієї ознаки.

Значення коефіцієнта в даних експериментах вибиралося з діапазону від 0.5 до 1.0.

Перша група експериментів проводилася при наступних умовах: усереднення навчальних даних не виконувалося (usr=неправда); кількість ознак (n) – 6; кількість навчальних даних (z_t) змінювалася в діапазоні від 100 до 1300; кількість користувачів у системі (l) змінювалася в діапазоні від 2 до 8; а виключення записів з

грубими помилками взагалі не виконувалося ($\left|u_{ijt} - Sr_{it}\right|=\infty$). Результати деяких експериментів показані на рисунках.

Друга група експериментів проводилася за тих самих умов, але з експериментів виключалися записи, в яких хоч одна з ознак відрізнялася від її середнього значення більш ніж на 100% ($k=1$; $\left|u_{ijt} - Sr_{it}\right|=Sr$). Результати деяких експериментів показані на рисунках.

Порівнявши результати першої і другої групи експериментів можна сказати, що в другому випадку:

♦ ІПР збільшується приблизно на 10-18% (показано на рисунках).

♦ При збільшенні l на 1 зменшення ІПР знижується приблизно на 0.2-4% (показано на рисунку).

Крім того, при виключенні записів, в яких помилки більші, ніж 100% зріст ІПР за рахунок збільшення z_t:

♦ Від 100 до 500 понизився приблизно на 3.5-6% (показано на рисунках).

♦ Від 100 до 1000 понизився приблизно на 4.5-8% (показано на рисунках).

♦ Від 100 до 1300 понизився приблизно на 4.7-8.5% (показано на рисунках).

При цьому якщо збільшується кількість користувачів в системі від 2 до 8, тоді збільшення росту ІПР за рахунок збільшення кількості навчальних даних зростає в 2 рази (показано на рисунках).

Третя група експериментів проводилася за тих самих умов, але z_t вибиралася рівною 100, 200, 1000 та з експериментів виключалися записи, в яких хоч одна з ознак відрізнялася від її середнього значення більш ніж на 75% ($k=0.75$; $\left|u_{ijt} - Sr_{it}\right|=0.75*Sr$). Результати деяких експериментів показані на рисунках.

Порівнявши результати першої та третьої груп експериментів можна сказати, що в третій групі:

♦ ІПР збільшується приблизно на 11-21% (показано на рисунках).

♦ При збільшенні l на 1 зменшення ІПР знижується приблизно на 0.4-4.5% (показано на рисунку).

Четверта група експериментів проводилася за тих самих умов, але z_t вибиралася рівною 100, 200 та з експериментів виключалися записи, в яких хоч одна з ознак відрізнялася від її середнього значення більш ніж на 50% (k=0.50; $\left|u_{ijt} - Sr_{it}\right|$=0.50*$Sr$). Результати деяких експериментів показані на рисунках.

Порівнявши результати першої та четвертої групи експериментів можна сказати, що в четвертій групі:

♦ ІПР зростає приблизно на 13-27% (показано на рисунках).

♦ При збільшенні l на 1 зменшення ІПР знижується приблизно на 0.5-6% (показано на рисунку).

Проаналізувавши результати всіх чотирьох груп експериментів можна сказати:

♦ При збільшенні l в системі від 2 до 8 ефективність виключення записів з грубими помилками збільшується приблизно в 2 рази (показано на рисунках).

♦ При збільшенні l від 2 до 8 зменшення зниження ІПР через збільшення l на 1, за рахунок виключення записів з грубими помилками зменшується приблизно в 10 разів (показано на рисунку).

У п'ятій групі експериментів були наступні умови: *usr*=неправда; z_t=100; кількість ознак (n) – 6, l змінювалася в діапазоні від 2 до 8; а $\left|u_{ijt} - Sr_{it}\right|$ змінювалося від 50% до 100% з кроком 10% (k змінювалося від 0.5 до 1 з кроком 0.1).

Проаналізувавши результати цієї групи, можна сказати, що при виключенні записів з грубими помилками зменшення ІПР через збільшення l від 2 до 8 знижується приблизно на 8-14% (показано на рисунку).

Потім була проведена група експериментів, в якій досліджувався вплив кількості аналізованих ознак (n) на якість розпізнавання, за умови, що проводиться виключення навчальних зразків з грубими помилками (k=1). У даній групі експериментів розглядалися не тільки варіанти для n=3 та n=6, але й для n=11, тому відбір за словами проводився (*usr*=істина).

Отримані результати відображені на рисунку. Проаналізувавши отримані результати, можна сказати, що залежність, яка була виявлена на перших двох етапах, здебільшого зберігається, а при збільшенні кількості ознак (n) з 6 до 11 ІПР

збільшується, але її приріст набагато менший (1-6%), ніж при збільшенні кількості ознак (n) з 3 до 6 (12-27%).

Підсумувавши аналіз проведених експериментів цього етапу, можна зробити наступні висновки:

1. Цілком підтвердилося зроблене раніше припущення з приводу залежності ІПР від максимально допустимого відхилення значення аналізованої ознаки від її середнього значення ($\left|u_{ijt} - Sr_{it}\right|$).

2. Стали очевидними наступні факти:

♦ Чим більше користувачів в системі (l), тим більше збільшення ефективності (Ef) при виключенні записів з грубими помилками.

♦ Чим менше максимально допустиме відхилення значення аналізованої ознаки від її середнього значення ($\left|u_{ijt} - Sr_{it}\right|$), тим менша залежність ІПР від кількості користувачів в системі (l).

♦ При виключенні записів з грубими помилками залежність ІПР від кількості навчальних даних (z_t) значно зменшується.

♦ При виключенні записів з грубими помилками, чим більше користувачів в системі (l), тим більший зріст ІПР у випадку збільшення кількості навчальних даних (z_t).

♦ При збільшенні кількості ознак (n) з 3 до 6 тенденція, яка була виявлена на перших двох етапах, зберігається, але при збільшенні кількості ознак (n) з 6 до 11 приріст ІПР різко знизився.

Проблеми застосування даного підходу:

При проведенні експериментів виникли труднощі, які пов'язані по-перше з часовими витратами, а по-друге з підвищеними вимогами до обчислювальної техніки, яка використовується. Ці труднощі пояснюються, тим, що ІНМ дуже вибагливі у відношенні ресурсів [99-116].

Для накопичення необхідної кількості навчальних даних потрібні великі часові витрати. Так, якщо вважати, що мінімальна необхідна кількість навчальних даних (z_t) – 1000 та по 1000 екземплярів використовувати в якості невідомиз зразків, тоді для накопичення даних для одного користувача системи необхідно приблизно 4-5

годин, а якщо тестувати мережу за умови що в системі 20 користувачів, тоді необхідно 80-100 годин. Для зменшення часових витрат адміністратора, який повинен контролювати процес накопичення навчальних даних, в програмі передбачена можливість накопичення навчальних даних на різних комп'ютерах, але при цьому не знижуються часові витрати користувачів системи та витрати, які пов'язані з використанням комп'ютерів.

При аналізі накопиченої інформації також виникають проблеми. Як вже було сказано, чим більша кількість навчальних зразків (z_t) та кількість ознак (n), тим ефективніше працює побудована мережа, але при цьому збільшується обсяг оброблюваної інформації, а отже збільшується необхідна кількість необхідної пам'яті, а також при цьому знижується швидкодія. А це означає, що пред'являються підвищені вимоги до комп'ютера. Ці вимоги стосуються оперативної пам'яті, ємкості твердого диска, частоти процесора. Наприклад, файл свопінгу займав приблизно 700 МБ. І це при тому, що сама програма ІНМАККП займає відносно мало місця. Весь інсталяційний пакет займає приблизно 8 МБ і в нього входить сама інсталяційна програма з необхідними dll-файлами та іншими інсталяційними файлами, крім того, сюди входить необхідна база даних. Тому що для роботи програми ІНМАККП необхідна наявність на комп'ютері Database Engine, то у випадку відсутності цього пакету його необхідно установити. Тому в інсталяційний пакет створеної програми ІНМАККП входять файли, необхідні для установки Database Engine. Для інсталяції необхідна наявність на твердому диску приблизно 27 МБ, з них для самої програми потрібно 19 МБ, та для Database Engine – 8 МБ. При накопиченні великої кількості навчальних даних та при їх аналізі буде потрібно ще приблизно 15 МБ на робочому диску (крім обговорених раніше 700 МБ для файлу свопінгу).

При тестуванні мережі також виникають часові витрати. Так, наприклад, якщо тестувати мережу для всіх комбінацій по 5 користувачів з 8, за умови, що не проводиться усереднення навчальних даних (*usr*=неправда) та використовується по 1000 навчальних зразків для кожного класу (z_t=1000), тоді один такий експеримент займає приблизно 4 години. А якщо проводити експерименти, наприклад, для всіх комбінацій по 5 користувачів з 20, тоді час збільшується до декількох діб. Крім того,

при проведенні експериментів, в яких для розрахунків вибирається оптимальне значення ширини функції активності (σ), тоді часові витрати збільшуються як мінімум у 20 разів.

Головними недоліками даного підходу є часові витрати для накопичення навчальних даних та їх обробки, а також підвищені вимоги до обчислювальної техніки, яка використовується.

Результати експериментів

Основні результати експериментів відображені в табл. 3 та на рис.18-31 [99-116].

Таблиця 3

Залежність ІПР АК ІС за їх КП, від різних значеннях критичних конфігураційних параметрів системи розпізнавання

Параметр, що змінюється	Залежність
$l = \overline{2,8}$ з кроком 1	$\Delta P<0 \therefore \Delta l>0$
$knz_t = \overline{100,1500}$ з кроком 100	$\Delta P>0 \therefore \Delta knz_t >0$
n={3,6,11}	$\Delta P>0 \therefore \Delta n>0$; $\Delta P_{(n(3\to6))} >> \Delta P_{n(6\to11)}$
$\sigma = \overline{0.01,1}$ з кроком 0.01	$\Delta \sigma$ слабо впливає на ΔP
$OCH\_K_{t,in} = \{\overline{1\%,6\%}, \overline{7.5\%,9.6\%}, \overline{10\%,16.4\%}\}$	$\Delta P>0 \therefore \Delta OCH\_K_{t,in} <0$
A_OCH_K	$\Delta P>0 \therefore \Delta A\_OCH\_K <0$
Наявність усереднення навчальних даних (usr={true, false})	$\Delta P>0 \therefore usr$=false
Наявність відбору за словами (ots= {true,false})	$\Delta P>0 \therefore ots$ =false
Наявність виключення НЗ з ГПМ ($vykl$={true,false}). Виключення за яким методом виконується (ПОЗКП, або еталонним) (a_vykl ={1,2})	$\Delta P>0 \therefore vykl$=true; $\Delta H <0 \therefore vykl$=true; ($\Delta P_{(vykl=\text{true})} > \Delta P_{(vykl=\text{false})}$) $\therefore$ чим більше l; залежність P від knz_t значно зменшується $\therefore$ $vykl$=true; $H_{(a\_vykl=1)}<H_{(a\_vykl=2)}$; $S_{t,i(a\_vykl=1)}< S_{t,i(a\_vykl=2)}$; $P_{(a\_vykl=1)}> P_{(a\_vykl=2)}$; чим більше значення l, тим більший ΔP у випадку $\Delta knz_t>0$.
k в $\left\|TM\_K_{t,nz,i} - Sr_{t,i}\right\| \le k * Sr_{t,i}$	$\Delta P > 0 \therefore \Delta k < 0$; чим менше k, тим менша залежність P від l

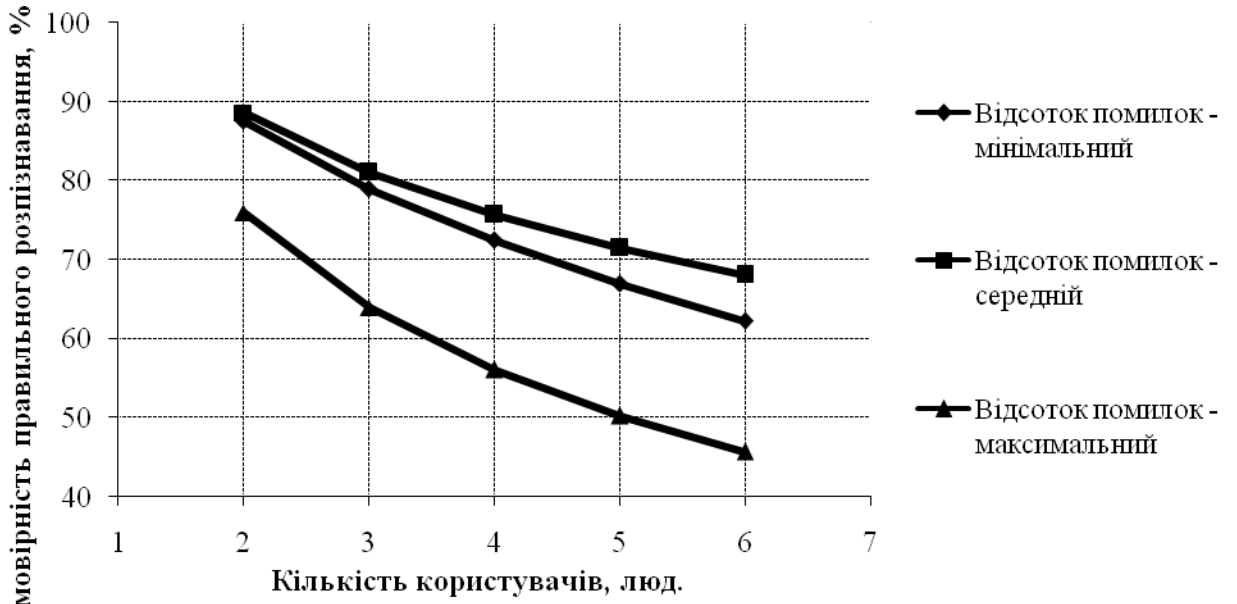


Рис. 18. Вплив кількості помилок під час набору тексту на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак-6, навч. даних-1500, відбір за словами - немає, усереднення - немає

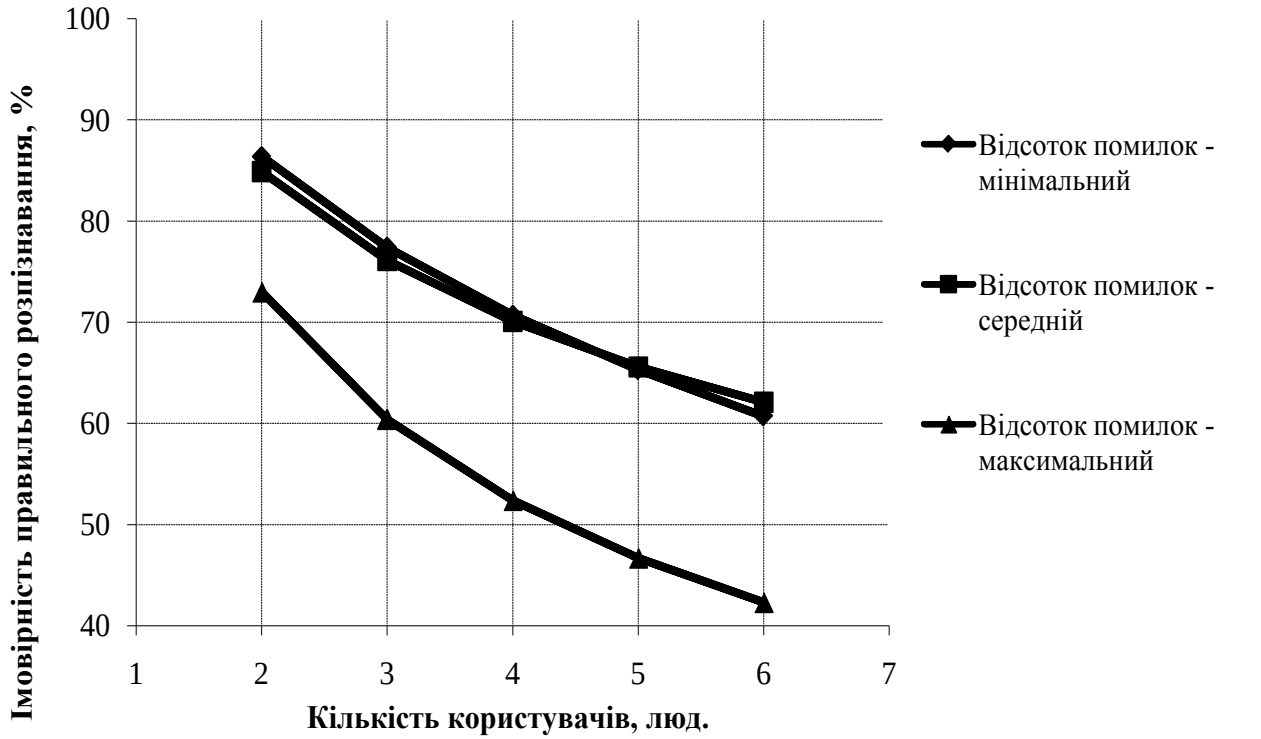


Рис. 19. Вплив кількості помилок під час набору тексту на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак-6, навч. даних-1500, відбір за словами - немає, усереднення - є

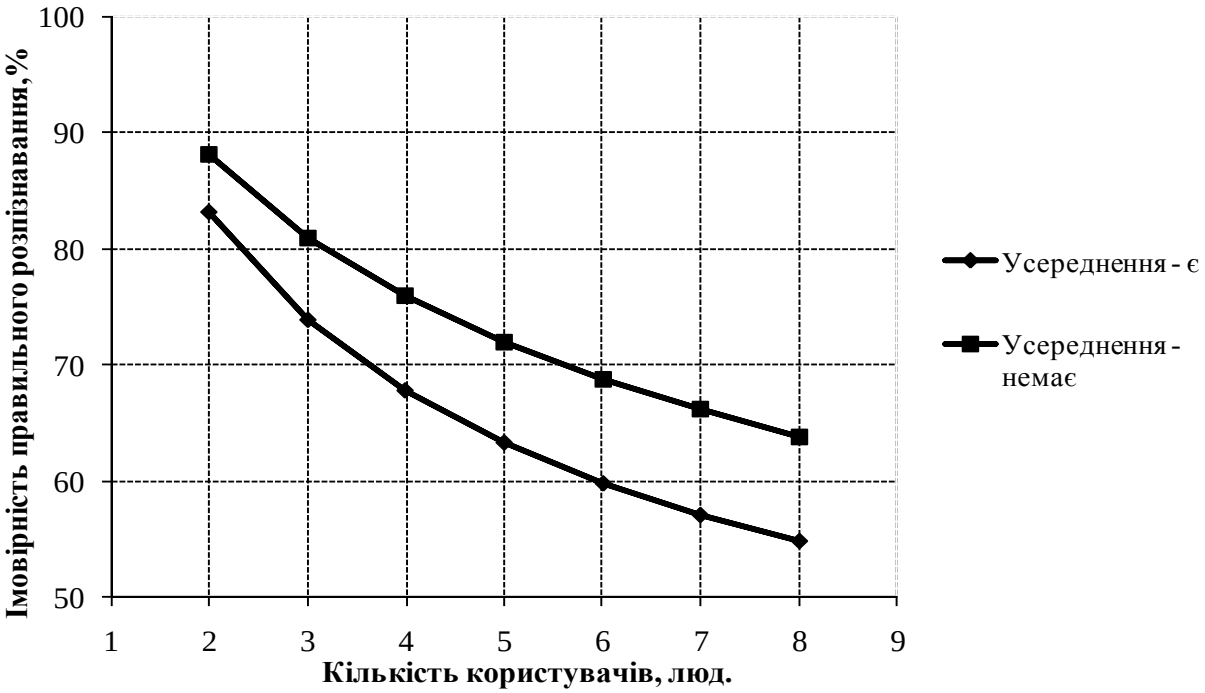


Рис. 20. Вплив усереднення навчальних данних на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак-6, навч. даних-1500, відсоток помилок-середній, відбір за словами - є

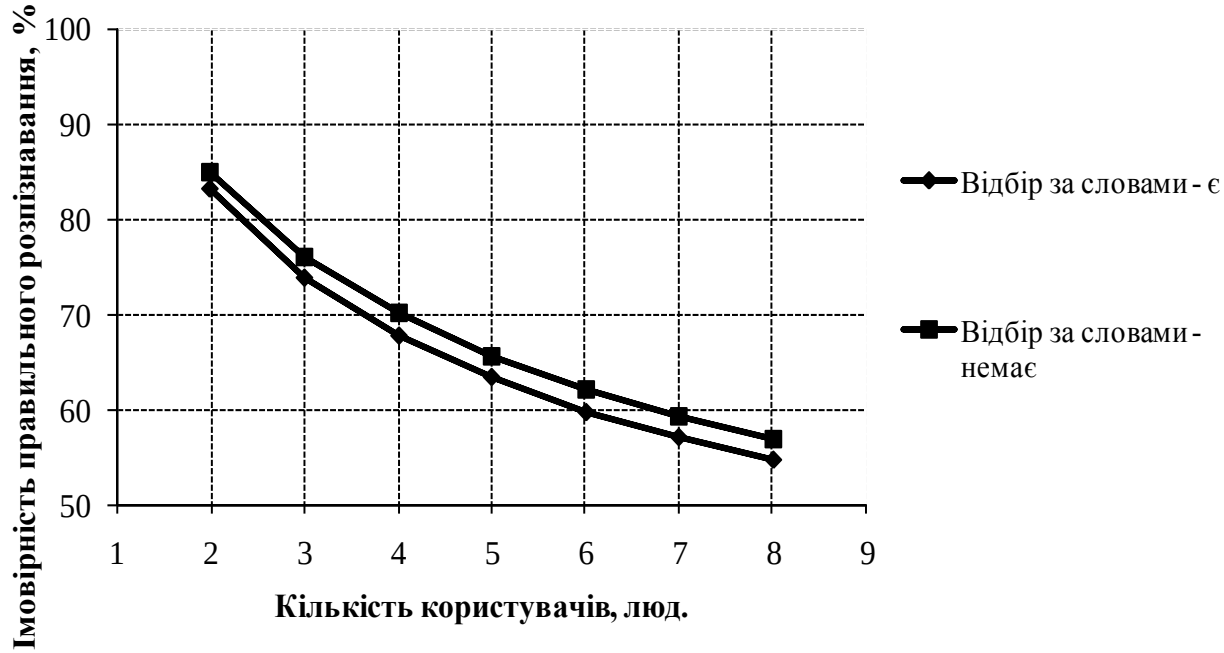


Рис. 21. Вплив відбору за словами в навчальних данних на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак - 6, навч. даних - 1500, відсоток помилок - середній, усереднення - є

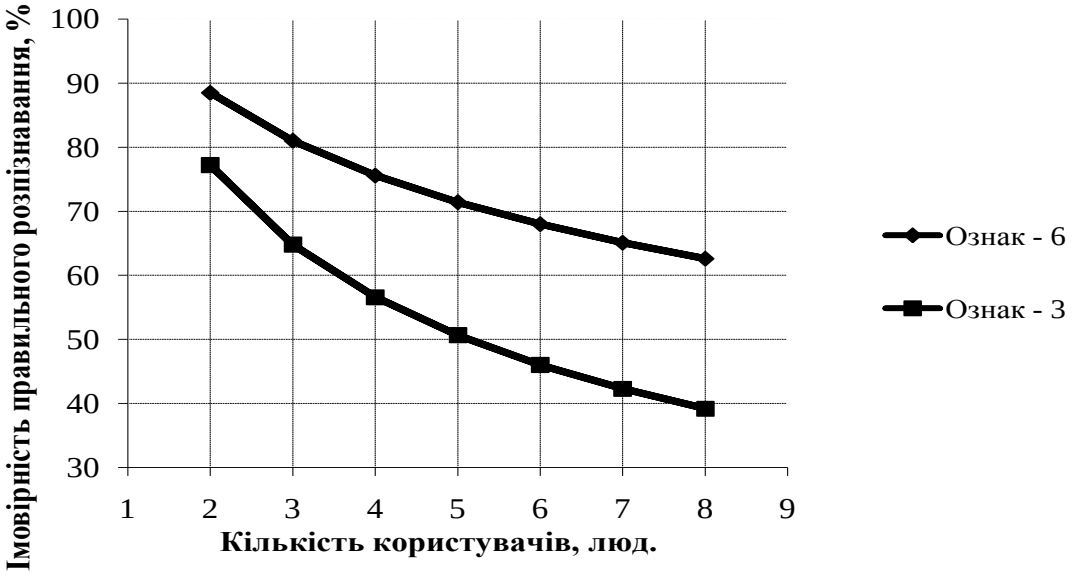


Рис. 22. Вплив кількості ознак на імовірність правильного розпізнавання користувачів ІС, за умови, що навч. даних - 1500, відбір за словами - немає, відсоток помилок - середній, усереднення - немає

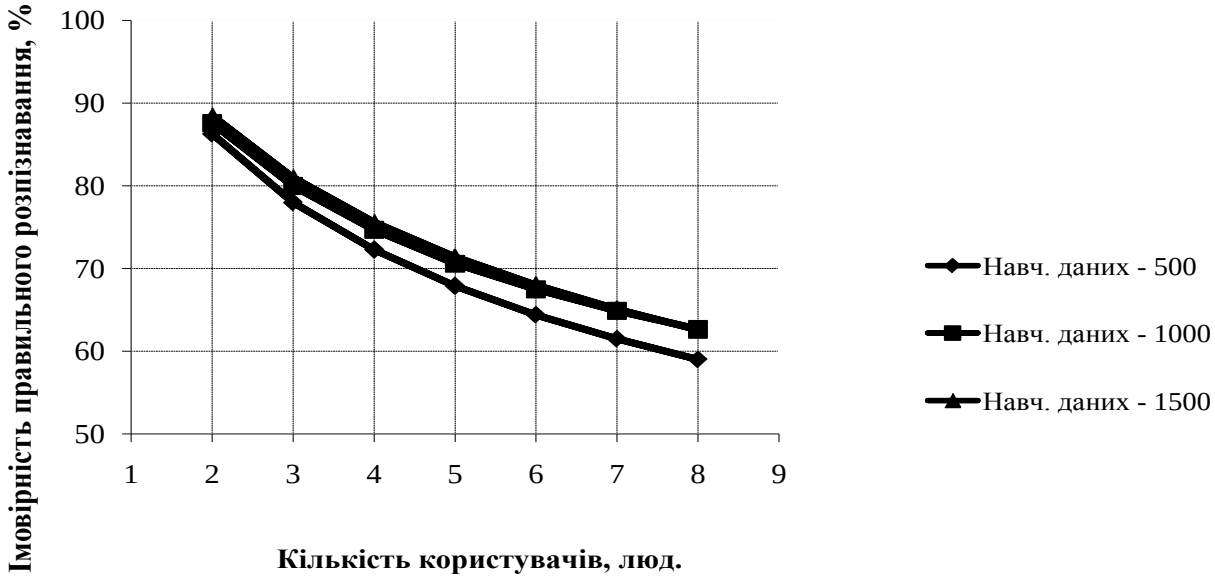


Рис. 23. Вплив кількості навчальних даних на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак - 6, відбір за словами - немає, відсоток помилок - середній, усереднення - немає

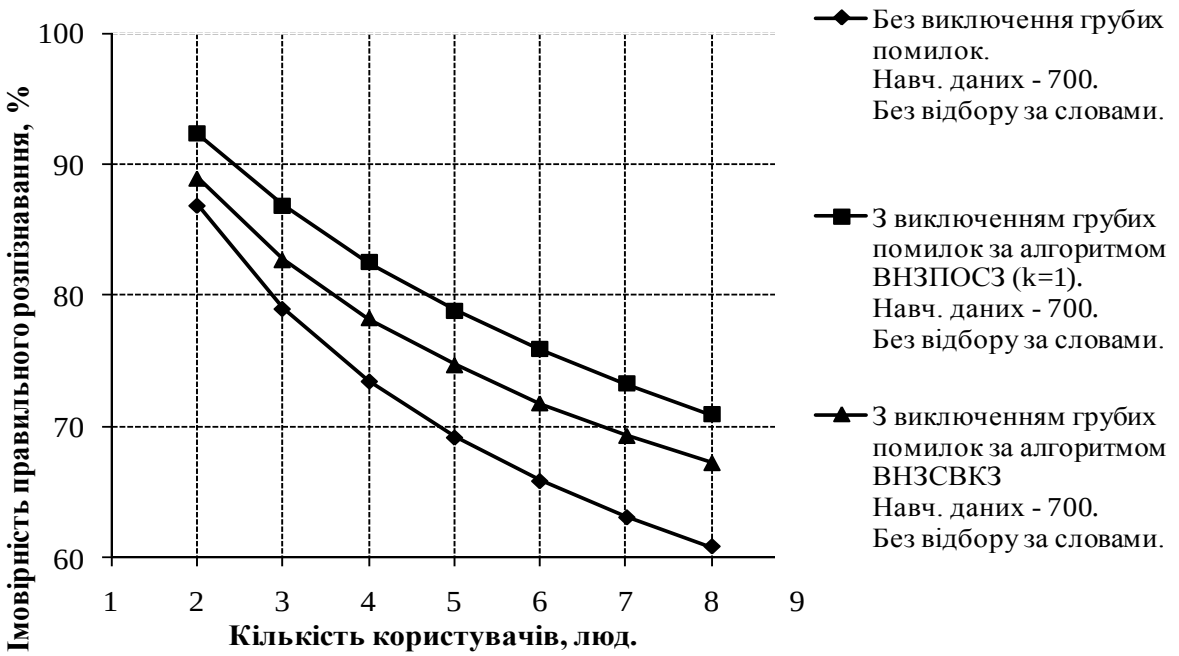


Рис. 24. Вплив виключення навчальних даних з грубими помилками на імовірність правильного розпізнавання користувачів ІС, за умови, що ознак - 6

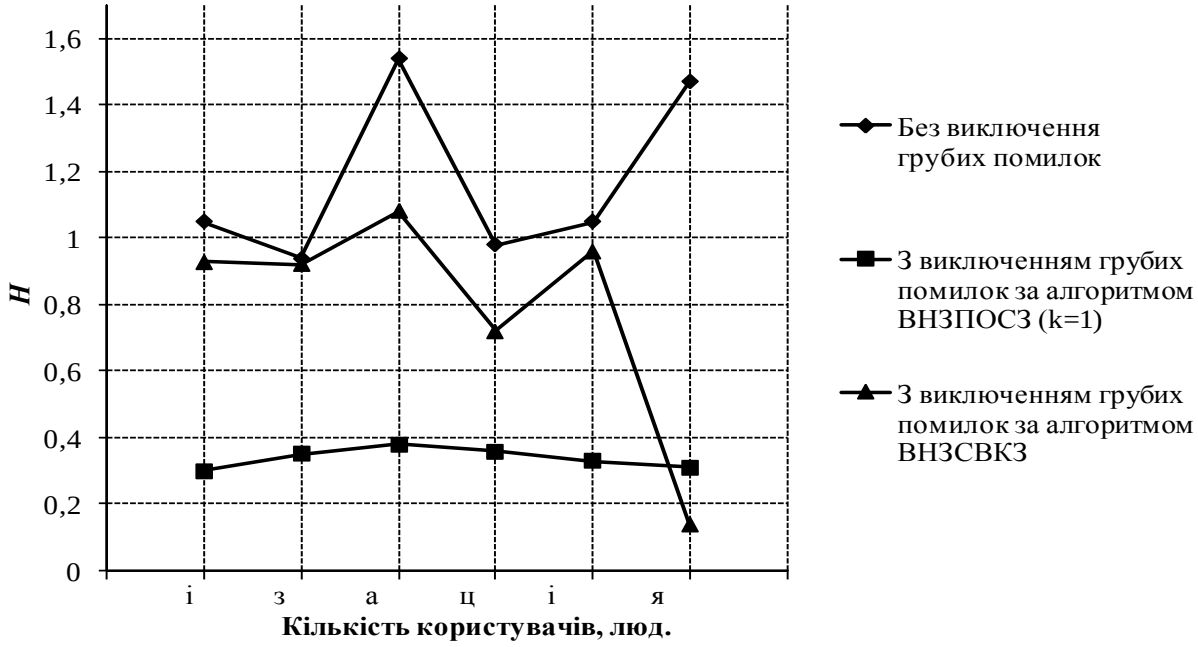


Рис. 25. Вплив виключення грубих помилок на якість останніх 6 ознак в слові "телефонізація"

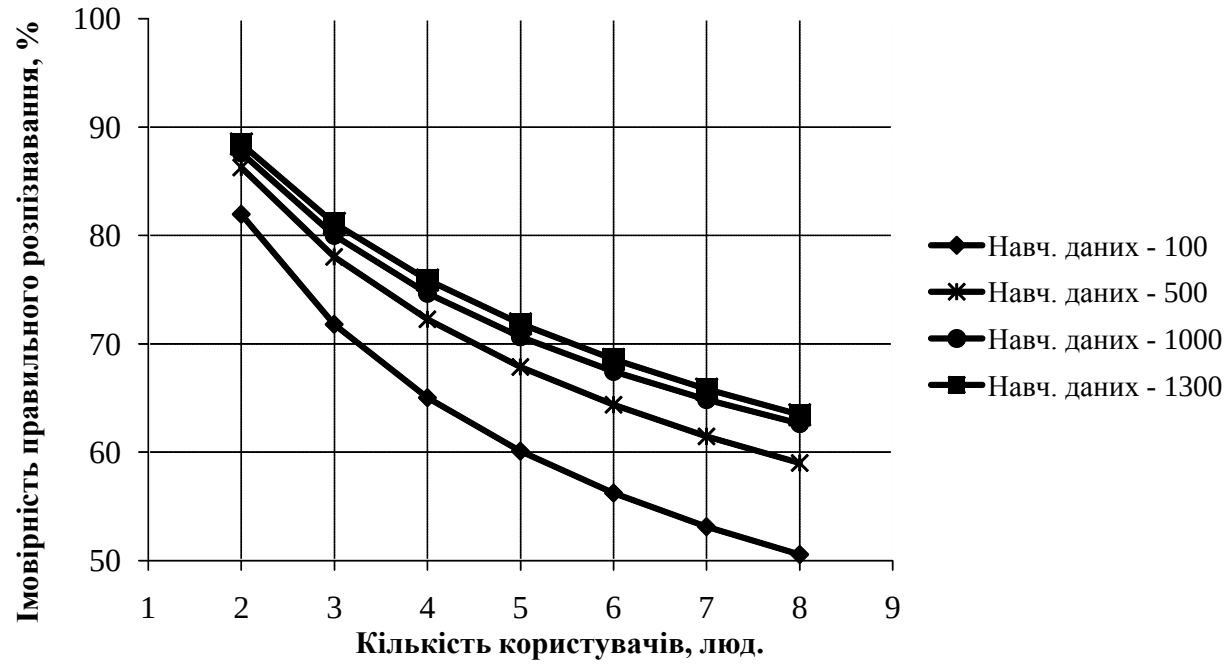


Рис. 26. Вплив кількості навчальних даних на імовірність правильного розпізнавання користувачів ІС, за умови, що виключення грубих помилок не виконується, усереднення не виконується, ознак - 6

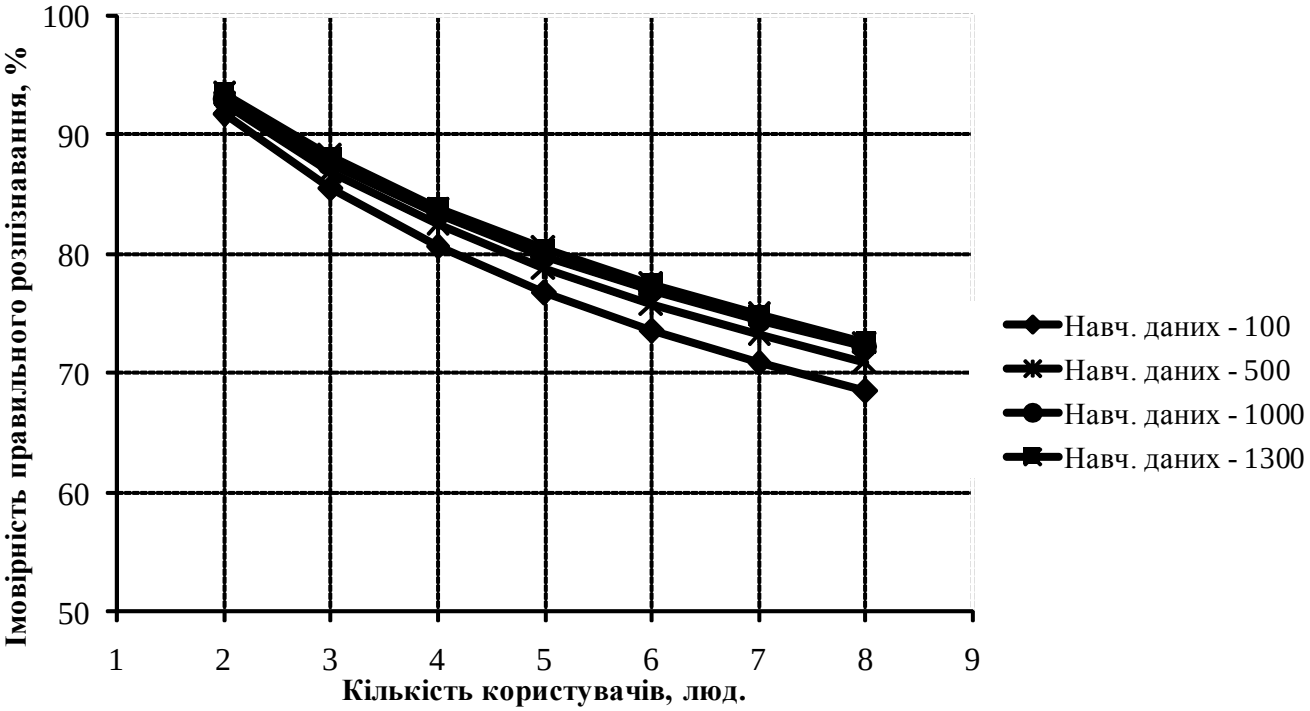


Рис. 27. Вплив кількості навчальних даних на імовірність правильного розпізнавання користувачів ІС, за умови, що виключення грубих помилок виконується за алгоритмом ВНЗПОСЗ (k=1), усереднення не виконується, ознак - 6

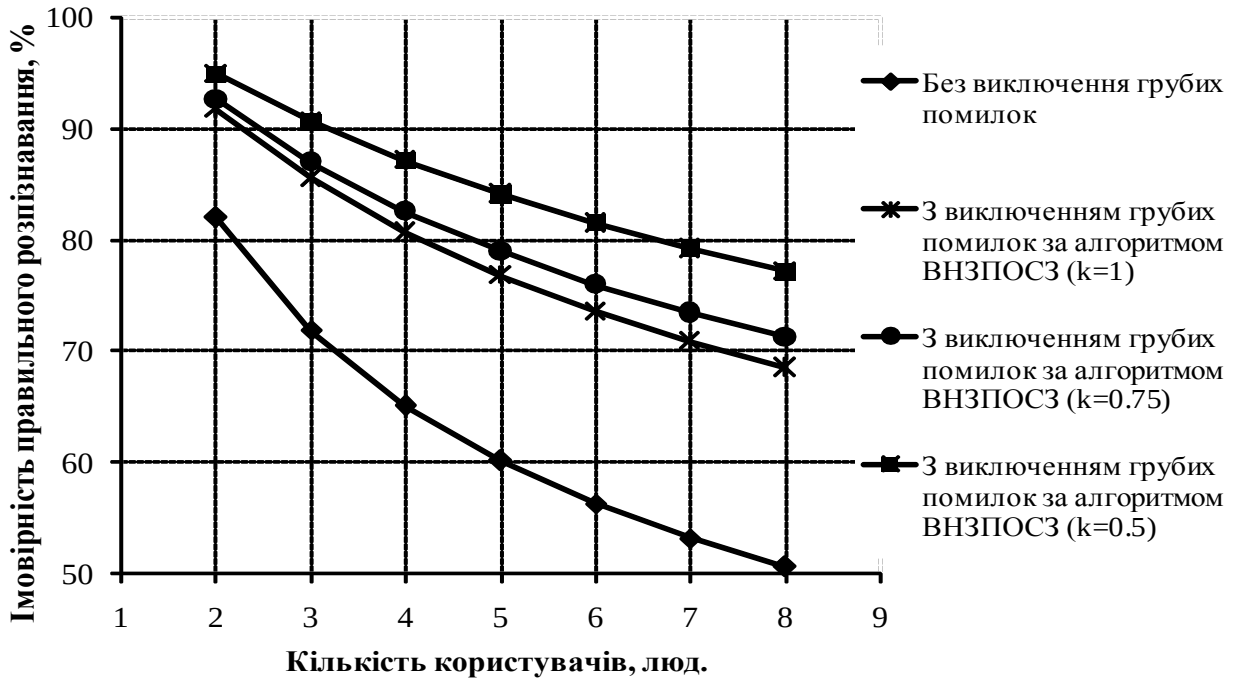


Рис. 28. Вплив виключення навчальних даних з грубими помилками на імовірність правильного розпізнавання користувачів ІС, за умови, що навч. даних-100, усереднення не виконується, ознак- 6

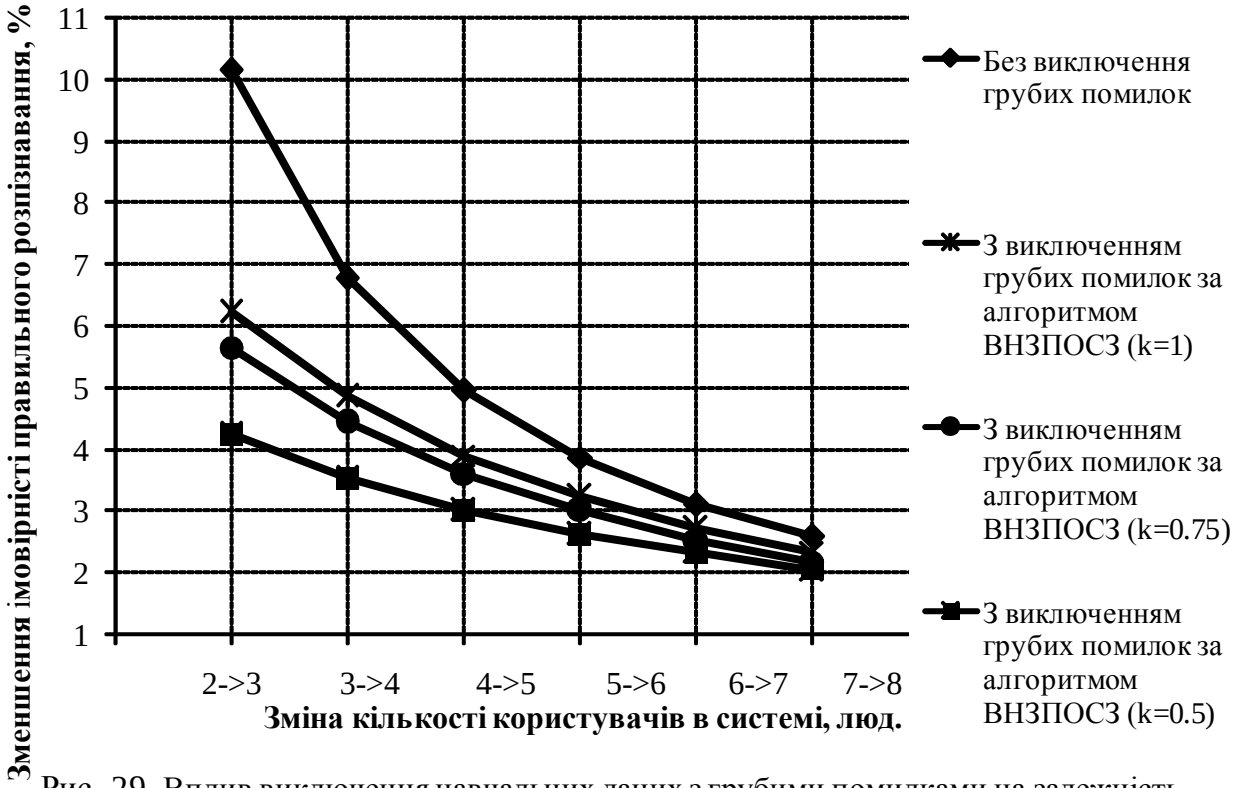


Рис. 29. Вплив виключення навчальних даних з грубими помилками на залежність імовірності правильного розпізнавання користувачів ІС від кількості користувачів, за умови, що навч. даних-100, усереднення не виконується, ознак - 6

Рис. 30. Вплив виключення навчальних даних з грубими помилками за алгоритмом ВНЗПОСЗ на зміну імовірності правильного розпізнавання користувачів ІС при збільшенні кількості користувачів від 2 до 8, за умови, що навчальних даних-100, усереднення-немає, ознак - 6

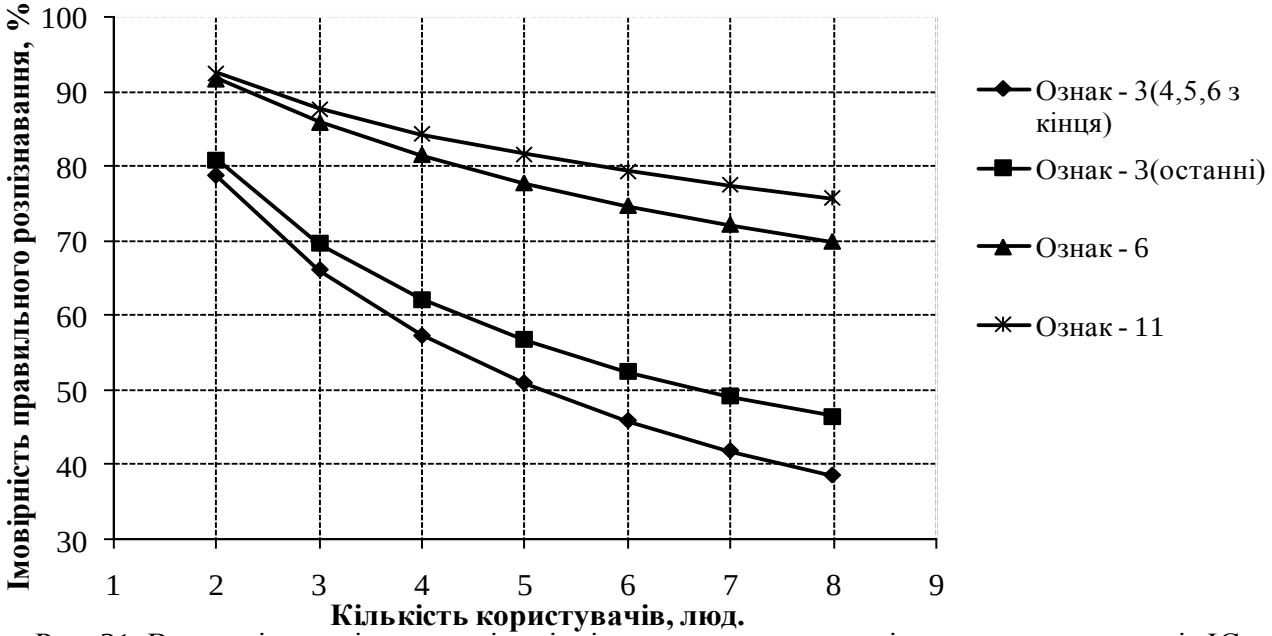


Рис. 31. Вплив кількості ознак на імовірність правильного розпізнавання користувачів ІС, за умови, що виключення навчальних даних з грубими помилками виконується за алгоритмом ВНЗПОСЗ (k=1), відбір за словами - є, навч. даних - 700.

4.2. Тестування методу автентифікації користувачів інформаційних за їх рукописним почерком

Опис експериментів

Для виконання необхідних досліджень відносно АК за їх РП спочатку, на основі розробленої технології АК за їх РП, була створена відповідна автоматизована система. Ця автоматизована система написана мовою Delphi 7 [200]. Для обробки та зберігання даних використовувалася програма для роботи з базами даних Database Desktop та SQL-запити. Це програмне забезпечення реалізує технологію АК за їх РП 1-го раунду, тобто, при оцінюванні ІПР (P) користувачів АС за їх РП, аналізується тільки правильність написання слова-пароля, не перевіряючи, при цьому, стиль написання цього слова-пароля. За допомогою створеного програмного забезпечення була накопичена пробна БДНЗ РП користувачів, після чого, на основі цих даних було проведено ряд експериментів. В цих експериментах змінюючи значення критичних конфігураційних параметрів системи розпізнавання, досліджувалося наступне [116-123]:

1. Вплив наявності в зразку РП параметрів контрольних точок кожного з наступних трьох видів на P: початкових та кінцевих точок; кутових точок; точок перетинання.

2. Вплив застосування процедури видалення помилок 2-4 типів з даних, що аналізуються, на значення P.

3. Вплив застосування, вказаних в роботі, трьох типів корекції даних, що аналізуються, на значення P.

4. Вплив застосування процедури відбору найбільш значимих кутових КТ і вплив значення D на значення P.

5. Вплив застосування процедури згладжування даних, що аналізуються, тобто їх усереднення, на значення P.

На основі аналізу результатів проведених експериментів були зроблені наступні висновки:

1. Наявність в зразку РП параметрів контрольних точок кожного з трьох видів (початкових та кінцевих точок, кутових точок, точок перетинання) збільшую

значення *P*.

2. Застосування процедури видалення помилок 2-4 типів з даних, що аналізуються, збільшує значення *P* та зменшую час розпізнавання зразка почерку.

3. Застосування, вказаних в роботі, трьох типів корекції даних, що аналізуються, необхідне для правильного розпізнавання зразків почерку.

4. Застосування процедури відбору найбільш значимих кутових КТ збільшує значення *P* та зменшую час розпізнавання зразка почерку.

5. При зменшенні значення *D* збільшується значення *P*, але збільшується і час розпізнавання зразка почерку.

6. Застосування процедури згладжування даних, що аналізуються, тобто їх усереднення, внаслідок втрати значимих КТ, зменшує значення *P*.

Результати експериментів

Основні результати експериментів відображені в табл. 4 та на рис.32-35 [116-123].

Таблиця 4

Залежність ІПР АК ІС за їх РП та часу розпізнавання (*TR*), від різних значеннях критичних конфігураційних параметрів системи розпізнавання

Параметр, що змінюється	Залежність
Використання множини КТ $\boldsymbol{KTS}_c$ кожного з 3-х типів (*mkt*={true,false})	$\Delta P > 0 \dot{\cdot}\dot{:} \; mkt = \text{true}; \Delta TR < 0 \dot{\cdot}\dot{:} \; mkt = \text{true}$
Відбір найбільш значимих кутових КТ (*vkt*= {true,false})	$\Delta\ P{>}0 \dot{\cdot}\dot{:}\ mkt{=}\ vkt;\ \Delta\, TR{<}0 \dot{\cdot}\dot{:}\ mkt{=}\ vkt$
Видалення помилок 2-4 типів (*p*24={true,false})	$\Delta TR < 0 \dot{\cdot}\dot{:} \; mkt = \text{true}; \Delta P > 0 \dot{\cdot}\dot{:} \; mkt = \text{true}$
Виконання корекції 1-3 типів (*kor*13={true,false})	$\Delta\ P{>}0 \dot{\cdot}\dot{:}\ kor13{=}\text{true}$
D	$\Delta P > 0 \dot{\cdot}\dot{:} \; \Delta D < 0; \Delta TR > 0 \dot{\cdot}\dot{:} \; \Delta D < 0$
Виконання згладжування даних (*sgl*={true,false})	$\Delta\ P{<}0 \dot{\cdot}\dot{:}\ \Delta\, sgl{=}\ \text{true}$

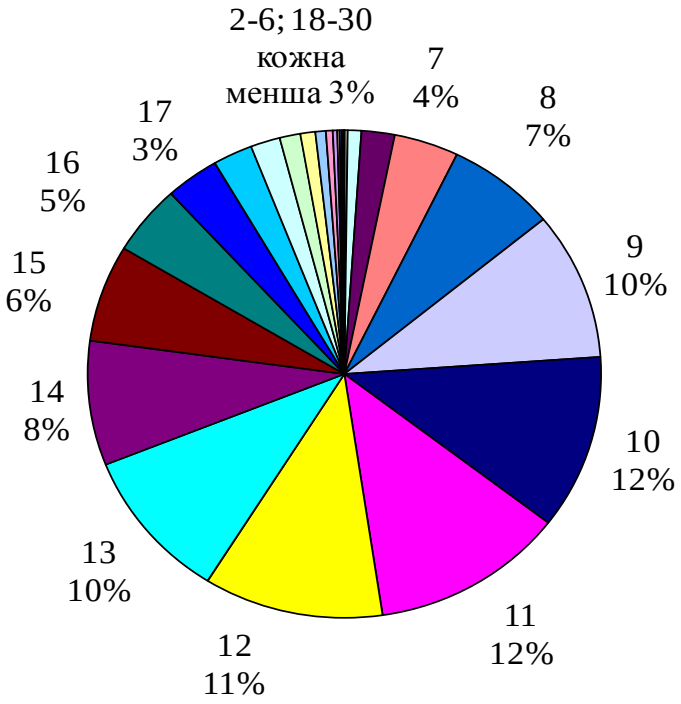


Рис. 32. Розподіл значень кількості КТ в одному символі

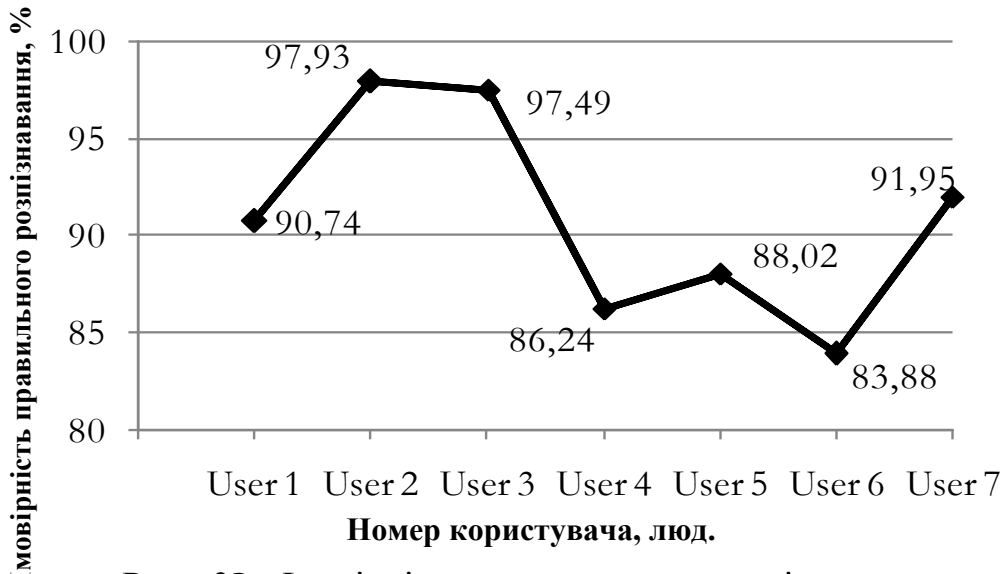


Рис. 33. Імовірність правильного розпізнавання користувачів ІС за їх РП

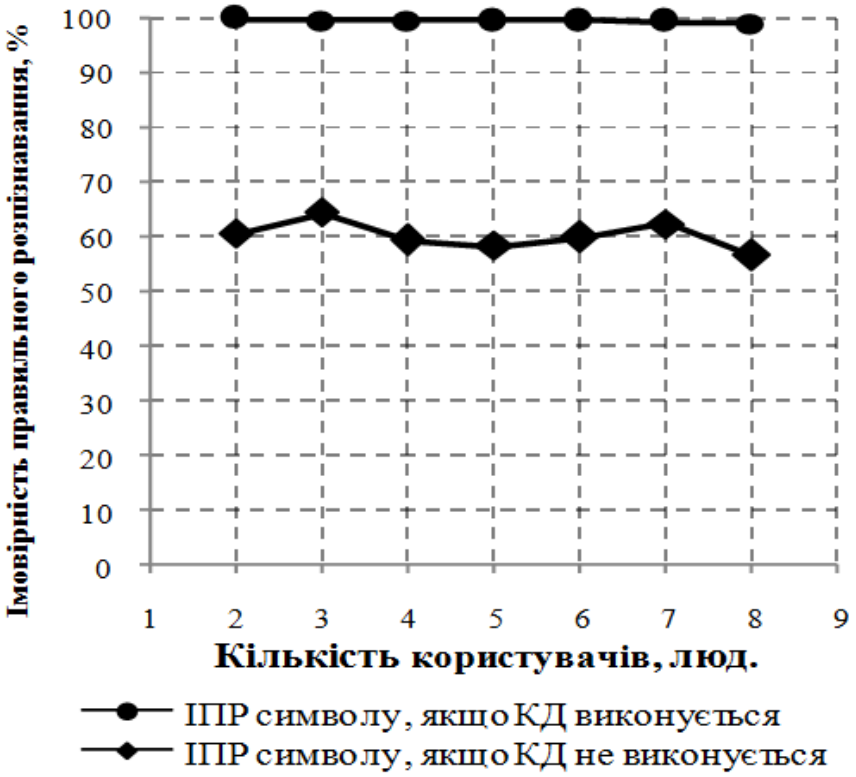


Рис. 34. Імовірність правильного розпізнавання

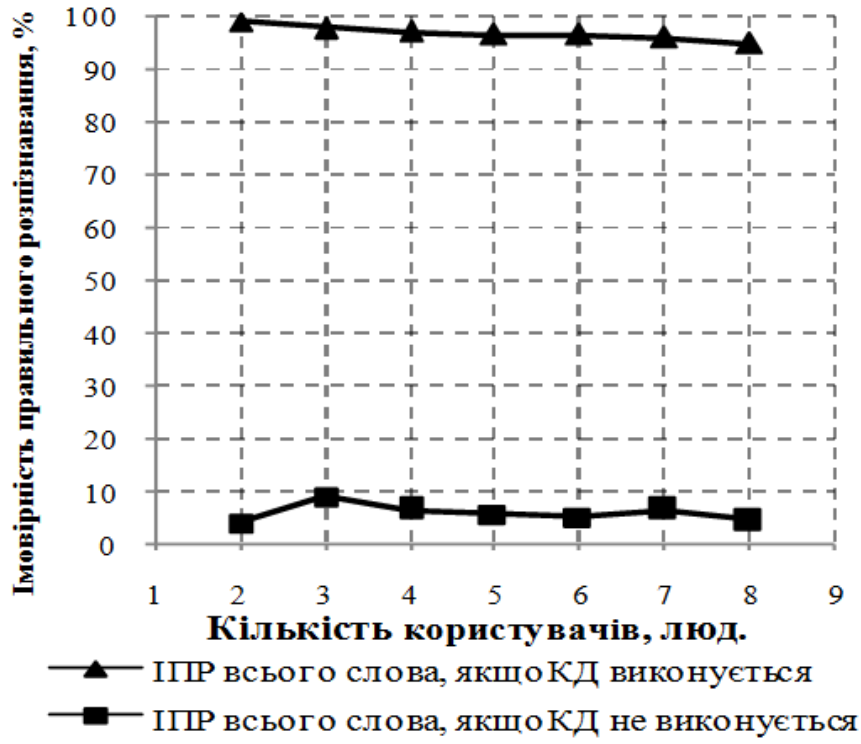


Рис. 35. Імовірність правильного розпізнавання

4.3. Порівняльний аналіз розроблених методів з найбільш поширеними методами розпізнавання за обраними біометричними характеристиками

Виконаємо порівняльний аналіз розроблених методів з найбільш поширеними методами розпізнавання (з відкритих джерел) за обраними біометричними характеристиками (табл.5) [70-123].

Таблиця 5

Порівняння найбільш поширених методів розпізнавання користувачів

Характеристика / Метод	Т	О	Н	Д	Ф	Х	Б	А	Л	П	У	К
АФЩРІВЗ	КП	М	М	В	-	-	-	-	В	С	С	С
ВКВБ	КП	В	М	В	-	-	-	-	В	С	С	С
ВШНМ	КП	В	М	С	-	-	-	-	В	С	С	С
АСХ	КП	С	С	В	-	-	-	-	В	С	С	С
АСХФПФ	КП	С	В	С	-	-	-	-	В	С	С	С
АМОД	КП	М	М	В	-	-	-	-	В	С	С	С
АМОДДЗ	КП	М	М	В	-	-	-	-	В	С	С	С
ВРКС	КП	М	М	В	-	-	-	-	В	С	С	С
ВКР	КП	М	М	В	-	-	-	-	В	С	С	С
АРКНПРЗК	КП	М	М	В	-	-	-	-	В	С	С	С
АРКНПЗК	КП	М	М	В	-	-	-	-	В	С	С	С
ВКНС	КП	В	М	С	-	-	-	-	В	С	С	С
ВБП	КП	М	В	С	-	-	-	-	В	С	С	С
ТААВ	КП	В	С	В	-	-	-	-	В	С	С	С
ТАІ	КП	С	С	М	+	-	+	-	В	С	С	С
ВПММ	РП	В	В	В	-	-	-	+	В	С	С	С
ВБРНМ	РП	С	С	М	-	-	-	+	В	С	С	С
FSRP	РП	С	С	С	-	-	-	+	В	С	С	С
ТАГК	РП	М	М	С	-	-	-	+	В	С	С	С
ВПНК	РП	М	М	С	-	-	-	+	В	С	С	С
АККП з ПОЗКП	КП	С	С	М	+	+	+	-	В	С	С	С
АКРП з ПОЗРП	РП	М	М	М	-	+	+	+	В	С	С	С

Проаналізовані наступні методи розпізнавання, які засновані на: АФЩРІВЗ – аналізі функції щільності розподілу імовірностей випадкових змінних; ВКВБ – використанні класифікатора з векторним базисом; ВШНМ – використанні штучних нейронних мереж; АСХ – аналізі статистичних характеристик; АСХФПФ – аналізі статистичних характеристик для фіксованої парольної фрази; АМОД – аналізі математичного очікування та дисперсії; АМОДДЗ – аналізі математичного очікування та дисперсії, з виконанням додаткового зважування; ВРКС – використанні рангової кореляції Спірмена; ВКР – використанні корекції рангів; АРКНПРЗК – аналізі ритму клавіатурного набору з пороговим завданням класів; АРКНПЗК – аналізі ритму клавіатурного набору з пропорційним завданням класів; ВКНС – використанні К-найближчих сусідів; ВБП – використанні багатошарового персептрона; ТААУ – технології аналізу Амфреса Д. і Вільямса Г.; ТАІ – технології аналізу Іванова; ВПММ – використанні прихованих Марковських моделей; ВБРНМ – використанні багатови-

мірних рекурентних нейронних мереж; FSRP – метод синтаксичного аналізу; ТАГК – технології аналізу Генуе Р. і Кечаді Т.; ВПНК – використанні параметричного навчання класифікатора. Данні методи проаналізовані за наступними характеристиками: Т – біометрична характеристика, яка використовується для автентифікації; О – імовірність помилкової відмови доступу легальному користувачеві; Н – імовірність помилкового надання доступу порушникові; Д – необхідна довжина зразка почерку; Ф – використання для аналізу функціонального стану користувача; Х – використання для аналізу характеристик, необхідних працівникам критичних професій (уважності, сконцентрованості, акуратності); Б – використання в якості етапу багатофак.-торної автентифікації; А – необхідність додаткового апаратного забезпечення; Л – невід'ємність; П – непідробленість; У – унікальність; К – стабільність. Деякі позначення значень характеристик: В – велика; С – середня; М – маленька.

Дана порівняльна таблиця демонструю перевагу по ряду характеристик, розроблених методів АККП та АКРП, відносно вже існуючих методів.

4.4. Висновки до четвертого розділу

В даному розділі на основі запропонованих методу та алгоритму розпізнавання користувачів за їх КП, з використанням методу обробки навчальних даних, створене програмне забезпечення для реалізації біометричної АК ІС за їх КП, яке дозволяє збільшити ступень багатофакторності автентифікації ІС не вимагаючи для цього додаткового обладнання. Також на основі запропонованих методу та алгоритму розпізнавання користувачів за їх РП, з використанням методу обробки навчальних даних, створене програмне забезпечення для реалізації біометричної АК ІС за їх РП, що дозволяє збільшити ступень багатофакторності автентифікації ІС при наявності стандартних сенсорних засобів вводу графічної інформації. Після чого на основі результатів проведених експериментів, за допомогою розробленого програмного за безпечення, здійснено вибір конфігураційних параметрів, налаштування яких є найбільш критичними для збільшення ІПР користувачів ІС та отримана оцінка ІПР користувачів ІС за обраними біометричними характеристиками, яку забезпечує використання ІНМ.

ВИСНОВКИ

У дисертаційній роботі, на основі проведених теоретичних та експериментальних досліджень, розроблені методи АК ІС, які використовують для розпізнавання біометричні характеристики користувача, застосовуючи для ідентифікації біометричного образу користувача нейронні мережі, а також здійснено пошук способів підвищення ІПР користувачів ІС та зниження об'єму затрачуваних ресурсів. У результаті виконання дисертаційної роботи отримані наступні результати:

1. На основі удосконаленої класифікації існуючих біометричних систем розпізнавання та їх порівняльного аналізу, здійснено вибір біометричних методів АК ІС, застосування яких забезпечує задану ІПР користувачів та не потребує суттєвих витрат на впровадження.

2. Вперше запропоновано метод первинної обробки зразків КП, який за рахунок аналізу спектральних характеристик почерку користувача, дозволяє виключити хибні зразки почерку, які є нехарактерними і викликані випадковими помилками користувача при наборі тексту і відрізняється від еталонного методу (метод С.П. Расторгуєва) тим, що забезпечує більш високу (в 2-3 рази) та рівномірну якість характеристик почерку та, завдяки цьому, збільшується ІПР користувачів на 10-20%.

3. Вдосконалено метод АК ІС за їх КП, який за рахунок використання для розпізнавання ІНМ та виконання первинної обробки зразків КП, збільшує ІПР користувачів ІС, в порівнянні з еталонним методом (метод ВШНМ), на 4-8%.

4. Вперше запропоновано метод первинної обробки зразків РП, в якому за рахунок автоматизації процесу відбору КТ в зразках РП, чиї характеристики аналізуються, видаленню помилкових точок п'яти типів та проведенню корекції даних трьох типів, досягається збільшення ІПР користувачів ІС на 35-40% та, завдяки зменшенню кількості ознак в зразках, що аналізуються, зменшення затрачуваних ресурсів.

5. Вдосконалено метод АК ІС за їх РП, який за рахунок використання для розпізнавання ІНМ та виконання первинної обробки зразків РП, збільшує ІПР користувачів ІС, в порівнянні з еталонним методом (метод ВБРНМ), на 7-8%.

6. На основі запропонованих методу та алгоритму розпізнавання користувачів за їх КП, з використанням методу обробки навчальних даних, створене програмне

забезпечення для реалізації біометричної АК ІС за їх КП, яке дозволяє збільшити ступень багатофакторності автентифікації ІС не вимагаючи для цього додаткового обладнання.

7. На основі запропонованих методу та алгоритму розпізнавання користувачів за їх РП, з використанням методу обробки навчальних даних, створене програмне забезпечення для реалізації біометричної АК ІС за їх РП, що дозволяє збільшити ступень багатофакторності автентифікації ІС при наявності стандартних сенсорних засобів вводу графічної інформації.

8. На основі результатів проведених експериментів, за допомогою розробленого програмного забезпечення, здійснено вибір конфігураційних параметрів, налаштування яких є найбільш критичними для збільшення ІПР користувачів ІС та отримана оцінка ІПР користувачів ІС за обраними біометричними характеристиками, яку забезпечує використання ІНМ.

9. Результати дисертаційних досліджень впроваджені в наступних організаціях: в Управлінні верифікації Генерального штабу Збройних сил України (акт від 25.03.2010р.), в управлінні Пенсійного фонду України у Києво-Святошинському районі (акт від 27.05.2010р.), в підприємстві «Інтегратор» (акт від 24.12.2013р.), в ТОВ «НВЦ «ІНФОЗАХИСТ» (акт від 12.12.2018р.), в ІПМЕ ім. Г.Є. Пухова НАН України (акт від 11.01.2019р.), а також використовуються у навчальному процесі кафедри комп’ютеризованих систем захисту інформації Національного авіаційного університету (акт від 29.05.2019р.).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах», 1994.

2. «Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу», *НД ТЗІ 2.5-004-99*, К.: ДСТСЗІ СБ України, 1999р.

3. Д.П. Зегжда, А.М. Ивашко, *Основы безопасности информационных систем*, М.: Горячая линия - Телеком, 2000, С. 452.

4. К.Ю. Гундарь, А.Ю. Гундарь, Д.А. Янишевский, *Защита информации в компьютерных системах*, К.: «Корнейчук», 2000, С. 152.

5. А.П. Зайцев, А.А. Шелупанов, Р.В. Мещеряков, *Технические средства и методы защиты информации*, Машиностроение, 2009, С. 508.

6. Г.А. Бузов, *Защита информации ограниченного доступа от утечки по техническим каналам*, ТОВ «Диалектика-Вильям», 2017, С. 586.

7. Hossein Bidgoli, *Handbook of Information Security: Threats, Vulnerabilities, Prevention, Detection, and Management*, vol.3, Wiley, 2006, С. 1152.

8. Mark Stamp, *Information Security: Principles and Practice*, Wiley, 2011, С. 606.

9. А.Ю. Щеглов, *Защита информации: основы теории. Учебник для бакалавриата и магистратуры*, Юрайт, 2017, С. 309.

10. В.П. Мельников, С.А. Клейменов, А.М. Петраков, *Информационная безопасность. Учебник*, Академия, 2008, С. 336.

11. Н. Голдуев, С. Зефиров, В. Голованов, В. Андрианов, *Обеспечение информационной безопасности бизнеса (2-е изд.)*, Альпина Паблишер, 2011, С. 373.

12. Д. Мельников, *Информационная безопасность открытых систем. Учебник*, А.: ФЛИНТА, 2019, С. 444.

13. Ю.М. Краковский, *Защита информации. Учебное пособие*, Феникс, 2017, С. 348.

14. А. Корченко, А. Архипов, С. Казмирчук, *Анализ и оценивание рисков информационной безопасности. Монография*, Киев: ООО «Лазурит-Полиграф», 2013, С. 275.

15. Ф. Приставка, П. Павленко, С. Казмирчук, М. Коломиец «Исследование средств оценивания рисков безопасности ресурсов информационных систем», *Захист інформації*, Т.19, №1, С. 47-56, 2017.

16. В.А. Ворона, В.А. Тихонов, *Системы контроля и управления доступом*, Изд-во «Горячая линия-Телеком», 2012, С. 272.

17. А.Ю. Зубов, *Коды аутентификации*, Изд-во «Гелиос АРВ», 2017, С. 256.

18. Richard E. Smith, *Authentication: From Passwords to Public Keys*, Addison-Wesley Professional, 2013, С. 576.

19. А.А. Афанасьев, Л.Т. Веденьев, А.А. Воронцов, *Аутентификация. Теория и практика обеспечения безопасного доступа к информационным ресурсам*, Изд-во «Горячая линия-Телеком», 2012, С. 550.

20. М.А. Стюгин, *Метод аутентификации с использованием динамических ключей*, Изд-во «Синергия», 2016, С. 110.

21. А.В. Еременко, *Двухфакторная аутентификация пользователей компьютерных систем на удаленном сервере по клавиатурному почерку*, Изд-во «Синергия», 2015, С. 153.

22. Ричард Э. Смит, *Аутентификация: от паролей до открытых ключей.: Пер. с англ.*, М.: Издательский дом «Вильямс», 2002, С. 432.

23. А.И. Иванов, «Объединение протоколов аутентификации», *Защита информации. Конфидент*, № 1, С. 64-69, 2002.

24. А.В. Еременко, Е.А. Левитская, А.Е. Сулавко, А.Е. Самотуга, «Разграничение доступа к информации на основе скрытого мониторинга действий пользователей в информационных системах: скрытая идентификация»,

Вестник Сибирской государственной автомобильно-дорожной академии, № 6 (40), 2014.

25. J.C. Liou et al., «A Sophisticated RFID Application on Multi-Factor Authentication», *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on IEEE*, 2011, С. 180 – 185.

26. А.Е. Сарбуков, «Аутентификация в компьютерных системах», *Системы безопасност*, № 5(53), С. 118–122, 2003.

27. Р. Болл, *Руководство по биометрии*, Изд-во «Техносфера», 2015, С. 370.

28. Ю.И. Лебеденко, *Биометрические системы безопасности: учебное пособие*, Изд-во «ТулГу», 2012, С. 159.

29. А.М. Прудник, Г.А. Власова, Я.В. Рощупкин, *Биометрические методы защиты информации*, БГУИР, 2014, С. 123.

30. M. Birmingham, I. Turner, *Biometric Security Hardcover*, Изд-во «Owlkids», 2017, С. 70.

31. G. Gud, H. Wechsler, *Mobile Biometrics*, Изд-во «Institution of Engineering and Technology», 2017, С. 450.

32. C. Vielhauer, *User-Centric Privacy and Security in Biometrics Hardcover*, Изд-во «Institution of Engineering and Technology», 2017, С. 432.

33. P. Tuyls, B. Skoric, *Security with Noisy Data: On Private Biometrics, Secure Key Storage and Anti-Counterfeiting*, Изд-во «Springer», 2013, С.340.

34. P. Reid, *Biometrics for Network Security*, Изд-во «Prentice Hall PTR», 2013, С. 288.

35. David Chek Ling Ngo, Andrew Beng, Jin Teoh, Jiankun Hu, *Biometric Security*, Изд-во «Cambridge Scholars Publishing», 2015, С. 497.

36. «Biometric Recognition: Challenges and Opportunities», *National Research Council*, Изд-во «National Academies Press», 2010, С. 182.

37. Е.А. Высоцкая, А.Н. Давиденко, «Классификация биометрических систем аутентификации», *Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г. Є. Пухова*, Вип. 27, С. 108-114, 2004.

38. Е.А. Высоцкая, А.Н. Давиденко, «Определение критичных параметров при выборе биометрической системы аутентификации», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 27, С. 80-86, 2004.

39. Е.А. Высоцкая, «Оценка качества методов биометрической аутентификации и способы его повышения», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 28, С. 94-102, 2004.

40. В.В. Вихман, А.А. Якименко, *Биометрические системы контроля и управления доступом в задачах защиты информации*, Изд-во НГТУ, 2016, С. 52.

41. А.В. Харитонов, «Обзор биометрических методов идентификации личности», *NB: Кибернетика и программирование*, № 2, С. 12-19, 2013.

42. А.А. Якименко, В.В. Вихман, *Внедрение биометрической идентификации в системы контроля и управления доступом: учебное пособие*, Новосибирск: Изд-во НГТУ, 2016, С. 46.

43. Д.В. Брилюк, В.В. Старовойтов, *Распознавание человека по изображению лица нейросетевыми методами*, Изд-во БИТК, 2012, С. 53.

44. С.Л. Бочкарев, Л.Н. Сапегин, «Новые возможности биометрических голосовых технологий», *Защита информации. Конфидент*, № 5, С. 34-39, 2003.

45. В.И. Волчихин, А.И. Иванов, «Использование тайных биометрических образов человека», *Системы безопасности*, №2, С. 40-41, 2002.

46. А.И. Иванов, «Оценка эффекта от использования тайных биометрических образов», *Защита информации. Конфидент*, № 4-5, С. 128-131, 2002.

47. В.В. Задорожный, «Обзор биометрических технологий», *Защита*

информации. Конфидент, № 5, С. 26-29, 2003.

48. Y. Dodis, L. Reyzin, A. Smith, «Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data», *Proceedings from Advances in Cryptology. EuroCrypt*, 2004.

49. С.П. Расторгуев, *Программные методы защиты информации: Учебное пособие*, Пенза: Издательство Пензенского государственного университета, 2000, С. 95.

50. Р.А. Васильев, *Биометрическая идентификация пользователей информационных систем на основе кластерной модели элементарных речевых единиц*, Изд-во МИФИ, 2016, С. 153.

51. А.И. Иванов, *Биометрическая идентификация личности по динамике подсознательных движений*, Пенза: Изд-во Пенз. гос. ун-та, 2000, С. 188.

52. А.И. Иванов, *Нейросетевая защита конфиденциальных биометрических образов гражданина и его личных криптографических ключей: монография*, Пенза, 2014, С. 57.

53. А.И. Иванов, *Многомерная нейросетевая обработка биометрических данных с программным воспроизведением эффектов квантовой суперпозиции. Монография*, Пенза: Издательство ОА «ПНИЭИ», 2016, С. 133.

54. А.И. Иванов, О.В. Ефимов, В.А. Фунтиков, «Оценка усиления стойкости коротких цифровых паролей (PIN кодов) при их рукописном воспроизведении», *Защита информации. INSIDE*, № 1, С. 55-57, 2006.

55. В.И. Волчихин, А.Ю. Малыгин, Ю.И. Олейник, «Отличить "своего" от "чужого". Проблемы развития и тестирования высоконадежной биометрии», *Системы безопасности – межотраслевой каталог*, С. 164-168, 2006.

56. А.Л. Горелик, В.А. Скрипкин, *Методы распознавания*, М.: Высшая школа, 1984, С. 80.

57. А.И. Иванов, *Нейросетевые алгоритмы биометрической идентификации личности*, М.: Радиотехника, 2004, С. 143.

58. А.В. Еременко, А.Е. Сулавко, «Исследование алгоритма генерации криптографических ключей из биометрической информации пользователей компьютерных систем», *Информационные технологии*, № 11. С. 47–51, 2013.

59. А.Е. Сулавко, А.В. Еременко, «Метод сжатия собственных областей классов образов в пространстве малоинформативных признаков», *Искусственный интеллект и принятие решений*, № 2, С. 95–102, 2014.

60. А.Е. Сулавко, А.В. Еременко, А.Е. Самотуга, «Исключение искаженных биометрических данных из эталона субъекта в системах идентификации», *Информационные технологии и вычислительные системы*, № 3. С. 96–101, 2013.

61. П.Н. Корнюшин, «Создание системы аутентификации на основе клавиатурного почерка пользователей с использованием процедуры генерации ключевых последовательностей из нечетких данных», *Интеллектуальные технологии в образовании, экономике и управлении – 2007: IV Междунар. науч.-практ. конф.*, Воронеж, 2007.

62. А. Вакуленко, А. Юхин, *Биометрические методы идентификации личности: обоснованный выбор и внедрение*, М.: Наука, 2007, С. 224.

63. А.И. Зиятдинов, *Принципы построения систем биометрической аутентификации*, М.: МФТИ, 2005, С. 188.

64. Е.А. Харин, «Генерация ключевой информации на основе биометрических данных пользователей», *XIV международная научная студенческая конференция*, Новосибирск, 2007, С. 181–187.

65. П.А. Бондарев, М.С. Астафьев, «Распознавание отпечатков пальцев методами, использующими нейросети». [Электронный ресурс]. Режим доступа: http://neurnews.iu4.bmstu.ru/book/it/it898/stat1.htm. Дата обращения: 20.01.2004.

66. А.А. Демин, *Электронная пропись: монографія*, Саарбрюккен: Ламберт Академик Паблишинг, 2009, С. 80.

67. А.Б. Мерков, «Основные методы, применяемые для распознавания рукописного текста». [Электронный ресурс]. Режим доступа: http://www.recognition.mccme.ru/pub/RecognitionLab.html/methods.html. Дата обращения: 25.05.2018.

68. C.P. Pfleeger, «Security computing», *IEEE Computer Society*, 1996.

69. N.K., J.H. Ratha Connell, R.M. Bolle, «Enhancing security and privacy in biometrics-based authentication systems», *IBM Systems Journal 40*, pp. 614–634, 2001.

70. А.Н. Савинов, И.Г. Сидоркина, В.И. Иванов, «Анализ решения проблемы использования клавиатурного почерка для обеспечения безопасности ключевой системы предприятия», *Конгресс по 46 интеллектуальным системам и информационным технологиям «IS&IT'11»*, Москва, Т.3, С. 40-47, 2011.

71. Р.Р. Шарипов, «Технология биометрической защиты информации на основе клавиатурного почерка», *X туполевские чтения. Всероссийская молодёжная научная конференція*, Казань, 2002, С. 137.

72. В.П. Широчин, А.В. Кулик, В.В. Марченко, «Динамическая аутентификация на основе анализа клавиатурного почерка», *Вестник Национального технического университета Украины «Информатика, управление и вычислительная техника»*, № 32, С. 3–16, 1999.

73. Д.Е. Ян, К.В. Анисимович, А.Л. Шамис, *Новая технология распознавания символов. Теория, практическая реализация, перспективы*, М.: Препринт, 1995, С. 36.

74. Н.Д. Горский, В.А. Анисимов, Л.М. Горская, *Распознавание рукописного текста: от теории к практике*, СПб.: Политехника, 1997, С. 126.

75. В.Л. Арлазаров, О.А.Славин, «Алгоритмы распознавания и технологии ввода текстов в ЭВМ», *Информационные технологии и вычислительные системы*, №1, С. 22–27, 1996.

76. «Программа распознавания образов и прогноза». [Электронный ресурс]. Режим доступа: http://nnet.chat.ru.nbp.html. Дата обращения: 20.01.2004.

77. *Ward Systems Group, Inc. и компания НейроПроект*. [Электронный ресурс]. Режим доступа: http:// www.neuroproject.ru/index.htm. Дата обращения: 25.05.2010.

78. «Система определения настроения». [Электронный ресурс]. Режим доступа: http://www.neuroproject.ru. Дата обращения: 25.05.2010.

79. «Распознавание способа печати». [Электронный ресурс]. Режим доступа: http://www.neuroproject.ru. Дата обращения: 25.05.2010.

80. Л.Э. Чалая, «Модель идентификации пользователей по клавиатурному почерку», *Искусственный интеллект*, № 4, С. 811-817, 2004.

81. J. Leggett, G. Williams, «Verifying identity via keystroke characteristics», *International Journal of Man-Machine Studies*, vol. 28, no. 1, pp. 67-76, 1988.

82. J. Leggett, G. Williams, M. Usnick, M. Longnecker, «Dynamic identity verification via keystroke characteristics», *International Journal of Man-Machine Studies*, vol. 35, no. 6, pp. 859-870, 1991.

83. O. Coltell, J.M. Badia, G. Torres, «Biometric Identification System Based in Keyboard Filtering», *Proc. of XXXIII Annual IEEE International Carnahan Conference on Security Technology*, 1999, pp. 203-209.

84. J. Kittler, M. Hatef, R.P.W. Duin, J. Matas, «On Combining Classifiers», *IEEE Trans. Pattern Anal. Mach.Intell., IEEE Computer Society*, vol. 20, pp. 226-239, 1998.

85. S. Bleha, C. Slivinsky, B. Hussein, «Computer-Access Security Systems Using Keystroke Dynamics», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-12, no. 12, pp. 1217-1222, 1990.

86. M. Brown, S.M. Rogers, «User Identification via Keystroke Characteristics of

Typed Names using Neural Networks», *International Journal of Man-Machine Studies*, vol. 39, no. 6, pp. 999-1014, 1993.

87. W.G. De Ru, J.H.P. Eloff , «Enhanced Password Authentication through Fuzzy Logic», *IEEE Expert/Intelligent Systems & Their Applications*, vol. 12, no. 6, 1997.

88. F. Deane, R. Henderson et al., «Theoretical Examination of the Effects of Anxiety and Electronic Performance Monitoring on Behavioural Biometric Security Systems», *Interacting with Computers*, vol. 7, no. 4, pp. 395-411, 1995.

89. G.K. Gupta, R. Joyce, *User Authorisation Based on Keystroke Latencies*, JCU-CS-89/5, Dpt. of Computer Science, James Cook University, 1989.

90. P. Haunold, W. Kuhn, «A Keystroke Level Analysis of Manual Map Digitizing», *Lecture Notes in Computer Science*, vol. 716, pp. 406-420, 1993.

91. R. Henderson, D. Mahar et al., «Electronic Monitoring Systems: An Examination of Physiological Activity and Task Performance within a Simulated Keystroke Security and Electronic Performance Monitoring System», *International Journal of Human-Computer Studies*, vol. 48, no. 2, pp. 143-157, 1998.

92. R. Joyce, G. Gupta, «Identity authentication based on keystroke latencies», *Communications of the ACM*, vol. 33, no. 2, pp. 168-176, 1990.

93. R. Joyce, G. Gupta, «User Authorization Based on Keystroke Latencies», *Communications of the ACM*, *CACM*, vol. 33, no. 2, 1990.

94. H. Kim, «Biometrics, Is it a Viable Proposition for Identity Authentication on Access Control», *Computers and Security*, vol. 14, no. 3, pp. 205-214, 1995.

95. L. Labuschagne, J.H.P. Eloff, «Improving system-access control using complementary technologies», *Computers & Security*, vol. 15, no. 6, pp. 543-550, 1997.

96. L. Labuschagne, J.H.P. Eloff, W.G. De Ru, *Practical Considerations for Access Control*, Dpt. Computer Science. Rand Afrikaans Univ., 1996.

97. Н.З. Сафиуллин, «Усилие нажатия – как параметр клавиатурного почерка», *Развитие технологий радиоэлектроники и телекоммуникаций. Региональная*

научно-техническая конференция, посвящённый 10-летию организации кафедры технологии радиоэлектронных средств, Казань, 2004, С. 98–99.

98. D. Umphress, G. Williams, «Identity Verification Through Keyboard Characteristics», *International Journal of Man-Machine Studies*, vol. 23, no. 3, pp. 263-273, 1985.

99. O. Vysotska, A. Davydenko, «Keystroke Pattern Authentication of Computer Systems Users as One of the Steps of Multifactor Authentication», *Advances in Computer Science for Engineering and Education II. Advances in Intelligent Systems and Computing*, vol 938, pp. 356-368, 2019.

100. Е.А. Высоцкая, А.Н. Давиденко, «Количественный и качественный анализ учебных данных с целью повышения эффективности аутентификации пользователей компьютерной системы при помощи нейронных сетей», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 24, С. 110-116, 2003.

101. Е.А. Высоцкая, «Компьютерное моделирование задач аутентификации пользователя компьютерных систем с помощью вероятностных нейронных сетей», *Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г. Є. Пухова*, Вип. 24, С. 3-9, 2004.

102. Е. Высоцкая, А. Давиденко, «Исследование эффективности применения вероятностных нейронных сетей для решения задачи аутентификации пользователя компьютерных систем», *Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні. Наук.-техн. зб.*, Вип. 9, С. 103-110, 2004.

103. Е.А. Высоцкая, «Исключение учебных данных с грубыми ошибками, как один из способов повышения эффективности применения вероятностных нейронных сетей для аутентификации пользователей компьютерных систем по клавиатурному почерку», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 29, С. 52-59, 2005.

104. Е.А. Высоцкая, «Выбор анализируемых параметров при аутентификации пользователей компьютерных систем по клавиатурному почерку при помощи вероятностной нейронной сети», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 30, С. 45-52, 2005.

105. Е.А. Высоцкая, «Влияние исключения учебных данных с грубыми ошибками на зависимость эффективности применения вероятностных нейронных сетей для аутентификации пользователей компьютерных систем по клавиатурному почерку от различных параметров», *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН Украіни*, Вип. 28, С. 3-10, 2005.

106.Е.А. Высоцкая, «Влияние параметров учебных данных на качество аутентификации при помощи вероятностной нейронной сети», *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН Украіни*, Вип. 32, С. 10-17, 2006.

107. О.О. Висоцька, «Моніторинг роботи користувачів комп'ютерних систем за допомогою технологій розпізнавання за клавіатурним почерком», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 84, С. 119-125, 2018.

108. Е.А. Высоцкая, «Влияние параметров учебных данных на качество аутентификации пользователей компьютерных систем», *Моделювання: XXIV Науково-технічна конференція*, Київ, 2005, С. 3.

109.А.М.Давиденко, С.Я.Гільгурт, О.О.Висоцька, А.А.Кочурков, Ю.О.Чернова, «Експериментальне дослідження програми для автентифікації користувачів комп'ютерної системи за клавіатурним почерком за допомогою імовірнісної нейронної мереж», *Проблемы информатики и моделирования: пятая междунар. научно-технич. конф.*, Харьков, 2005, С. 24, 66-69.

110. А.Н. Давиденко, Е.А. Высоцкая, «Использование биометрических систем защиты для уменьшения риска возникновения чрезвычайных ситуаций»,

Декларування безпеки об'єктів підвищеної небезпеки як засіб регулювання безпеки регіону (держави): науково-методич. семінар у рамках IV Міжнародного виставкового форуму «Технології захисту – 2007», Київ, 2007, С. 123-126.

111. Е.А. Высоцкая, «Аутентификация пользователей компьютерных систем по клавиатурному почерку при помощи вероятностной нейронной сети», *Моделювання: XXIX Науково-технічна конф.*, Київ, 2010, С. 10.

112. О.О.Висоцька, «Використання біометричних технологій розпізнавання в системах моніторингу роботи користувачів комп'ютерних систем», *Проблеми створення, розвитку та застосування інформаційних систем спеціального призначення: 18-а науково-практична конф.*, Житомир, 2011, С.219-220.

113. О.О. Висоцька, «Підвищення рівня безпеки комп'ютерних систем за допомогою біометричної автентифікації», *Проблеми створення, розвитку та застосування високотехнологічних систем спеціального призначення: XX Всеукраїнська науково-практична конференція*, Житомир, 2014, С. 190-191.

114.А.М. Давиденко, О.О. Висоцька, «Визначення функції моніторингу стану санкціонованих користувачів комп'ютерних систем за допомогою аналізу їх клавіатурного почерку», *Комп'ютерні системи та мережні технології" (CSNT-2019): XII Міжнародна науково-практична конф.*, Київ, 2019, С.41-42.

115.А.М. Давиденко, О.О. Висоцька, «Моніторинг функціонального стану представників критичних професій, за допомогою аналізу їх клавіатурного почерку», *Актуальні проблеми управління інформаційною безпекою держави: X Всеукраїнська науково-практична конф.*, Київ, 2019, С.201-203.

116.Е.А. Высоцкая, А.Н. Давиденко, «Анализ технологии предварительной обработки данных при аутентификации пользователей компьютерных систем по клавиатурному и рукописному почеркам», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 55, С. 34-41, 2010.

117. O. Vysotska, A. Davydenko, «Authentication of information systems users,

based on the analysis of their handwriting», *Scientific and Practical Cyber Security Journal (SPCSJ)*, *vol.* 2, no. 4, pp. 51-63, 2018.

118. О. Корченко, А. Давиденко, О. Висоцька, «Метод автентифікації користувачів інформаційних систем за їх рукописним почерком з багатокроковою корекцією первинних даних», *Захист інформації*, Том 21, №1, С. 40-51, 2019.

119. Е.А. Высоцкая, «Задача распознавания написанного ключевого слова, как одна из задач, решаемых при выполнении аутентификации пользователей компьютерных систем по рукописному почерку», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 36, С. 67-76, 2006.

120. Е.А. Высоцкая, «Выбор анализируемых характеристик при аутентификации пользователей компьютерных систем по рукописному почерку на разных этапах развития вычислительной техники», *Моделювання та інформаційні технології. Зб. наук. праць*, Вип. 56, С. 31-39, 2010.

121. Е.А. Высоцкая, «Метод проведения аутентификации пользователей компьютерных систем по рукописному почерку», *Моделювання: науково-технічна конф. молодих вчених і спеціалістів,* Київ, 2006, С. 9-10.

122. O. Vysotska, A. Davydenko, «The usage of handwriting recognition systems of information systems users for their authentication», *La science et la technologie à l'ère de la société de l'information: conférence scientifique et pratique internationale*, Bordeaux, France, 2019, vol. 9, pp. 48-51.

123. О. Висоцька, А. Давиденко, В. Щербина, «Формалізація процедури аналізу рукописного почерку людини для організації розмежування доступу до інформаційних систем», *ITSec: Безпека інформаційних технологій: IX Міжнародна науково-технічна конф.*, Київ, 2019, С.22-23.

124. Е.С. Вентцель, *Теория вероятностей*, М.: КНОРУС, 2010, С. 664.

125. А.Г. Курош, *Курс высшей алгебры*, М.: Государственное издательство физико-математической литературы, 1963, С. 432.

126. Р. Каллан, *Основные концепции нейронных сетей.: Пер. с англ.*, М.: Издательский дом «Вильямс», 2001, С. 290.

127. В.И. Архангельский и др., *Нейронные сети в системах автоматизации*, К.: «Техника», 1999, С. 364.

128. А.Г. Корченко, И.А. Терейковкий, Н.П. Карпинский, С.Т. Тнымбаев, *Нейросетевые модели, методы и средства оценки параметров безопасности и Интернет-ориентированных информационных систем*, К.: ТОВ «Наш Формат», 2016, С. 276.

129. О.Г. Корченко, І.А. Терейковський, А.О. Білощинський, *Методологія розроблення нейромережевих засобів інформаційної безпеки Інтернет-орінєтованих інформаційних систем*, К.: ТОВ «Наш Формат», 2016, С. 249.

130. И. Терейковский, А. Корченко, П. Викулов, А. Шаховал, «Модели эталонов лингвистических переменных для обнаружения сниффинг-атак», *Захист інформації*, Т.19, №3, С. 228-242, 2017.

131. І. Терейковський, А. Корченко, П. Вікулов, І. Ірейфідж, «Моделі еталонів лінгвістичних змінних для систем виявлення email-спуфінг-атак», *Безпека інформації*. Т.24, №2, С. 99-109, 2018.

132. Терейковский И.А., Терейковская Л.А., Корченко А.О., Ахметов Б.Б., Алібієва Ж.М., «Нейросетевое распознавание рукописных символов в системе биометрической аутентификации», *Інформаційні технології в економіці і природокористуванні*, №2, С. 29-44, 2017.

133. І.А. Терейковський, «Вдосконалення алгоритму навчання багатошарового персептрону, призначеного для розпізнавання мережевих атак», *Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні*, Вип. 2(24), С. 65-70, 2012.

134. І.А. Терейковський, «Вдосконалення методики захисту інформації в корпоративних мережах, що використовують ресурси Internet», *Вісник*

національного транспортного університету, № 8, С. 13-16, 2003.

135. І.А. Терейковський, «Визначення оптимального методу контролю об'єктів захисту комп'ютерних мереж», *Вісник КНУТД*, № 5, С. 39-44, 2006.

136. І.А. Терейковський, «Визначення оптимального типу нейронної мережі, призначеної для використання в програмних засобах захисту інформації», *Сучасні тенденції розвитку технологій в інфокомунікаціях та освіті: VIII наук. конф.*, Київ, 2011, С. 372-379.

137. І.А. Терейковський, «Використання нейронної мережі з радіальними базисними функціями в задачах діагностики стану захищеності програмного забезпечення», *Науково-технічний збірник «Управління розвитком складних систем» Київського національного університету будівництва і архітектури*, Вип. 3, С. 111-114, 2010.

138. І.А. Терейковський, «Використання семантичної нейронної мережі в задачах моніторингу текстової інформації», *Вісник ДУІКТ*, № 1, Т.10, С. 36-41, 2012.

139. І. Терейковський, *Нейронні мережі в засобах захисту комп'ютерної інформації: монографія*, К.: ПоліграфКонсалтинг, 2007, С. 209.

140. І.А. Терейковський, «Оптимізація архітектури нейронної мережі, призначеної для діагностики стану комп'ютерної мереж», *Науково-технічний збірник «Управління розвитком складних систем» Київського національного університету будівництва і архітектури*, Вип. 6, С. 155-158, 2011.

141. І. Терейковський «Оптимізація захисту відкритих корпоративних мереж», *Вісник КНТЕУ*, № 1, С. 103-112, 2004.

142. С. Хайкин, *Нейронные сети*, ТОВ «Диалектика-Вильям», 2018, С. 1104.

143. В.В. Круглов, *Нечеткая логика и искусственные нейронные сети: учеб. пособие*, М.: Физматлит, 2001, С. 224.

144. Т.Н. Байдык, *Нейронные сети и задачи искусственного интеллекта*, К.: «Наукова думка», 2001, С. 265.

145. Тарик Рашид, *Создаем нейронную сеть*, ТОВ «Вильям», 2016, С. 272.

146. Я. Гудфеллоу, И. Бенджио, А. Курвилль, *Глубокое обучение*, The MIT Press, 2016, С. 800.

147. Т.Г. Визель, *Основы нейропсихологии: учеб. для студентов вузов*, М.: АСТАстрель Транзиткнига, 2005, С. 384.

148. С. Николенко, А. Кадурин, Е. Архангельская, *Глубокое обучение. Погружение в мир нейронных сетей*, ТОВ «Питер», 2012, С. 480.

149. В. Редько, *Эволюция, нейронные сети, интеллект. Модели и концепции эволюционной кібернетики*, ТОВ «Ленанд», 2016, С. 224.

150. Р. Каллан, *Нейронные сети. Краткий справочник*, ТОВ «Вильям», 2012, С. 288.

151. В.В. Круглов, В.В. Борисов, *Искусственные нейронные сети: Теория и практика*, Горячая линия-Телеком, 2002, С. 382.

152. Ф. Розенблатт, *Принципы нейродинамики. Перцептроны и теория механизмов мозга*, М.: Мир, 1965, С. 480.

153. Martin D. Buhmann, *Radial Basis Functions: Theory and Implementations*. Cambridge University Press., 2003.

154. «Цепь Маркова – это просто: подробно разбираем принцип». [Електронний ресурс]. Режим доступу: https://proglib.io/p/markov-chain. Дата обращения: 14.04.2019.

155. «Персептрон». [Електронний ресурс]. Режим доступу: http://www. machinelearning.ru/wiki/index.php?title=Персептрон. Дата обращения: 14.04.2019.

156. «Сеть Хопфилда». [Електронний ресурс]. Режим доступу: http://apsheronsk. bozo.ru/Neural/Lec6.htm. Дата обращения: 14.04.2019.

157. «Машина Больцмана». [Електронний ресурс]. Режим доступу: https://studopedia.org/1-12520.html. Дата обращения: 14.04.2019.

158. M. Carreira-Perpignan, G. Hinton, «On contrastive divergence learning», *In Proc. 11th Workshop on Artificial Intelligence and Statistics*, 2005.

159. Geoffrey Hinton, «A Practical Guide to Training Restricted Boltzmann Machines», *Department of Computer Science, Toronto University of Toronto,* 2010.

160. В.А. Крисилов, К.В. Чумичкин, А.В. Кондратюк, «Представление исходных данных в задачах нейросетевого прогнозирования», *Нейроинформатика*, Ч. 1, С. 184–191, 2003.

161. В.С. Медведев, В.Г. Потемкин, *Нейронные сети. MATLAB 6*, М.: ДИАЛОГ-МИФИ, 2002, С. 496.

162. А.Б. Барский, *Нейронные сети: распознавание, управление, принятие решений*, М.: Финансы и статистика, 2004, С. 176.

163. Г.К. Вороновский, К.В. Махотило, С.А. Сергеев, *Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности*, Харьков: Основа, 1997, С. 112.

164. А. И. Галушкин, *Теория нейронных сетей*, М.: ИПРЖР, 2000, С. 416.

165. А.Ф. Гареев, «Применение вероятностной нейронной сети для автоматического рубрицирования текстов», *Нейроинформатика-99: науч. конф.*, Москва, 1999, С. 71–79.

166. М.М. Глибовець, О.В. Олецький, *Штучний інтелект*, К.: Києво-Могилян. акад., 2002, С. 366.

167. В.А. Головко, *Нейронные сети: обучение, организация и применение*, М.: ИПРЖР, 2001, С. 256.

168. А.Н. Горбань, Д.А. Россиев, *Нейронные сети на персональном компьютере*, Новосибирск: Наука, 1996, С. 276.

169. А.Н. Горбань, *Обучение нейронных сетей*, М.: ParaGraph, 1990, С. 160.

170. А.Ю. Дорогов, «Структурный синтез быстрых нейронных сетей», *Нейрокомпьютер*, № 1, С. 11–24, 1999.

171. А.Ю. Дорогов «Структурный синтез двухслойных быстрых нейронных сетей», *Кибернетика и системный аналіз*, № 4, С. 47–56, 2000.

172. А.Ю. Дорогов, А.А. Алексеев, «Быстрые нейронные сети», *Пятьдесят лет развития кибернетики: междунар. научн.-техн. конф.*, Санкт-Петербург, 1999, С. 120–121.

173. А.Ю. Дорогов, А.А. Алексеев, «Категории ядерных нейронных сетей», *Нейроинформатика-99: науч. конф.*, Москва, 1999, С. 55–64.

174. З.В. Дударь, Д.Е. Шуклин «Реализация нейронов в семантических нейронных сетях», *Радиоэлектроника и информатика*, № 4, С. 89–96, 2000.

175. Франсуа Шолле, *Глубокое обучение на Python*, ТОВ «Питер», 2016, С. 400.

176. Орельен Жерон, *Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow*, ТОВ «Диалектика-Вильям», 2018, С. 688.

177. З.Ю. Кочладзе, А.Л. Оганезов, «Использование двухслойной нейронной модели в процессах распознавания образов», *Научно-технический журнал «Энергия»* №4 (36), С. 78, 2005.

178. Р.А. Чоговадзе, «Синтез искусственных нейронных сетей для процессов распознавания образов», *Труды Грузинского технического университета*, №4 (454), 2004.

179. Н.М. Амосова, *Нейрокомпьютеры и интеллектуальные роботы*, Киев, 1991.

180. Ю.В. Чернухин, *Микропроцессорное и нейрокомпьютерное управление адаптивными мобильными роботами*, Таганрог, 1993.

181. Ю.В. Чернухин, *Нейропроцессоры*, Таганрог, 1994.

182. Ф. Уоссермен, *Нейрокомпьютерная техника: Теория и практика*, Москва, 1992.

183. В.Д. Цыганков, *Нейрокомпьютер и его применение*, Москва: СолСистем, 1993.

184. Е.Н. Соколов, Г.Г. Вайткявичус, *Нейроинтеллект от нейрона к нейрокомпьютеру*, Москва: Наука, 1989.

185. Л.И. Волгин, *Комплиментарная алгебра нейросетей*, Таллин: АО KLTK, 1993.

186. А.Н. Горошкин, «Обработка изображений в системах распознавания рукописного текста», *10-ая международная конференция и выставка «Цифровая обработка сигналов и ее применение»*, Москва, 2008, 120 с.

187. Д.Н. Олешко, В.А. Крисилов, А.А. Блажко «Построение качественной обучающей выборки для прогнозирующих нейросетевых моделей» *Штучний інтелект*, № 3, С. 567–573, 2004.

188. Э. Хант, *Искусственный интеллект*, М.: Мир, 1978, С. 560.

189. A. Rosebrock, *Deep Learning for Computer Vision with Python. Practitioner Bundle*, PyImageSearch, 2017, С. 210.

190. Andriy Burkov, *The Hundred-Page Machine Learning Book*, Andriy Burkov, 2019, С. 160.

191. Maxim Lapan, *Deep Reinforcement Learning Hands-On: Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo Zero and more*, Packt Publishing, 2018, С. 546.

192. А.А. Демин, «Обзор интеллектуальных систем для оценки каллиграфии», *Инженерный вестник: электронный научно-технический журнал*, №9, С. 1–25, 2012.

193. А.А. Демин, «Интеллектуальная интерактивная обучающая система "Электронная пропись"», *Наукоемкие технологии и интеллектуальные системы 2007: 9-ая Молодежная научно-техническая конференція*, Москва, 2007, С. 152–154.

194. В.А. Ковалевский, *Методы оптимальных решений в распознавании изображений*, М.: Наука, 1967, С. 328.

195. И. Гайдышев, *Анализ и обработка данных: специальный справочник*, СПб.: Питер, 2001, С. 750.

196. К. Фукунага, *Введение в статистическую теорию распознавания образов.: Пер. с англ.*, М.: Наука, 1979, С. 368.

197. Р. Гонсалес, Дж. Ту, *Принципы распознавания образов*, М.: Мир, 1978, С. 414.

198. Ю.И. Журавлев, И.Б. Гуревич, «Распознавание образов и распознавание изображений», *Распознавание, классификация, прогноз*, Т. 2, С. 5–73, 1989.

199. А.Я. Архангельский, *Программирование в C++ Builder 6*, Москва: Издательство Бином, 2003, С. 1140.

200. А.Я. Архангельский, *Программирование в Delphi 7*, Москва: Издательство Бином, 2003, С. 1152.

Додаток А. Документи, що підтверджують впровадження результатів дисертаційної роботи

АКТ
про впровадження результатів дисертаційної роботи
Висоцької Олени Олександрівни по темі "Побудова інформаційних технологій біометричної автентифікації користувачів автоматизованих систем на основі нейронних мереж"

Даний акт підтверджує, що в Управлінні верифікації Генерального штабу Збройних сил України при розробці системи захисту інформації в автоматизованих систем управління, а саме для організації процесу автентифікації користувачів, використовувались результати дисертаційної роботи Висоцької О.О., а саме технологія автентифікації користувачів автоматизованих систем за їх клавіатурним почерком.

Начальник відділу
полковник Гудзь А.М.

Начальник Управління верифікації
Генерального штабу Збройних сил України
генерал – майор Федотов І.М.

«25» 03 2010р

ЗАТВЕРДЖУЮ
Заступник начальника
управління ПФУ у Києво-
Святошинському районі
_____Вітер С.І.
27 травня 2010р.

Акт

про впровадження результатів дисертаційної роботи Висоцької Олени Олександрівни по темі "Побудова інформаційних технологій біометричної автентифікації користувачів автоматизованих систем на основі нейронних мереж"

Даний акт підтверджує, що в відділі обліку надходження платежів управління ПФУ у Києво-Святошинському районі, для організації процесу автентифікації користувачів, при побудові системи захисту інформації в автоматизованих систем управління ПФУ у Києво-Святошинському районі , використовувались результати дисертаційної роботи Висоцької О.О., а саме технологія автентифікації користувачів автоматизованих систем за їх рукописним почерком . Розробка системи захисту інформації проводилась в межах програми "Захист інформації у системі ПФУ", згідно наказу "Про забезпечення захисту інформації в управлінні" № 42 від 05.05.2010р.

Заступник начальника відділу
обліку надходження платежів Демура А.І.

ЗАТВЕРДЖУЮ
Кривошей А.М.
ТОВ «Інтегратор»

24 12 2013р.

Акт

про впровадження результатів дисертаційної роботи Висоцької Олени Олександрівни по темі "Побудова інформаційних технологій біометричної автентифікації користувачів автоматизованих систем на основі нейронних мереж"

Даний акт підтверджує, що у підприємстві «Інтегратор» при розробці підсистеми захисту інформації в автоматизованих систем "Фортеця", а саме для організації в системі процесу автентифікації користувачів, використовувались результати дисертаційної роботи Висоцької О.О., а саме технологія автентифікації користувачів автоматизованих систем за їх клавіатурним почерком за допомогою імовірнісної нейронної мережі. Розробка системи захисту інформації "Фортеця" проводилась в межах програми "Забезпечення програми інтегрованої системи доступу та охоронної сигналізації", зареєстрованої у Державному департаменті інтелектуальної власності №7306 від 20.03.2003.

Директор ТОВ «Інтегратор» Кривошей А.М.

ЗАТВЕРДЖУЮ
Директор
ТОВ "НВЦ"ІНФОЗАХИСТ"
_____СігнаєвськийК.М.
12 2018р.

Акт

про впровадження результатів дисертаційної роботи Висоцької Олени Олександрівни по темі "Методи біометричної автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком"

Даний акт підтверджує, що в ТОВ "НВЦ"ІНФОЗАХИСТ" при розробці інформаційно-аналітичної системи, а саме для організації процесу автентифікації користувачів використовувались результати дисертаційної роботи Висоцької О.О., а саме метод біометричної автентифікації користувачів інформаційних систем за їх клавіатурним почерком. Розробка інформаційно-аналітичної системи «Контур» проводиться в межах Державного оборонного замовлення.

Технічний директор Калінін Я.В.

ЗАТВЕРДЖУЮ
Директор Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України
чл.-кор. НАН України, д-р техн. наук, професор
_____ В. В. МОХОР
11.01. 2019 р.

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОБЛЕМ МОДЕЛЮВАННЯ В ЕНЕРГЕТИЦІ ім. Г.Є.Пухова
Ідентифікаційний код 05516949
м.Київ

Акт

про впровадження результатів дисертаційної роботи Висоцької Олени Олександрівни по темі "Методи біометричної автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком"

Даний акт підтверджує, що у відділі №2 ІПМЕ ім. Г.Є. Пухова НАН України, при розробці НДР «МОД-Д» «Дослідження та розробка методів оцінювання захищеності інформації в розподілених високопродуктивних інформаційних системах при вирішенні задач енергетики» № 0114U002361 (2014р.-2018р.), а саме для розробки системи біометричної автентифікації користувачів, використовувались результати дисертаційної роботи Висоцької О.О., а саме метод біометричної автентифікації користувачів інформаційних систем за їх клавіатурним почерком. Розробка НДР «МОД-Д» виконувалась у відповідності з планом науково-дослідних робіт ІПМЕ ім. Г.Є. Пухова НАН України.

В.о. начальника відділу № 2
к.т.н., с.н.с. Гільгурт С.Я.

УЗГОДЖЕНО
Проректор Національного авіаційного університету з навчальної роботи
_____ А. Г. Гудманян
«___» _____ 2019 р.

ЗАТВЕРДЖУЮ
Проректор Національного авіаційного університету з наукової роботи
_____ В. П. Харченко
«___» _____ 2019 р.

АКТ ВПРОВАДЖЕННЯ
результатів дисертаційної роботи асистента кафедри КСЗІ Висоцької О.О. в навчальний процес Національного авіаційного університету

Ми, що нижче підписалися, *завідувач кафедри комп'ютеризованих систем захисту інформації С.В. Казмірчук, вчений секретар кафедри комп'ютеризованих систем захисту інформації О.О.Мелешко*
склали цей акт про те, що результати наукових досліджень *за темою кандидатської дисертаційної роботи Висоцької Олени Олександрівни «Методи біометричної автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком»* використовуються у навчальному процесі *Факультету кібербезпеки, комп'ютерної та програмної інженерії НАУ на кафедрі комп'ютеризованих систем захисту інформації*

Найменування впровадженого результату	Форма впровадження і досягнутий фактичний ефект
Методи автентифікації користувачів інформаційних систем за їх клавіатурним та рукописним почерком, з використанням розроблених методів первинної обробки зразків почерку	Матеріали дисертаційної роботи впроваджені у вигляді лабораторних та розрахунково-графічних робіт з дисциплін «Безпека інформаційно-комунікаційних систем та мереж», «Інформаційно-аналітичні системи» та «Методи побудови та аналізу криптосистем». Впровадження вказаних методично-навчальних матеріалів дозволило підвищити рівень вивчення студентами методів біометричної автентифікації користувачів інформаційних систем, акцентуючи при цьому увагу на необхідність первинної обробки зразків біометричних характеристик, та привело до використання методів класифікації об'єктів для вирішення задач захисту інформації.

Завідувач кафедри
комп'ютеризованих систем захисту інформації _____ С.В. Казмірчук

Вчений секретар кафедри
комп'ютеризованих систем захисту інформації _____ О.О.Мелешко

Додаток Б. Система автентифікації користувачів інформаційних систем за клавіатурним почерком «АКП»

Система «АКП» розроблена на мові C++ Builder з використанням, для обробки та зберігання інформації, програми Database Desktop (програма для роботи з базами даних) та SQL-запитів [199]. Програма працює під керуванням ОС Windows. Для формування та динамічного передавання характеристик КП користувача в комп'ютер використовується клавіатура.

Початкові коди програми розміщені в наступних основних файлах:

- файл формування проекта: Project1.bpr;
- 15 файлів форм: main.dfm, access.dfm, analiz.dfm, error.dfm, nastr.dfm, net.dfm, net_otch.dfm, net_test.dfm, new_nab.dfm, print.dfm, prizn.dfm, read.dfm, scor.dfm, tabl.dfm, test.dfm;
- 16 файлів реалізацій модулів: Project1.cpp, main.cpp, access.cpp, analiz.cpp, error.cpp, nastr.cpp, net.cpp, net_otch.cpp, net_test.cpp, new_nab.cpp, print.cpp, prizn.cpp, read.cpp, scor.cpp, tabl.cpp, test.cpp;
- 15 заголовочних файлів модулів: main.h, access.h, analiz.h, error.h, nastr.h, net.h, net_otch.h, net_test.h, new_nab.h, print.h, prizn.h, read.h, scor.h, tabl.h, test.h.

Для зберігання БДНЗ, наборів КФ та результатів роботи системи «АКП» *використовуються таблиці, які знаходяться в наступних 9 файлах, в окремій папці*: main.db, main_boch.db, main_print_i.db, main_promeg.db, nabor.db, neizv_ekz.db, scor.db, uch_dan.db, vyborka.db.

Призначення основних файлів реалізацій модулів:

Project1.cpp – містить код головної функції WinMain, яка ініціалізує додаток та запускає його на виконання.

main.cpp – містить код реалізації модуля головної (батьківської) форми проекта. З цієї форми викликаються всі інші (дочірні) форми.

access.cpp – містить код реалізацій дочірнього модуля (КРДМ), в якому виконується введення пароля доступу до даної системи, його перевірка і надання відповідних прав користувачу системи «АКП».

analiz.cpp – містить КРДМ, в якому виконується аналіз накопичених НЗ в БДНЗ, а саме аналіз залежності значення часу, необхідного для натискання клавіши, від самої клавіши.

error.cpp – містить КРДМ, який викликається при виконанні користувачем якихось помилкових дій і вказує в чому саме помилка.

nastr.cpp – містить КРДМ, в якому виконується налаштування параметрів системи, таких як: номера активного та робочого набору КФ; кількість слів в наборі та кількість необхідних повторень введення кожного слова під час тестування; список слів в наборі.

net.cpp – містить КРДМ, в якому виконується вибір параметрів побудови ІНМ; побудова даної мережі для автентифікації користувачів; тестування роботи розробленої системи розпізнавання.

net_otch.cpp – містить КРДМ, в якому виконується перегляд звіту результатів роботи побудованої нейронної мережі (результатів розпізнавання).

net_test.cpp – містить КРДМ, який викликається під час тестування побудованої нейронної мережі, після розпізнавання користувачів, для відображення результатів розпізнавання.

new_nab.cpp – містить КРДМ, в якому виконується додавання в систему «АКП» нового набору КФ.

print.cpp – містить КРДМ, в якому виконується друк звіту результатів тестування відносно залежності значення часу, необхідного для натискання клавіши, від самої клавіши.

prizn.cpp – містить КРДМ, в якому виконується визначення якості характеристик, що аналізуються, для різних КФ.

read.cpp – містить КРДМ, в якому виконується перегляд звітів результатів розпізнавання, які були збережені раніше.

scor.cpp – містить КРДМ, в якому виконується визначення та відображення статистичних характеристик зразків КП всіх користуваів, що накопичені в БДНЗ, серед яких: сумарний час, який користувач витратив під час набору тексту; кількість набораних ним символів; середній час набору; швидкість набору; кількість зроблених під час набору помилок; відсоток помилок відносно загальної кількості введених символів; нерівномірність вводу.

tabl.cpp – містить КРДМ, в якому виконується копіювання НЗ КП з одного файлу БДНЗ в інший.

test.cpp – містить КРДМ, в якому виконується введення з клавіатури КФ (слово-пароль) та супутньої інформації (прізвище, ім'я, по-батькові, дата). Слово-пароль, який було введено, відображається, а потім обробляється.

Лістинг основних файлів реалізацій модулів:

Файл Project1.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
USERES("Project1.res");
USEFORM("main.cpp", fm_main);
USEFORM("test.cpp", fm_test);
USEFORM("error.cpp", fm_error);
USEFORM("nastr.cpp", fm_nastr);
USEFORM("new_nab.cpp", fm_new_nab);
USEFORM("print.cpp", fm_print);
USEFORM("tabl.cpp", fm_tabl);
USEFORM("analiz.cpp", fm_analiz);
USEFORM("net.cpp", fm_net);
USEFORM("net_otch.cpp", fm_net_otch);
USEFORM("net_test.cpp", fm_net_test);
USEFORM("read.cpp", fm_read);
USEFORM("scor.cpp", fm_scor);
USEFORM("access.cpp", fm_access);
USEFORM("prizn.cpp", fm_prizn);
//---------------------------------------------------------------------------
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
   try
   {
      Application->Initialize();
      Application->CreateForm(__classid(Tfm_main), &fm_main);
      Application->CreateForm(__classid(Tfm_net_otch), &fm_net_otch);
      Application->CreateForm(__classid(Tfm_net_test), &fm_net_test);
      Application->CreateForm(__classid(Tfm_print), &fm_print);
      Application->CreateForm(__classid(Tfm_read), &fm_read);
      Application->Run();
   }
   catch (Exception &exception)
   {
      Application->ShowException(&exception);
   }
   return 0;
}
//---------------------------------------------------------------------------
```

Файл main.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "main.h"
#include "analiz.h"
#include "error.h"
#include "nastr.h"
#include "new_nab.h"
#include "print.h"
#include "tabl.h"
#include "test.h"
#include "net.h"
#include "net_otch.h"
#include "net_test.h"
#include "scor.h"
#include "access.h"
#include "prizn.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_main *fm_main;
//---------------------------------------------------------------------------
__fastcall Tfm_main::Tfm_main(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N1Click(TObject *Sender)
{Tfm_test *fm_test=new Tfm_test(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N8Click(TObject *Sender)
```

```
{Tfm_analiz *fm_analiz=new Tfm_analiz(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N5Click(TObject *Sender)
{Tfm_nastr *fm_nastr=new Tfm_nastr(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N6Click(TObject *Sender)
{fm_main->k=0;
Tfm_new_nab *fm_new_nab=new Tfm_new_nab(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N7Click(TObject *Sender)
{Tfm_tabl *fm_tabl=new Tfm_tabl(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N4Click(TObject *Sender)
{Close(); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N9Click(TObject *Sender)
{Tfm_net *fm_net=new Tfm_net(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N10Click(TObject *Sender)
{Tfm_scor *fm_scor=new Tfm_scor(Application); }
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N11Click(TObject *Sender)
{Tfm_access *fm_access=new Tfm_access(Application); }
//---------------------------------------------------------------------------

void __fastcall Tfm_main::FormCreate(TObject *Sender)
{
fm_main->N5->Enabled=false; fm_main->N6->Enabled=false;
fm_main->N7->Enabled=false; fm_main->N9->Enabled=false;
fm_main->N10->Enabled=false; fm_main->N12->Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_main::N12Click(TObject *Sender)
{Tfm_prizn *fm_prizn=new Tfm_prizn(Application); }
//---------------------------------------------------------------------------
```

Файл access.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "access.h"
#include "main.h"
#include "error.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_access *fm_access;
//---------------------------------------------------------------------------
__fastcall Tfm_access::Tfm_access(TComponent* Owner)
    : TForm(Owner)
{ }
//---------------------------------------------------------------------------
void __fastcall Tfm_access::FormClose(TObject *Sender,
     TCloseAction &Action)
{Action=caFree;    }
//---------------------------------------------------------------------------
void __fastcall Tfm_access::FormCreate(TObject *Sender)
{ ed_pas->Text=""; }
//---------------------------------------------------------------------------
void __fastcall Tfm_access::bb_OkClick(TObject *Sender)
{
if(ed_pas->Text=="імовірність")
{
 fm_main->N5->Enabled=true;  fm_main->N6->Enabled=true;
 fm_main->N7->Enabled=true;  fm_main->N9->Enabled=true;
 fm_main->N10->Enabled=true;  fm_main->N12->Enabled=true;
}
else
{
 fm_main->er="Цей пароль не додає права доступу.";
 Tfm_error *fm_error=new Tfm_error(Application);
}
Close();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_access::ed_pasKeyPress(TObject *Sender, char &Key)
{if (Key==VK_RETURN) bb_Ok->Click();}
//---------------------------------------------------------------------------
```

Файл analiz.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "analiz.h"
#include "main.h"
#include "error.h"
#include "print.h"
#include "net_test.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_analiz *fm_analiz;
int k_L[10],k_R[10],Si_L[10][10][20],Si_R[10][10][20],
Simv,dl_L[10],dl_R[10];
char *s,*s2,s3_L[20],s3_R[20];
AnsiString s1,str;
//---------------------------------------------------------------------------
__fastcall Tfm_analiz::Tfm_analiz(TComponent* Owner)
    : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::FormClose(TObject *Sender,
     TCloseAction &Action)
{Action=caFree; }
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::FormCreate(TObject *Sender)
{
int i,r,t,j,i1,i2;
AnsiString S[1000],S1[1000];
AnsiString SS;
tb_main->Close(); tb_main->Open(); tb_main->First();
while (!tb_main->Eof)
{
if (tb_mainAll_Slovo->AsInteger==0)
   tb_main->Delete();
else
   tb_main->Next();
}
tb_main->Close(); bb_Otkl->Enabled=false;
//Заповнення cb_Fam даними та їх сортування
 cb_Fam->Clear();  tb_main->Open();  tb_main->First();
 i=0;
 while (!tb_main->Eof)
 {
 r=0;
 if (tb_mainFam->AsString!="")
 {
  for (t=0;t<i && r!=1;t++)
 if (S[t]==tb_mainFam->AsString) r=1;
 if (r!=1){S[i]=tb_mainFam->AsString;
          cb_Fam->Items->Add(tb_mainFam->AsString);
          i++;}
 }
 tb_main->Next();
 }
 cb_Fam->Sorted=true;  cb_Fam->Text="";
 // Заповнення cb_Slovo даними та їх сортування
  cb_Slovo->Clear();  tb_main->Open();  tb_main->First();
 i=0;
 while (!tb_main->Eof)
 {
 r=0;
 if (tb_mainSlovo->AsString!="")
 {
  for (t=0;t<i && r!=1;t++)
 if (S1[t]==tb_mainSlovo->AsString) r=1;
 if (r!=1){S1[i]=tb_mainSlovo->AsString;
          cb_Slovo->Items->Add(tb_mainSlovo->AsString);
          i++;}
 }
 tb_main->Next();
 }
 cb_Slovo->Sorted=true;   cb_Slovo->Text="";  lb_Kol->Visible=false;
 for(i2=0;i2<10;i2++)
 {
 k_L[i2]=0;  k_R[i2]=0;
 for(i1=0;i1<10;i1++)
 for(j=0;j<20;j++)
```

```
 {   Si_L[i2][i1][j]=0;   Si_R[i2][i1][j]=0;   }
 }
 rgr->ItemIndex=0;
 lb_Fam_1L->Caption="";  lb_Data_1L->Caption="";
 lb_Time_1L->Caption="";  lb_Nab_1L->Caption="";
lb_Fam_2L->Caption="";  lb_Data_2L->Caption="";
lb_Time_2L->Caption="";  lb_Nab_2L->Caption="";
lb_Fam_3L->Caption="";  lb_Data_3L->Caption="";
lb_Time_3L->Caption="";  lb_Nab_3L->Caption="";
lb_Fam_4L->Caption="";  lb_Data_4L->Caption="";
lb_Time_4L->Caption="";  lb_Nab_4L->Caption="";
lb_Fam_5L->Caption="";  lb_Data_5L->Caption="";
lb_Time_5L->Caption="";  lb_Nab_5L->Caption="";
lb_Fam_6L->Caption="";  lb_Data_6L->Caption="";
lb_Time_6L->Caption="";  lb_Nab_6L->Caption="";
lb_Fam_7L->Caption="";  lb_Data_7L->Caption="";
lb_Time_7L->Caption="";  lb_Nab_7L->Caption="";
lb_Fam_8L->Caption="";  lb_Data_8L->Caption="";
lb_Time_8L->Caption="";  lb_Nab_8L->Caption="";
lb_Fam_9L->Caption="";  lb_Data_9L->Caption="";
lb_Time_9L->Caption="";  lb_Nab_9L->Caption="";
lb_Fam_10L->Caption="";  lb_Data_10L->Caption="";
lb_Time_10L->Caption="";  lb_Nab_10L->Caption="";
lb_Fam_1R->Caption="";  lb_Data_1R->Caption="";
lb_Time_1R->Caption="";  lb_Nab_1R->Caption="";
lb_Fam_2R->Caption="";  lb_Data_2R->Caption="";
lb_Time_2R->Caption="";  lb_Nab_2R->Caption="";
lb_Fam_3R->Caption="";  lb_Data_3R->Caption="";
lb_Time_3R->Caption="";  lb_Nab_3R->Caption="";
lb_Fam_4R->Caption="";  lb_Data_4R->Caption="";
lb_Time_4R->Caption="";  lb_Nab_4R->Caption="";
lb_Fam_5R->Caption="";  lb_Data_5R->Caption="";
lb_Time_5R->Caption="";  lb_Nab_5R->Caption="";
lb_Fam_6R->Caption="";  lb_Data_6R->Caption="";
lb_Time_6R->Caption="";  lb_Nab_6R->Caption="";
lb_Fam_7R->Caption="";  lb_Data_7R->Caption="";
lb_Time_7R->Caption="";  lb_Nab_7R->Caption="";
lb_Fam_8R->Caption="";  lb_Data_8R->Caption="";
lb_Time_8R->Caption="";  lb_Nab_8R->Caption="";
lb_Fam_9R->Caption="";  lb_Data_9R->Caption="";
lb_Time_9R->Caption="";  lb_Nab_9R->Caption="";
lb_Fam_10R->Caption="";  lb_Data_10R->Caption="";
lb_Time_10R->Caption="";  lb_Nab_10R->Caption="";
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::bb_analizClick(TObject *Sender)
{
AnsiString S;
lb_Kol->Visible=true;
if (cb_Fam->Text=="" && cb_Slovo->Text=="")
      {
       S="select count(*) from 'DATA\\main.db'";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->Open();
       lb_Kol->Caption="Кількість знайдених записів: " +
            qr_analiz->Fields[0]->Value;
       S="select * from 'DATA\\main.db'";
      qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S); qr_analiz->Open();
      }
if (cb_Fam->Text!="" && cb_Slovo->Text=="")
      {
       S="select count(*) from 'DATA\\main.db' where Fam=:F";
       qr_analiz->SQL->Clear();  qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("F")->AsString=cb_Fam->Text.c_str();
       qr_analiz->Open();
       lb_Kol->Caption="Кількість знайдених записів:" +
            qr_analiz->Fields[0]->Value;
       S="select * from 'DATA\\main.db' where Fam=:F";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("F")->AsString=cb_Fam->Text.c_str();
       qr_analiz->Open();
      }
if (cb_Fam->Text=="" && cb_Slovo->Text!="")
      {
       S="select count(*) from 'DATA\\main.db' where Slovo=:S";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("S")->AsString=cb_Slovo->Text.c_str();
       qr_analiz->Open();
       lb_Kol->Caption="Кількість знайдених записів:" +
            qr_analiz->Fields[0]->Value;
       S="select * from 'DATA\\main.db' where Slovo=:S";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("S")->AsString=cb_Slovo->Text.c_str();
       qr_analiz->Open();
      }
if (cb_Fam->Text!="" && cb_Slovo->Text!="")
      {
       S="select count(*) from 'DATA\\main.db' where Fam=:F and
Slovo=:S";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("F")->AsString=cb_Fam->Text.c_str();
       qr_analiz->ParamByName("S")->AsString=cb_Slovo->Text.c_str();
       qr_analiz->Open();
       lb_Kol->Caption="Кількість знайдених записів:" +
            qr_analiz->Fields[0]->Value;

       S="select * from 'DATA\\main.db' where Fam=:F and Slovo=:S";
       qr_analiz->SQL->Clear(); qr_analiz->SQL->Add(S);
       qr_analiz->ParamByName("F")->AsString=cb_Fam->Text.c_str();
       qr_analiz->ParamByName("S")->AsString=cb_Slovo->Text.c_str();
       qr_analiz->Open();
      }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::bb_Gr_CClick(TObject *Sender)
{
int i,j;
AnsiString S1;
if(rgr->ItemIndex==0)
 switch (pc_gr_L->ActivePage->PageIndex)
 {
 case 0:
 {
   k_L[pc_gr_L->ActivePage->PageIndex]=0;
   Series1->Clear();  Series2->Clear();  Series3->Clear(); Series4->Clear();
Series5->Clear(); Series6->Clear(); Series7->Clear(); Series8->Clear();
   Series9->Clear(); Series10->Clear(); Series11->Clear();
   dl_L[pc_gr_L->ActivePage->PageIndex]=0;
   for(i=0;i<10;i++)
   for(j=0;j<20;j++)
   Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
   lb_Fam_1L->Caption=""; lb_Data_1L->Caption=""; lb_Time_1L->
Caption="";  lb_Nab_1L->Caption="";
   ch_b_pr_1L->State=cbUnchecked;
   break;
 }
 case 1:
 {
   k_L[pc_gr_L->ActivePage->PageIndex]=0;
   Series12->Clear(); Series13->Clear(); Series14->Clear();
Series15->Clear(); Series16->Clear(); Series17->Clear(); Series18->Clear();
Series19->Clear(); Series20->Clear(); Series21->Clear(); Series22->Clear();
   dl_L[pc_gr_L->ActivePage->PageIndex]=0;
   for(i=0;i<10;i++)
   for(j=0;j<20;j++)
   Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
   lb_Fam_2L->Caption=""; lb_Data_2L->Caption="";
   lb_Time_2L->Caption=""; lb_Nab_2L->Caption="";
   ch_b_pr_2L->State=cbUnchecked;
   break;
 }
 case 2:
 {
   k_L[pc_gr_L->ActivePage->PageIndex]=0;
   Series23->Clear(); Series24->Clear(); Series25->Clear();
Series26->Clear(); Series27->Clear(); Series28->Clear(); Series29->Clear();
Series30->Clear(); Series31->Clear(); Series32->Clear(); Series33->Clear();
   dl_L[pc_gr_L->ActivePage->PageIndex]=0;
   for(i=0;i<10;i++)
   for(j=0;j<20;j++)
   Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
   lb_Fam_3L->Caption=""; lb_Data_3L->Caption="";
   lb_Time_3L->Caption=""; lb_Nab_3L->Caption="";
   ch_b_pr_3L->State=cbUnchecked;
   break;
 }
 case 3:
 {
```

```
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series34->Clear(); Series35->Clear(); Series36->Clear();
    Series37->Clear(); Series38->Clear(); Series39->Clear();
    Series40->Clear(); Series41->Clear(); Series42->Clear();
    Series43->Clear(); Series44->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_4L->Caption=""; lb_Data_4L->Caption="";
    lb_Time_4L->Caption=""; lb_Nab_4L->Caption="";
    ch_b_pr_4L->State=cbUnchecked;
    break;
  }
  case 4:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series45->Clear(); Series46->Clear(); Series47->Clear();
    Series48->Clear(); Series49->Clear(); Series50->Clear();
    Series51->Clear(); Series52->Clear(); Series53->Clear();
    Series54->Clear(); Series55->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_5L->Caption=""; lb_Data_5L->Caption="";
    lb_Time_5L->Caption=""; lb_Nab_5L->Caption="";
    ch_b_pr_5L->State=cbUnchecked;
    break;
  }
  case 5:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series56->Clear(); Series57->Clear(); Series58->Clear();
    Series59->Clear(); Series60->Clear(); Series61->Clear();
    Series62->Clear(); Series63->Clear(); Series64->Clear();
    Series65->Clear(); Series66->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_6L->Caption=""; lb_Data_6L->Caption="";
    lb_Time_6L->Caption=""; lb_Nab_6L->Caption="";
    ch_b_pr_6L->State=cbUnchecked;
    break;
  }
  case 6:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series67->Clear(); Series68->Clear(); Series69->Clear();
    Series70->Clear(); Series71->Clear(); Series72->Clear();
    Series73->Clear(); Series74->Clear(); Series75->Clear();
    Series76->Clear(); Series77->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_7L->Caption=""; lb_Data_7L->Caption="";
    lb_Time_7L->Caption=""; lb_Nab_7L->Caption="";
    ch_b_pr_7L->State=cbUnchecked;
    break;
  }
  case 7:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series78->Clear(); Series79->Clear(); Series80->Clear();
   Series81->Clear(); Series82->Clear(); Series83->Clear(); Series84->Clear();
Series85->Clear(); Series86->Clear(); Series87->Clear(); Series88->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_8L->Caption=""; lb_Data_8L->Caption="";
    lb_Time_8L->Caption=""; lb_Nab_8L->Caption="";
    ch_b_pr_8L->State=cbUnchecked;
    break;
  }
  case 8:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series89->Clear(); Series90->Clear(); Series91->Clear();
    Series92->Clear(); Series93->Clear(); Series94->Clear();
    Series95->Clear(); Series96->Clear(); Series97->Clear();
    Series98->Clear(); Series99->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_9L->Caption=""; lb_Data_9L->Caption="";
    lb_Time_9L->Caption=""; lb_Nab_9L->Caption="";
    ch_b_pr_9L->State=cbUnchecked;
    break;
  }
  case 9:
  {
    k_L[pc_gr_L->ActivePage->PageIndex]=0;
    Series100->Clear(); Series101->Clear(); Series102->Clear();
    Series103->Clear(); Series104->Clear(); Series105->Clear();
    Series106->Clear(); Series107->Clear(); Series108->Clear();
    Series109->Clear(); Series110->Clear();
    dl_L[pc_gr_L->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][j]=0;
    lb_Fam_10L->Caption=""; lb_Data_10L->Caption="";
    lb_Time_10L->Caption=""; lb_Nab_10L->Caption="";
    ch_b_pr_10L->State=cbUnchecked;
    break;
  }
  }
if(rgr->ItemIndex==1)
  switch (pc_gr_R->ActivePage->PageIndex)
  {
  case 0:
  {
    k_R[pc_gr_R->ActivePage->PageIndex]=0;
    Series111->Clear(); Series112->Clear(); Series113->Clear();
    Series114->Clear(); Series115->Clear(); Series116->Clear();
    Series117->Clear(); Series118->Clear(); Series119->Clear();
    Series120->Clear(); Series121->Clear();
    dl_R[pc_gr_R->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
    lb_Fam_1R->Caption=""; lb_Data_1R->Caption="";
    lb_Time_1R->Caption=""; lb_Nab_1R->Caption="";
    ch_b_pr_1R->State=cbUnchecked;
    break;
  }
  case 1:
  {
    k_R[pc_gr_R->ActivePage->PageIndex]=0;
    Series122->Clear(); Series123->Clear(); Series124->Clear();
    Series125->Clear(); Series126->Clear(); Series127->Clear();
    Series128->Clear(); Series129->Clear(); Series130->Clear();
    Series131->Clear(); Series132->Clear();
    dl_R[pc_gr_R->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
    lb_Fam_2R->Caption=""; lb_Data_2R->Caption="";
    lb_Time_2R->Caption=""; lb_Nab_2R->Caption="";
    ch_b_pr_2R->State=cbUnchecked;
    break;
  }
  case 2:
  {
    k_R[pc_gr_R->ActivePage->PageIndex]=0;
    Series133->Clear(); Series134->Clear(); Series135->Clear();
    Series136->Clear(); Series137->Clear(); Series138->Clear();
    Series139->Clear(); Series140->Clear(); Series141->Clear();
    Series142->Clear(); Series143->Clear();
    dl_R[pc_gr_R->ActivePage->PageIndex]=0;
    for(i=0;i<10;i++)
    for(j=0;j<20;j++)
    Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
    lb_Fam_3R->Caption=""; lb_Data_3R->Caption="";
    lb_Time_3R->Caption=""; lb_Nab_3R->Caption="";
```

```
  ch_b_pr_3R->State=cbUnchecked;
  break;
}
case 3:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series144->Clear(); Series145->Clear(); Series146->Clear();
  Series147->Clear(); Series148->Clear(); Series149->Clear();
  Series150->Clear(); Series151->Clear(); Series152->Clear();
  Series153->Clear(); Series154->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_4R->Caption=""; lb_Data_4R->Caption="";
  lb_Time_4R->Caption=""; lb_Nab_4R->Caption="";
  ch_b_pr_4R->State=cbUnchecked;
  break;
}
case 4:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series155->Clear(); Series156->Clear(); Series157->Clear();
  Series158->Clear(); Series159->Clear(); Series160->Clear();
  Series161->Clear(); Series162->Clear(); Series163->Clear();
  Series164->Clear(); Series165->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_5R->Caption=""; lb_Data_5R->Caption="";
  lb_Time_5R->Caption=""; lb_Nab_5R->Caption="";
  ch_b_pr_5R->State=cbUnchecked;
  break;
}
case 5:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series166->Clear(); Series167->Clear(); Series168->Clear();
  Series169->Clear(); Series170->Clear(); Series171->Clear();
  Series172->Clear(); Series173->Clear(); Series174->Clear();
  Series175->Clear(); Series176->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_6R->Caption=""; lb_Data_6R->Caption="";
  lb_Time_6R->Caption=""; lb_Nab_6R->Caption="";
  ch_b_pr_6R->State=cbUnchecked;
  break;
}
case 6:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series177->Clear(); Series178->Clear(); Series179->Clear();
  Series180->Clear(); Series181->Clear(); Series182->Clear();
  Series183->Clear(); Series184->Clear(); Series185->Clear();
  Series186->Clear(); Series187->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_7R->Caption=""; lb_Data_7R->Caption="";
  lb_Time_7R->Caption=""; lb_Nab_7R->Caption="";
  ch_b_pr_7R->State=cbUnchecked;
  break;
}
case 7:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series188->Clear(); Series189->Clear(); Series190->Clear();
  Series191->Clear(); Series192->Clear(); Series193->Clear();
  Series194->Clear(); Series195->Clear(); Series196->Clear();
  Series197->Clear(); Series198->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_8R->Caption=""; lb_Data_8R->Caption="";
  lb_Time_8R->Caption=""; lb_Nab_8R->Caption="";
  ch_b_pr_8R->State=cbUnchecked;
  break;
}
case 8:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series199->Clear(); Series200->Clear(); Series201->Clear();
  Series202->Clear(); Series203->Clear(); Series204->Clear();
  Series205->Clear(); Series206->Clear(); Series207->Clear();
  Series208->Clear(); Series209->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_9R->Caption=""; lb_Data_9R->Caption="";
  lb_Time_9R->Caption=""; lb_Nab_9R->Caption="";
  ch_b_pr_9R->State=cbUnchecked;
  break;
}
case 9:
{
  k_R[pc_gr_R->ActivePage->PageIndex]=0;
  Series210->Clear(); Series211->Clear(); Series212->Clear();
  Series213->Clear(); Series214->Clear(); Series215->Clear();
  Series216->Clear(); Series217->Clear(); Series218->Clear();
  Series219->Clear(); Series220->Clear();
  dl_R[pc_gr_R->ActivePage->PageIndex]=0;
  for(i=0;i<10;i++)
  for(j=0;j<20;j++)
  Si_R[pc_gr_R->ActivePage->PageIndex][i][j]=0;
  lb_Fam_10R->Caption=""; lb_Data_10R->Caption="";
  lb_Time_10R->Caption=""; lb_Nab_10R->Caption="";
  ch_b_pr_10R->State=cbUnchecked;
  break;
}
}
S1="delete from 'DATA\\main_print_i.db' where N_Gr=:N";
qr_analiz_print->SQL->Clear();
qr_analiz_print->SQL->Add(S1);
if (rgr->ItemIndex==0) qr_analiz_print->ParamByName("N")->
AsString=(IntToStr(pc_gr_L->ActivePage->PageIndex+1)+"L");
      else if (rgr->ItemIndex==1) qr_analiz_print->ParamByName("N")->
AsString=(IntToStr(pc_gr_R->ActivePage->PageIndex+1)+"R");
qr_analiz_print->ExecSQL();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::dbgr_qr_analizCellClick(TColumn *Column)
{
int i,sr,c;
qr_analiz->Open();
if (qr_analiz->Fields[7]->AsInteger<5) c=qr_analiz->Fields[7]->AsInteger-1;
      else c=qr_analiz->Fields[7]->AsInteger;
if(rgr->ItemIndex==0)
{
switch (k_L[pc_gr_L->ActivePage->PageIndex])
  {
   case 0:
     {
      switch (pc_gr_L->ActivePage->PageIndex)
         {
         case 0:
         {
         lb_Fam_1L->Caption=qr_analiz->Fields[0]->AsString;
         lb_Data_1L->Caption=qr_analiz->Fields[3]->Value;
         lb_Time_1L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_1L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
         break;
         }
         case 1:
         {
         lb_Fam_2L->Caption=qr_analiz->Fields[0]->AsString;
         lb_Data_2L->Caption=qr_analiz->Fields[3]->Value;
         lb_Time_2L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_2L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
         break;
         }
         case 2:
         {
```

```
        lb_Fam_3L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_3L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_3L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_3L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 3:
        {
        lb_Fam_4L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_4L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_4L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_4L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 4:
        {
        lb_Fam_5L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_5L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_5L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_5L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 5:
        {
        lb_Fam_6L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_6L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_6L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_6L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 6:
        {
        lb_Fam_7L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_7L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_7L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_7L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 7:
        {
        lb_Fam_8L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_8L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_8L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_8L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 8:
        {
        lb_Fam_9L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_9L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_9L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_9L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 9:
        {
        lb_Fam_10L->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_10L->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_10L->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_10L->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger)+ " набір";
        break;
        }
        }
qr_analiz->Open(); tb_main_print->Open(); tb_main_print->Insert();
tb_main_printFam->Value=qr_analiz->Fields[0]->Value;
tb_main_printData->Value=qr_analiz->Fields[3]->Value;
tb_main_printTime->Value=qr_analiz->Fields[4]->Value;
tb_main_printN_Nab->Value=qr_analiz->Fields[5]->Value;
tb_main_printN_Gr->Value=IntToStr(pc_gr_L->ActivePage->PageIndex+1)
+ "L";
 switch (pc_gr_L->ActivePage->PageIndex)
 {
  case 0 :
    {    if (ch_b_pr_1L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_1L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 1 :
    {    if (ch_b_pr_2L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_2L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 2  :
    {    if (ch_b_pr_3L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_3L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 3  :
    {    if (ch_b_pr_4L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_4L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 4  :
    {    if (ch_b_pr_5L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_5L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 5  :
    {    if (ch_b_pr_6L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_6L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 6  :
    {    if (ch_b_pr_7L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_7L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 7  :
    {    if (ch_b_pr_8L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_8L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 8  :
    {    if (ch_b_pr_9L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_9L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
  case 9  :
    {    if (ch_b_pr_10L->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_10L->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;    }
}
tb_main_print->Post(); qr_analiz->Open();
        s1=qr_analiz->Fields[8]->AsString;
        s=s1.c_str();
        strcpy(s3_L,s);
        dl_L[pc_gr_L->ActivePage->PageIndex]=StrLen(s);
        i=0;
        switch (pc_gr_L->ActivePage->PageIndex)
           {
           case 0:
           { Series1->Clear(); break;  }
           case 1:
           { Series12->Clear(); break; }
           case 2:
           { Series23->Clear(); break; }
           case 3:
           { Series34->Clear(); break; }
           case 4:
           { Series45->Clear(); break; }
           case 5:
           { Series56->Clear(); break; }
           case 6:
           { Series67->Clear(); break; }
           case 7:
           { Series78->Clear(); break; }
           case 8:
           { Series89->Clear(); break; }
           case 9:
           {Series100->Clear(); break; }
           }
        while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
        {
           Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
           switch (pc_gr_L->ActivePage->PageIndex)
           {
```

```
          case 0:
          { Series1->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 1:
          { Series12->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 2:
          { Series23->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]);  break; }
          case 3:
          { Series34->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 4:
          { Series45->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 5:
          {Series56->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 6:
          { Series67->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 7:
          { Series78->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 8:
          { Series89->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          case 9:
          { Series100->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,
s[i],ColorPalette[c]); break; }
          }
          i++;
        }
         k_L[pc_gr_L->ActivePage->PageIndex]++;
         break;
        }
      case 1:
        {
         s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
         if (sr==0)
           {
           i=0;
           switch (pc_gr_L->ActivePage->PageIndex)
           {
           case 0:
           { Series2->Clear(); break; }
           case 1:
           { Series13->Clear(); break; }
           case 2:
           { Series24->Clear(); break; }
           case 3:
           { Series35->Clear(); break; }
           case 4:
           { Series46->Clear(); break; }
           case 5:
           { Series57->Clear(); break; }
           case 6:
           { Series68->Clear(); break; }
           case 7:
           { Series79->Clear(); break; }
           case 8:
           { Series90->Clear(); break; }
           case 9:
           { Series101->Clear();  break; }
           }
           while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
           {
           switch (pc_gr_L->ActivePage->PageIndex)
           {
           case 0:
           {Series2->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 1:
           { Series13->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 2:
           { Series24->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 3:
           { Series35->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 4:
           { Series46->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 5:
           { Series57->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 6:
           { Series68->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 7:
           { Series79->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 8:
           { Series90->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 9:
           { Series101->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           }
              Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
             i++;
           }
           k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
          else
             {
              fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";            Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
             }
            break;
       }
       case 2:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
        if (sr==0)
          {
           i=0;
           switch (pc_gr_L->ActivePage->PageIndex)
           {
           case 0:
           { Series3->Clear(); break; }
           case 1:
           { Series14->Clear(); break; }
           case 2:
           { Series25->Clear(); break; }
           case 3:
           { Series36->Clear(); break; }
           case 4:
           { Series47->Clear(); break; }
           case 5:
           { Series58->Clear(); break; }
           case 6:
           { Series69->Clear(); break; }
           case 7:
           { Series80->Clear(); break; }
           case 8:
           { Series91->Clear(); break; }
           case 9:
           { Series102->Clear(); break; }
           }
           while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
           {
           switch (pc_gr_L->ActivePage->PageIndex)
           {
           case 0:
           { Series3->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 1:
           { Series14->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 2:
           { Series25->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 3:
           { Series36->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
```

```
ColorPalette[c]); break; }
          case 4:
          { Series47->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series58->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series69->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series80->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series91->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series102->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
          else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
          }
       case 3:
       {
        s1=qr_analiz->Fields[8]->AsString; s2=s1.c_str();  sr=strcmp(s3_L,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series4->Clear(); break; }
          case 1:
          { Series15->Clear(); break; }
          case 2:
          { Series26->Clear(); break; }
          case 3:
          { Series37->Clear(); break; }
          case 4:
          { Series48->Clear(); break; }
          case 5:
          { Series59->Clear(); break; }
          case 6:
          { Series70->Clear(); break; }
          case 7:
          { Series81->Clear(); break; }
          case 8:
          { Series92->Clear(); break; }
          case 9:
          { Series103->Clear(); break; }
          }
          while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
          {
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series4->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series15->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series26->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series37->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series48->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series59->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series70->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series81->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series92->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series103->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
          else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
          }
       case 4:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series5->Clear(); break; }
          case 1:
          { Series16->Clear(); break; }
          case 2:
          { Series27->Clear(); break; }
          case 3:
          { Series38->Clear(); break; }
          case 4:
          { Series49->Clear(); break; }
          case 5:
          { Series60->Clear(); break; }
          case 6:
          { Series71->Clear(); break; }
          case 7:
          { Series82->Clear(); break; }
          case 8:
          { Series93->Clear(); break; }
          case 9:
          { Series104->Clear(); break; }
          }
          while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
          {
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series5->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series16->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series27->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series38->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
```

```
        { Series49->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 5:
        { Series60->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 6:
        { Series71->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 7:
        { Series82->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 8:
        { Series93->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 9:
        { Series104->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        }
          Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
          i++;
        }
        k_L[pc_gr_L->ActivePage->PageIndex]++;
        }
      else
        {
        fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";   Beep(400,500);
        Tfm_error *fm_error=new Tfm_error(Application);
        }
      break;
      }
    case 5:
    {
     s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
     if (sr==0)
       {
       i=0;
       switch (pc_gr_L->ActivePage->PageIndex)
       {
       case 0:
       { Series6->Clear(); break; }
       case 1:
       { Series17->Clear(); break; }
       case 2:
       { Series28->Clear(); break; }
       case 3:
       { Series39->Clear(); break; }
       case 4:
       { Series50->Clear(); break; }
       case 5:
       { Series61->Clear(); break; }
       case 6:
       { Series72->Clear(); break; }
       case 7:
       { Series83->Clear(); break; }
       case 8:
       { Series94->Clear(); break; }
       case 9:
       { Series105->Clear(); break; }
       }
       while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
       {
       switch (pc_gr_L->ActivePage->PageIndex)
       {
       case 0:
       { Series6->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 1:
       { Series17->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 2:
       { Series28->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 3:
       { Series39->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 4:
       { Series50->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 5:
       { Series61->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 6:
       { Series72->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 7:
       { Series83->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 8:
       { Series94->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       case 9:
       { Series105->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
       }
         Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
         i++;
       }
       k_L[pc_gr_L->ActivePage->PageIndex]++;
       }
     else
       {
       fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";   Beep(400,500);
       Tfm_error *fm_error=new Tfm_error(Application);
       }
     break;
     }
   case 6:
   {
    s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
    if (sr==0)
      {
      i=0;
      switch (pc_gr_L->ActivePage->PageIndex)
      {
      case 0:
      { Series7->Clear(); break; }
      case 1:
      { Series18->Clear(); break; }
      case 2:
      { Series29->Clear(); break; }
      case 3:
      { Series40->Clear(); break; }
      case 4:
      { Series51->Clear(); break; }
      case 5:
      { Series62->Clear(); break; }
      case 6:
      { Series73->Clear(); break; }
      case 7:
      { Series84->Clear(); break; }
      case 8:
      { Series95->Clear(); break; }
      case 9:
      { Series106->Clear(); break; }
      }
      while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
      {
      switch (pc_gr_L->ActivePage->PageIndex)
      {
      case 0:
      { Series7->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
      case 1:
      { Series18->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
      case 2:
      { Series29->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
      case 3:
      { Series40->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
      case 4:
      { Series51->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
```

```
          case 5:
          { Series62->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series73->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series84->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series95->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series106->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
       i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
             }
          break;
         }
       case 7:
       {
        s1=qr_analiz->Fields[8]->AsString; s2=s1.c_str(); sr=strcmp(s3_L,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series8->Clear(); break; }
          case 1:
          { Series19->Clear(); break; }
          case 2:
          { Series30->Clear(); break; }
          case 3:
          { Series41->Clear(); break; }
          case 4:
          { Series52->Clear(); break; }
          case 5:
          { Series63->Clear(); break; }
          case 6:
          { Series74->Clear(); break; }
          case 7:
          { Series85->Clear(); break; }
          case 8:
          { Series96->Clear(); break; }
          case 9:
          { Series107->Clear(); break; }
          }
          while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
          {
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series8->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series19->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series30->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series41->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series52->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series63->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series74->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series85->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series96->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series107->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
             Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
       i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
             }
          break;
         }
       case 8:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str(); sr=strcmp(s3_L,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series9->Clear(); break; }
          case 1:
          { Series20->Clear(); break; }
          case 2:
          { Series31->Clear(); break; }
          case 3:
          { Series42->Clear(); break; }
          case 4:
          { Series53->Clear(); break; }
          case 5:
          { Series64->Clear(); break; }
          case 6:
          { Series75->Clear(); break; }
          case 7:
          { Series86->Clear(); break; }
          case 8:
          { Series97->Clear(); break; }
          case 9:
          { Series108->Clear(); break; }
          }
          while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
          {
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series9->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series20->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series31->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series42->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series53->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series64->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
```

```
ColorPalette[c]); break; }
          case 6:
          { Series75->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series86->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series97->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series108->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       case 9:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_L,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series10->Clear(); break; }
          case 1:
          { Series21->Clear(); break; }
          case 2:
          { Series32->Clear(); break; }
          case 3:
          { Series43->Clear(); break; }
          case 4:
          { Series54->Clear(); break; }
          case 5:
          { Series65->Clear(); break; }
          case 6:
          { Series76->Clear(); break; }
          case 7:
          { Series87->Clear(); break; }
          case 8:
          { Series98->Clear(); break; }
          case 9:
          { Series109->Clear(); break; }
          }
          while (i<dl_L[pc_gr_L->ActivePage->PageIndex])
          {
          switch (pc_gr_L->ActivePage->PageIndex)
          {
          case 0:
          { Series10->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series21->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series32->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series43->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series54->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series65->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series76->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series87->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series98->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series109->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
             Si_L[pc_gr_L->ActivePage->PageIndex][k_L[pc_gr_L->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
             i++;
          }
          k_L[pc_gr_L->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       default:
       {
        fm_main->er="Графік можна будувати для різних повторень одного
й того самого слова.";  Beep(400,500);
        Tfm_error *fm_error=new Tfm_error(Application);
        break;
       }
    }
}
if(rgr->ItemIndex==1)
{
 switch (k_R[pc_gr_R->ActivePage->PageIndex])
    {
     case 0:
       {
        switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          {
          lb_Fam_1R->Caption=qr_analiz->Fields[0]->AsString;
          lb_Data_1R->Caption=qr_analiz->Fields[3]->Value;
          lb_Time_1R->Caption=qr_analiz->Fields[4]->Value;
 lb_Nab_1R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
          break;
          }
          case 1:
          {
          lb_Fam_2R->Caption=qr_analiz->Fields[0]->AsString;
          lb_Data_2R->Caption=qr_analiz->Fields[3]->Value;
          lb_Time_2R->Caption=qr_analiz->Fields[4]->Value;
 lb_Nab_2R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
          break;
          }
          case 2:
          {
          lb_Fam_3R->Caption=qr_analiz->Fields[0]->AsString;
          lb_Data_3R->Caption=qr_analiz->Fields[3]->Value;
          lb_Time_3R->Caption=qr_analiz->Fields[4]->Value;
 lb_Nab_3R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
          break;
          }
          case 3:
          {
          lb_Fam_4R->Caption=qr_analiz->Fields[0]->AsString;
          lb_Data_4R->Caption=qr_analiz->Fields[3]->Value;
          lb_Time_4R->Caption=qr_analiz->Fields[4]->Value;
 lb_Nab_4R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
          break;
          }
          case 4:
          {
          lb_Fam_5R->Caption=qr_analiz->Fields[0]->AsString;
```

```
        lb_Data_5R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_5R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_5R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 5:
        {
        lb_Fam_6R->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_6R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_6R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_6R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 6:
        {
        lb_Fam_7R->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_7R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_7R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_7R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 7:
        {
        lb_Fam_8R->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_8R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_8R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_8R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 8:
        {
        lb_Fam_9R->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_9R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_9R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_9R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger) + " набір";
        break;
        }
        case 9:
        {
        lb_Fam_10R->Caption=qr_analiz->Fields[0]->AsString;
        lb_Data_10R->Caption=qr_analiz->Fields[3]->Value;
        lb_Time_10R->Caption=qr_analiz->Fields[4]->Value;
lb_Nab_10R->Caption=IntToStr(qr_analiz->Fields[5]->AsInteger)+ " набір";
        break;
        }
        }
qr_analiz->Open(); tb_main_print->Open(); tb_main_print->Insert();
tb_main_printFam->Value=qr_analiz->Fields[0]->Value;
tb_main_printData->Value=qr_analiz->Fields[3]->Value;
tb_main_printTime->Value=qr_analiz->Fields[4]->Value;
tb_main_printN_Nab->Value=qr_analiz->Fields[5]->Value;
tb_main_printN_Gr->Value=IntToStr(pc_gr_R->ActivePage->PageIndex+1)
+ "R";
 switch (pc_gr_R->ActivePage->PageIndex)
 {
  case 0 :
    {
    if (ch_b_pr_1R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_1R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 1 :
    {
    if (ch_b_pr_2R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_2R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 2  :
    {
    if (ch_b_pr_3R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_3R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 3  :
    {
    if (ch_b_pr_4R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_4R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 4  :
    {
    if (ch_b_pr_5R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_5R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 5  :
    {
    if (ch_b_pr_6R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_6R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 6  :
    {
    if (ch_b_pr_7R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_7R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 7  :
    {
    if (ch_b_pr_8R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_8R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 8  :
    {
    if (ch_b_pr_9R->State==cbChecked) tb_main_printPrint->Value=true; else
if (ch_b_pr_9R->State==cbUnchecked) tb_main_printPrint->Value=false;
    break;
    }
  case 9  :
    {
    if (ch_b_pr_10R->State==cbChecked) tb_main_printPrint->Value=true;
else if (ch_b_pr_10R->State==cbUnchecked)
tb_main_printPrint->Value=false;
    break;
    }
}
tb_main_print->Post();  qr_analiz->Open();
        s1=qr_analiz->Fields[8]->AsString;  s=s1.c_str();  strcpy(s3_R,s);
        dl_R[pc_gr_R->ActivePage->PageIndex]=StrLen(s);  i=0;
        switch (pc_gr_R->ActivePage->PageIndex)
           {
           case 0:
           { Series111->Clear(); break; }
           case 1:
           { Series122->Clear(); break; }
           case 2:
           { Series133->Clear(); break; }
           case 3:
           { Series144->Clear(); break; }
           case 4:
           { Series155->Clear(); break; }
           case 5:
           { Series166->Clear(); break; }
           case 6:
           { Series177->Clear(); break; }
           case 7:
           { Series188->Clear(); break; }
           case 8:
           { Series199->Clear(); break; }
           case 9:
           { Series210->Clear(); break; }
           }
        while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
        {
           Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
           switch (pc_gr_R->ActivePage->PageIndex)
           {
           case 0:
           { Series111->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
           case 1:
           { Series122->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
           case 2:
           { Series133->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
```

```
          case 3:
          { Series144->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 4:
          { Series155->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 5:
          { Series166->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 6:
          { Series177->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 7:
          { Series188->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 8:
          { Series199->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          case 9:
          { Series210->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,s[i],
ColorPalette[c]); break; }
          }
          i++;
        }
         k_R[pc_gr_R->ActivePage->PageIndex]++;
         break;
        }
      case 1:
        {
         s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str(); sr=strcmp(s3_R,s2);
         if (sr==0)
           {
            i=0;
            switch (pc_gr_R->ActivePage->PageIndex)
            {
            case 0:
            { Series112->Clear(); break; }
            case 1:
            { Series123->Clear(); break; }
            case 2:
            { Series134->Clear(); break; }
            case 3:
            { Series145->Clear(); break; }
            case 4:
            { Series156->Clear(); break; }
            case 5:
            { Series167->Clear(); break; }
            case 6:
            { Series178->Clear(); break; }
            case 7:
            { Series189->Clear(); break; }
            case 8:
            { Series200->Clear(); break; }
            case 9:
            { Series211->Clear(); break; }
            }
            while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
            {
            switch (pc_gr_R->ActivePage->PageIndex)
            {
            case 0:
            { Series112->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 1:
            { Series123->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 2:
            { Series134->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 3:
            { Series145->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 4:
            { Series156->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 5:
            { Series167->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 6:
            { Series178->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 7:
            { Series189->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 8:
            { Series200->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            case 9:
            { Series211->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
            }
               Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
              i++;
            }
            k_R[pc_gr_R->ActivePage->PageIndex]++;
           }
           else
              {
               fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";    Beep(400,500);
               Tfm_error *fm_error=new Tfm_error(Application);
               }
             break;
        }
       case 2:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str(); sr=strcmp(s3_R,s2);
        if (sr==0)
          {
           i=0;
           switch (pc_gr_R->ActivePage->PageIndex)
           {
           case 0:
           { Series113->Clear(); break; }
           case 1:
           { Series124->Clear(); break; }
           case 2:
           { Series135->Clear(); break; }
           case 3:
           { Series146->Clear(); break; }
           case 4:
           { Series157->Clear(); break; }
           case 5:
           { Series168->Clear(); break; }
           case 6:
           { Series179->Clear(); break; }
           case 7:
           { Series190->Clear();  break; }
           case 8:
           { Series201->Clear(); break; }
           case 9:
           { Series212->Clear(); break; }
           }
           while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
           {
           switch (pc_gr_R->ActivePage->PageIndex)
           {
           case 0:
           { Series113->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 1:
           { Series124->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 2:
           { Series135->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 3:
           { Series146->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 4:
           { Series157->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 5:
           { Series168->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
           case 6:
           { Series179->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
```

```
ColorPalette[c]); break; }
          case 7:
          { Series190->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series201->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series212->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";   Beep(400,500);
             Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
          }
       case 3:
       {
        s1=qr_analiz->Fields[8]->AsString; s2=s1.c_str();  sr=strcmp(s3_R,s2);
        if (sr==0)
          {
          i=0;
         switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series114->Clear(); break; }
          case 1:
          { Series125->Clear(); break; }
          case 2:
          { Series136->Clear(); break; }
          case 3:
          { Series147->Clear(); break; }
          case 4:
          { Series158->Clear(); break; }
          case 5:
          { Series169->Clear(); break; }
          case 6:
          { Series180->Clear(); break; }
          case 7:
          { Series191->Clear(); break; }
          case 8:
          { Series202->Clear(); break; }
          case 9:
          { Series213->Clear(); break; }
          }
          while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
          {
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series114->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series125->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series136->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series147->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series158->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series169->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series180->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series191->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series202->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series213->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова."; Beep(400,500);
             Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
          }
       case 4:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_R,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series115->Clear(); break; }
          case 1:
          { Series126->Clear(); break; }
          case 2:
          { Series137->Clear(); break; }
          case 3:
          { Series148->Clear(); break; }
          case 4:
          { Series159->Clear(); break; }
          case 5:
          { Series170->Clear(); break; }
          case 6:
          { Series181->Clear(); break; }
          case 7:
          { Series192->Clear(); break; }
          case 8:
          { Series203->Clear(); break; }
          case 9:
          { Series214->Clear(); break; }
          }
          while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
          {
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series115->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series126->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series137->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series148->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series159->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series170->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series181->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
```

```
        { Series192->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 8:
        { Series203->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 9:
        { Series214->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        }
          Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
          i++;
        }
        k_R[pc_gr_R->ActivePage->PageIndex]++;
        }
       else
          {
          fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";   Beep(400,500);
          Tfm_error *fm_error=new Tfm_error(Application);
          }
        break;
        }
     case 5:
     {
      s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_R,s2);
      if (sr==0)
        {
        i=0;
        switch (pc_gr_R->ActivePage->PageIndex)
        {
        case 0:
        { Series116->Clear(); break; }
        case 1:
        { Series127->Clear(); break; }
        case 2:
        { Series138->Clear(); break; }
        case 3:
        { Series149->Clear(); break; }
        case 4:
        { Series160->Clear(); break; }
        case 5:
        { Series171->Clear(); break; }
        case 6:
        { Series182->Clear(); break; }
        case 7:
        { Series193->Clear(); break; }
        case 8:
        { Series204->Clear(); break; }
        case 9:
        { Series215->Clear(); break; }
        }
        while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
        {
        switch (pc_gr_R->ActivePage->PageIndex)
        {
        case 0:
        { Series116->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 1:
        { Series127->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 2:
        { Series138->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 3:
        { Series149->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 4:
        { Series160->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 5:
        { Series171->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 6:
        { Series182->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 7:
        { Series193->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 8:
        { Series204->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 9:
        { Series215->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        }
          Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
          i++;
        }
        k_R[pc_gr_R->ActivePage->PageIndex]++;
        }
       else
          {
          fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";  Beep(400,500);
          Tfm_error *fm_error=new Tfm_error(Application);
          }
        break;
        }
     case 6:
     {
      s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_R,s2);
      if (sr==0)
        {
        i=0;
        switch (pc_gr_R->ActivePage->PageIndex)
        {
        case 0:
        { Series117->Clear(); break; }
        case 1:
        { Series128->Clear(); break; }
        case 2:
        { Series139->Clear(); break; }
        case 3:
        { Series150->Clear(); break; }
        case 4:
        { Series161->Clear(); break; }
        case 5:
        { Series172->Clear(); break; }
        case 6:
        { Series183->Clear(); break; }
        case 7:
        { Series194->Clear(); break; }
        case 8:
        { Series205->Clear(); break; }
        case 9:
        { Series216->Clear(); break; }
        }
        while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
        {
        switch (pc_gr_R->ActivePage->PageIndex)
        {
        case 0:
        { Series117->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 1:
        { Series128->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 2:
        { Series139->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 3:
        { Series150->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 4:
        { Series161->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 5:
        { Series172->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 6:
        { Series183->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
        case 7:
        { Series194->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
```

```
          case 8:
          { Series205->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series216->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";    Beep(400,500);
             Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       case 7:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_R,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series118->Clear(); break; }
          case 1:
          { Series129->Clear(); break; }
          case 2:
          { Series140->Clear(); break; }
          case 3:
          { Series151->Clear(); break; }
          case 4:
          { Series162->Clear(); break; }
          case 5:
          { Series173->Clear(); break; }
          case 6:
          { Series184->Clear(); break; }
          case 7:
          { Series195->Clear(); break; }
          case 8:
          { Series206->Clear(); break; }
          case 9:
          { Series217->Clear(); break; }
          }
          while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
          {
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series118->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series129->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series140->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series151->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series162->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series173->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series184->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series195->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series206->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series217->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова.";   Beep(400,500);
             Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       case 8:
       {
        s1=qr_analiz->Fields[8]->AsString; s2=s1.c_str();  sr=strcmp(s3_R,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series119->Clear(); break; }
          case 1:
          { Series130->Clear(); break; }
          case 2:
          { Series141->Clear(); break; }
          case 3:
          { Series152->Clear(); break; }
          case 4:
          { Series163->Clear(); break; }
          case 5:
          { Series174->Clear(); break; }
          case 6:
          { Series185->Clear(); break; }
          case 7:
          { Series196->Clear(); break; }
          case 8:
          { Series207->Clear(); break; }
          case 9:
          { Series218->Clear(); break; }
          }
          while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
          {
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series119->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series130->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series141->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series152->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series163->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series174->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series185->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series196->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series207->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
```

```
ColorPalette[c]); break; }
          case 9:
          { Series218->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
            Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
            i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова."; Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       case 9:
       {
        s1=qr_analiz->Fields[8]->AsString;  s2=s1.c_str();  sr=strcmp(s3_R,s2);
        if (sr==0)
          {
          i=0;
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series120->Clear(); break; }
          case 1:
          { Series131->Clear(); break; }
          case 2:
          { Series142->Clear(); break; }
          case 3:
          { Series153->Clear(); break; }
          case 4:
          { Series164->Clear(); break; }
          case 5:
          { Series175->Clear(); break; }
          case 6:
          { Series186->Clear(); break; }
          case 7:
          { Series197->Clear(); break; }
          case 8:
          { Series208->Clear(); break; }
          case 9:
          { Series219->Clear(); break; }
          }
          while (i<dl_R[pc_gr_R->ActivePage->PageIndex])
          {
          switch (pc_gr_R->ActivePage->PageIndex)
          {
          case 0:
          { Series120->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 1:
          { Series131->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 2:
          { Series142->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 3:
          { Series153->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 4:
          { Series164->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 5:
          { Series175->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 6:
          { Series186->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 7:
          { Series197->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 8:
          { Series208->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          case 9:
          { Series219->AddXY(i+1,qr_analiz->Fields[10+i]->AsInteger,"",
ColorPalette[c]); break; }
          }
             Si_R[pc_gr_R->ActivePage->PageIndex][k_R[pc_gr_R->
ActivePage->PageIndex]][i]=qr_analiz->Fields[10+i]->AsInteger;
             i++;
          }
          k_R[pc_gr_R->ActivePage->PageIndex]++;
          }
         else
            {
             fm_main->er="Графік можна будувати для різних повторень
одного й того самого слова."; Beep(400,500);
              Tfm_error *fm_error=new Tfm_error(Application);
            }
          break;
         }
       default:
       {
        fm_main->er="Графік можна будувати для 10 повторень одного й
того самого слова. ";  Beep(400,500);
        Tfm_error *fm_error=new Tfm_error(Application);
        break;
       }
    }
}
tb_main_print->Open();  tb_main_print->First();
 if(rgr->ItemIndex==0)
 {
 switch (pc_gr_L->ActivePage->PageIndex)
 {
 case 0:
 {
 ch_1L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 1:
 {
 ch_2L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 2:
 {
 ch_3L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 3:
 {
 ch_4L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
```

```
 }
 case 4:
 {
 ch_5L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 5:
 {
 ch_6L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 6:
 {
 ch_7L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 7:
 {
 ch_8L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 8:
 {
 ch_9L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 9:
 {
 ch_10L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 }
 }
 if(rgr->ItemIndex==1)
 {
 switch (pc_gr_R->ActivePage->PageIndex)
 {
 case 0:
 {
 ch_1R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 1:
 {
 ch_2R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 2:
 {
 ch_3R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 3:
 {
 ch_4R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 4:
 {
 ch_5R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 5:
 {
 ch_6R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 6:
 {
 ch_7R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
```

```
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 7:
 {
 ch_8R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 8:
 {
 ch_9R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 9:
 {
 ch_10R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 }
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::FormActivate(TObject *Sender)
{ WindowState=wsMaximized; }
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::bb_Gr_SrClick(TObject *Sender)
{
 int i,j;
if(rgr->ItemIndex==0)
{
switch (pc_gr_L->ActivePage->PageIndex)
{
 case 0:
 { Series11->Clear(); break; }
 case 1:
 { Series22->Clear(); break; }
 case 2:
 { Series33->Clear(); break; }
 case 3:
 { Series44->Clear(); break; }
 case 4:
 { Series55->Clear(); break; }
 case 5:
 { Series66->Clear(); break; }
 case 6:
 { Series77->Clear(); break; }
 case 7:
 { Series88->Clear(); break; }
 case 8:
 { Series99->Clear(); break; }
 case 9:
 { Series110->Clear(); break; }
}
 Simv=0;
 for(j=0;j<dl_L[pc_gr_L->ActivePage->PageIndex];j++)
 {
  for(i=0;i<k_L[pc_gr_L->ActivePage->PageIndex];i++)
   Simv=Simv+Si_L[pc_gr_L->ActivePage->PageIndex][i][j];
  Simv=Simv/k_L[pc_gr_L->ActivePage->PageIndex];
  switch (pc_gr_L->ActivePage->PageIndex)
  {
   case 0:
   { Series11->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 1:
   { Series22->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 2:
   { Series33->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 3:
   { Series44->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 4:
   { Series55->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 5:
   { Series66->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 6:
   { Series77->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 7:
   { Series88->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 8:
   { Series99->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 9:
   { Series110->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
  }
  Simv=0;
 }
}
if(rgr->ItemIndex==1)
{
switch (pc_gr_R->ActivePage->PageIndex)
{
 case 0:
 { Series121->Clear(); break; }
 case 1:
 { Series132->Clear(); break; }
 case 2:
 { Series143->Clear(); break; }
 case 3:
 { Series154->Clear(); break; }
 case 4:
 { Series165->Clear(); break; }
 case 5:
 { Series176->Clear(); break; }
 case 6:
 { Series187->Clear(); break; }
 case 7:
 { Series198->Clear(); break; }
 case 8:
 { Series209->Clear(); break; }
 case 9:
 { Series220->Clear(); break; }
}
 Simv=0;
 for(j=0;j<dl_R[pc_gr_R->ActivePage->PageIndex];j++)
 {
  for(i=0;i<k_R[pc_gr_R->ActivePage->PageIndex];i++)
   Simv=Simv+Si_R[pc_gr_R->ActivePage->PageIndex][i][j];
  Simv=Simv/k_R[pc_gr_R->ActivePage->PageIndex];
  switch (pc_gr_R->ActivePage->PageIndex)
  {
   case 0:
   { Series121->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 1:
   { Series132->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 2:
   { Series143->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 3:
   { Series154->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 4:
   { Series165->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 5:
   { Series176->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 6:
   { Series187->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 7:
   { Series198->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 8:
```

```
   { Series209->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
   case 9:
   { Series220->AddXY(j+1,Simv,"",ColorPalette[11]); break; }
  }
  Simv=0;
 }
}
tb_main_print->Open();  tb_main_print->First();
 if(rgr->ItemIndex==0)
 {
 switch (pc_gr_L->ActivePage->PageIndex)
 {
 case 0:
 {
 ch_1L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 1:
 {
 ch_2L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 2:
 {
 ch_3L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 3:
 {
 ch_4L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 4:
 {
 ch_5L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 5:
 {
 ch_6L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 6:
 {
 ch_7L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 7:
 {
 ch_8L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 8:
 {
 ch_9L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 9:
 {
 ch_10L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 }
 }
 if(rgr->ItemIndex==1)
 {
 switch (pc_gr_R->ActivePage->PageIndex)
 {
 case 0:
 {
 ch_1R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 1:
 {
 ch_2R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
```

```
 tb_main_print->Next();
 }
 break;
 }
 case 2:
 {
 ch_3R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 3:
 {
 ch_4R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 4:
 {
 ch_5R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 5:
 {
 ch_6R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 6:
 {
 ch_7R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 7:
 {
 ch_8R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 8:
 {
 ch_9R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 case 9:
 {
 ch_10R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
 break;
 }
 }
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::bb_OtklClick(TObject *Sender)
{
 int i,j,t;
if(rgr->ItemIndex==0)
{
 Simv=0;
 for(j=0;j<dl_L[pc_gr_L->ActivePage->PageIndex];j++)
 {
  for(i=0;i<k_L[pc_gr_L->ActivePage->PageIndex];i++)
   Simv=Simv+Si_L[pc_gr_L->ActivePage->PageIndex][i][j];
  Simv=Simv/k_L[pc_gr_L->ActivePage->PageIndex];
  for (i=0;i<k_L[pc_gr_L->ActivePage->PageIndex];i++)
  if (Si_L[pc_gr_L->ActivePage->PageIndex][i][j]/Simv>5 || Si_L[pc_gr_L->
ActivePage->PageIndex][i][j]/Simv<0.2)
  {
   for(t=0;t<dl_L[pc_gr_L->ActivePage->PageIndex];t++)
    Si_L[pc_gr_L->ActivePage->PageIndex][i][t]=0;
  }
  switch (pc_gr_L->ActivePage->PageIndex)
  {
   case 0:
   { Series11->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 1:
   { Series22->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 2:
   { Series33->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 3:
   { Series44->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 4:
   { Series55->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 5:
   { Series66->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 6:
   { Series77->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 7:
   { Series88->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 8:
   { Series99->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 9:
   { Series110->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
  }
  Simv=0;
 }
}
if(rgr->ItemIndex==1)
{
switch (pc_gr_R->ActivePage->PageIndex)
{
 case 0:
 { Series121->Clear(); break; }
 case 1:
 { Series132->Clear(); break; }
 case 2:
 { Series143->Clear(); break; }
```

```
 case 3:
 { Series154->Clear(); break; }
 case 4:
 { Series165->Clear(); break; }
 case 5:
 { Series176->Clear(); break; }
 case 6:
 { Series187->Clear(); break; }
 case 7:
 { Series198->Clear(); break; }
 case 8:
 { Series209->Clear(); break; }
 case 9:
 { Series220->Clear(); break; }
}
 Simv=0;
 for(j=0;j<dl_R[pc_gr_R->ActivePage->PageIndex];j++)
 {
  for(i=0;i<k_R[pc_gr_R->ActivePage->PageIndex];i++)
   Simv=Simv+Si_R[pc_gr_R->ActivePage->PageIndex][i][j];
  Simv=Simv/k_R[pc_gr_R->ActivePage->PageIndex];
  switch (pc_gr_R->ActivePage->PageIndex)
  {
   case 0:
   { Series121->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 1:
   { Series132->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 2:
   { Series143->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 3:
   { Series154->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 4:
   { Series165->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 5:
   { Series176->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 6:
   { Series187->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 7:
   { Series198->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 8:
   { Series209->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
   case 9:
   { Series220->AddXY(j+1,Simv,"",ColorPalette[10]); break; }
  }
  Simv=0;
 }
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_1LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First(); while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_1L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_1L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
tb_main_print->Post(); tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_10LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First(); while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_10L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_10L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();  tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_10RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_10R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_10R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_1RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_1R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_1R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_2LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_2L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_2L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_2RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_2R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_2R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_3LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_3L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_3L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
```

```
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_3RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_3R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_3R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_4LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_4L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_4L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_4RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_4R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_4R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_5LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_5L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_5L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_5RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_5R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_5R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_6LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_6L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_6L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_6RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_6R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_6R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_7LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_7L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_7L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_7RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_7R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_7R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_8LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_8L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_8L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
```

```
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_8RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_8R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_8R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_9LClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_9L->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_9L->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_L->ActivePage->PageIndex+1) + "L"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_b_pr_9RClick(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
while(!tb_main_print->Eof)
{
  tb_main_print->Edit();
  if (ch_b_pr_9R->State==cbChecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=true;
       else if (ch_b_pr_9R->State==cbUnchecked && tb_main_printN_Gr->
Value==(IntToStr(pc_gr_R->ActivePage->PageIndex+1) + "R"))
tb_main_printPrint->Value=false;
  tb_main_print->Post();   tb_main_print->Next();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::bb_PrintClick(TObject *Sender)
{ fm_print->qrp_print->Preview(); }
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_1LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_1L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_10LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_10L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_10RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_10R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_1RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_1R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_2LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_2L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_2RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_2R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_3LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_3L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_3RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_3R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
```

```
void __fastcall Tfm_analiz::ch_4LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_4L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_4RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_4R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_5LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_5L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_5RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_5R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_6LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_6L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_6RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_6R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_7LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_7L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_7RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_7R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_8LZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_8L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_8RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_8R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_9LZoom(TObject *Sender)
{
tb_main_print->Open(); tb_main_print->First();
 ch_9L->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_L->ActivePage->
PageIndex+1) + "L")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::ch_9RZoom(TObject *Sender)
{
tb_main_print->Open();  tb_main_print->First();
 ch_9R->SaveToBitmapFile("Grafik.bmp");
 while(!tb_main_print->Eof)
 {
  tb_main_print->Edit();
  if (tb_main_printN_Gr->AsString==(IntToStr(pc_gr_R->ActivePage->
PageIndex+1) + "R")) tb_main_printGrafic->LoadFromFile("Grafik.bmp");
  tb_main_print->Next();
 }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_analiz::FormCloseQuery(TObject *Sender, bool
&CanClose)
```

```
{
 AnsiString S1;
 S1="delete from 'DATA\\main_print_i.db'";  qr_analiz_print->SQL->Clear();
qr_analiz_print->SQL->Add(S1);  qr_analiz_print->ExecSQL();
}
//---------------------------------------------------------------------------
```

Файл error.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "error.h"
#include "test.h"
#include "main.h"
#include "new_nab.h"
#include "nastr.h"
#include "analiz.h"
#include "net.h"
#include "access.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_error *fm_error;
//---------------------------------------------------------------------------
__fastcall Tfm_error::Tfm_error(TComponent* Owner)
    : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_error::FormCreate(TObject *Sender)
{
lb_err->Caption=""; lb_err->Caption=fm_main->er;
lb_err->Layout=tlCenter; lb_err->Font->Color=clMaroon;
lb_err->Font->Size=12;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_error::FormClose(TObject *Sender, TCloseAction
&Action)
{Action=caFree; }
//---------------------------------------------------------------------------
void __fastcall Tfm_error::FormActivate(TObject *Sender)
{WindowState=wsNormal;  }
//---------------------------------------------------------------------------
void __fastcall Tfm_error::bb_OkClick(TObject *Sender)
{Close();  }
//---------------------------------------------------------------------------
```

Файл nastr.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "nastr.h"
#include "main.h"
#include "new_nab.h"
#include "error.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "cspin"
#pragma resource "*.dfm"
Tfm_nastr *fm_nastr;
bool n;
//---------------------------------------------------------------------------
__fastcall Tfm_nastr::Tfm_nastr(TComponent* Owner)
    : TForm(Owner)
{ }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::FormClose(TObject *Sender, TCloseAction
&Action)
{ Action=caFree; }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::scr_b_NChange(TObject *Sender)
{
AnsiString S;
int i;
S="select N_Nab, K_Povt, K_Slov, Slovo_1, Slovo_2, Slovo_3, Slovo_4,
Slovo_5, Slovo_6, Slovo_7, Slovo_8, Slovo_9, Slovo_10 from
'DATA\\nabor.db' where N_Nab=:N";
qr_nastr->SQL->Clear(); qr_nastr->SQL->Add(S);
qr_nastr->ParamByName("N")->AsInteger=scr_b_N->Position;
qr_nastr->Open();
ed_N_Nab->Text=qr_nastr->Fields[0]->Value;
ed_K_Povt->Text=qr_nastr->Fields[1]->Value;
ed_K_Slov->Text=qr_nastr->Fields[2]->Value;
lst_bx_Sl->Clear();
for (i=0;i<10;i++)
   lst_bx_Sl->Items->Add(qr_nastr->Fields[3+i]->Value);
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::bb_Izm_NastrClick(TObject *Sender)
{
n=true; fm_main->s=false; scr_b_V_N->Visible=true;
scr_b_K_Povt->Visible=true; scr_b_K_Slov->Visible=true;
bb_Save_N->Visible=true; bb_Cancel->Visible=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::scr_b_V_NChange(TObject *Sender)
{
AnsiString S;
int i;
if (n==true)
      {
       ed_V_Nab->Text=scr_b_V_N->Position;
       ed_N_Nab->Text=ed_V_Nab->Text;
       S="select K_Povt, K_Slov, Slovo_1, Slovo_2, Slovo_3, Slovo_4,
Slovo_5, Slovo_6, Slovo_7, Slovo_8, Slovo_9, Slovo_10 from
'DATA\\nabor.db' where N_Nab=:N";
       qr_nastr->SQL->Clear(); qr_nastr->SQL->Add(S);
      qr_nastr->ParamByName("N")->AsInteger=StrToInt(ed_N_Nab->Text);
       qr_nastr->Open();
       ed_K_Povt->Text=qr_nastr->Fields[0]->Value;
       ed_K_Slov->Text=qr_nastr->Fields[1]->Value;
       lst_bx_Sl->Clear();
       for (i=0;i<10;i++)
          lst_bx_Sl->Items->Add(qr_nastr->Fields[2+i]->Value);
      }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::ed_N_NabChange(TObject *Sender)
{ scr_b_N->Position=StrToInt(ed_N_Nab->Text); fm_main->s=false; }
//---------------------------------------------------------------------------
oid __fastcall Tfm_nastr::ed_K_PovtChange(TObject *Sender)
{ scr_b_K_Povt->Position=StrToInt(ed_K_Povt->Text); fm_main->s=false; }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::ed_K_SlovChange(TObject *Sender)
{ scr_b_K_Slov->Position=StrToInt(ed_K_Slov->Text); fm_main->s=false; }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::scr_b_K_PovtChange(TObject *Sender)
{ ed_K_Povt->Text=scr_b_K_Povt->Position; }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::scr_b_K_SlovChange(TObject *Sender)
{ ed_K_Slov->Text=scr_b_K_Slov->Position; }
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::bb_Save_NClick(TObject *Sender)
{
tb_nastr->Open(); tb_nastr->First();
while(!tb_nastr->Eof)
   {
    tb_nastr->Edit();  tb_nastrV_Nab->AsBoolean=false;
    tb_nastrK_Povt->AsInteger=1; tb_nastrK_Slov->AsInteger=10;
    tb_nastr->Next();
   }
tb_nastr->First();
while (tb_nastrN_Nab->AsInteger!=StrToInt(ed_N_Nab->Text))
      {
      tb_nastr->Next();
      }
tb_nastr->Edit(); tb_nastrV_Nab->AsBoolean=true;
tb_nastrK_Povt->AsInteger=StrToInt(ed_K_Povt->Text);
tb_nastrK_Slov->AsInteger=StrToInt(ed_K_Slov->Text);
tb_nastr->Post(); fm_main->s=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::ed_V_NabChange(TObject *Sender)
{
fm_main->s=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::FormActivate(TObject *Sender)
{
```

```
WindowState=wsMaximized;
AnsiString S;
int i;
n=false;
S="select N_Nab, K_Povt, K_Slov, Slovo_1, Slovo_2, Slovo_3, Slovo_4,
Slovo_5, Slovo_6, Slovo_7, Slovo_8, Slovo_9, Slovo_10 from
'DATA\\nabor.db' where V_Nab=:V";
qr_nastr->SQL->Clear(); qr_nastr->SQL->Add(S);
qr_nastr->ParamByName("V")->AsBoolean=true; qr_nastr->Open();
ed_V_Nab->Text=qr_nastr->Fields[0]->Value;
ed_N_Nab->Text=qr_nastr->Fields[0]->Value;
scr_b_N->Position=qr_nastr->Fields[0]->Value;
ed_K_Povt->Text=qr_nastr->Fields[1]->Value;
ed_K_Slov->Text=qr_nastr->Fields[2]->Value;
tb_nastr->Open(); tb_nastr->First();
scr_b_N->Min=tb_nastrN_Nab->Value;
tb_nastr->Last();
scr_b_N->Max=tb_nastrN_Nab->Value;
ed_V_Nab->ReadOnly=true; ed_N_Nab->ReadOnly=true;
ed_K_Povt->ReadOnly=true; ed_K_Slov->ReadOnly=true;
scr_b_V_N->Min=scr_b_N->Min; scr_b_V_N->Max=scr_b_N->Max;
scr_b_K_Povt->Min=1; scr_b_K_Povt->Max=10; scr_b_K_Slov->Min=1;
scr_b_K_Slov->Max=10;
scr_b_V_N->Position=StrToInt(ed_V_Nab->Text);
scr_b_K_Povt->Position=StrToInt(ed_K_Povt->Text);
scr_b_K_Slov->Position=StrToInt(ed_K_Slov->Text);
if ( fm_main->z==false)
{
 scr_b_V_N->Visible=false;  scr_b_K_Povt->Visible=false;
 scr_b_K_Slov->Visible=false;  bb_Save_N->Visible=false;
 bb_Cancel->Visible=false;
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::FormCloseQuery(TObject *Sender, bool
&CanClose)
{
if (fm_main->s==false && bb_Save_N->Visible==true)
{
   fm_main->z=true;     CanClose=false;
   fm_main->er="Необхідно зберегти зміни."; Beep(400,500);
   Tfm_error *fm_error=new Tfm_error(Application);
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::FormCreate(TObject *Sender)
{
fm_main->s=true; fm_main->z=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_nastr::bb_CancelClick(TObject *Sender)
{
AnsiString S;
int i;
S="select N_Nab, K_Povt, K_Slov, Slovo_1, Slovo_2, Slovo_3, Slovo_4,
Slovo_5, Slovo_6, Slovo_7, Slovo_8, Slovo_9, Slovo_10 from
'DATA\\nabor.db' where V_Nab=:V";
qr_nastr->SQL->Clear(); qr_nastr->SQL->Add(S);
qr_nastr->ParamByName("V")->AsBoolean=true;
qr_nastr->Open();
ed_V_Nab->Text=qr_nastr->Fields[0]->Value;
ed_N_Nab->Text=qr_nastr->Fields[0]->Value;
scr_b_N->Position=qr_nastr->Fields[0]->Value;
ed_K_Povt->Text=qr_nastr->Fields[1]->Value;
ed_K_Slov->Text=qr_nastr->Fields[2]->Value;
tb_nastr->Open(); tb_nastr->First();
scr_b_N->Min=tb_nastrN_Nab->Value;
tb_nastr->Last();
scr_b_N->Max=tb_nastrN_Nab->Value;
ed_V_Nab->ReadOnly=true; ed_N_Nab->ReadOnly=true;
ed_K_Povt->ReadOnly=true; ed_K_Slov->ReadOnly=true;
scr_b_V_N->Min=scr_b_N->Min; scr_b_V_N->Max=scr_b_N->Max;
scr_b_K_Povt->Min=1; scr_b_K_Povt->Max=10;
scr_b_K_Slov->Min=1; scr_b_K_Slov->Max=10;
scr_b_V_N->Position=StrToInt(ed_V_Nab->Text);
scr_b_K_Povt->Position=StrToInt(ed_K_Povt->Text);
scr_b_K_Slov->Position=StrToInt(ed_K_Slov->Text);
fm_main->s=true;
}
```

```
//---------------------------------------------------------------------------
```

Файл net.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <math.h>
#pragma hdrstop
#include "net.h"
#include "main.h"
#include "net_otch.h"
#include "error.h"
#include "net_test.h"
#include "read.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_net *fm_net;
AnsiString S1[1000];
int i,r,t,kol_z,minim;
int met=0;
//---------------------------------------------------------------------------
__fastcall Tfm_net::Tfm_net(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::FormClose(TObject *Sender, TCloseAction
&Action)
{
AnsiString S;
S="delete from 'DATA\\main_promeg.db'";
         qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
         qr_net_test_boch->ExecSQL(); Action=caFree;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::FormCreate(TObject *Sender)
{
 //Заповнення clb_Fam даними та їх сортування
 clb_Fam->Clear();  tb_main->Open();  tb_main->First();
 kol_z=0;
 i=0;
 while (!tb_main->Eof)
 {
 r=0;
 if (tb_mainFam->AsString!="")
 {
  for (t=0;t<i && r!=1;t++)
 if (S1[t]==tb_mainFam->AsString) r=1;
 if (r!=1){S1[i]=tb_mainFam->AsString;
         clb_Fam->Items->Add(tb_mainFam->AsString);
         i++;}
 }
 kol_z++;
 tb_main->Next();
 }
 clb_Fam->Sorted=true;
 lb_Kol_z->Caption="Кількість знайденых записів: " + IntToStr(kol_z);
 lb_Kol_f->Caption="Кількість знайденых користувачів: " + IntToStr(i);
 rgr_otb->ItemIndex=0;  rgr_otb->Visible=false;
 rgr_pr->ItemIndex=0;  rgr_pr->Visible=false;
 rgr_usr->ItemIndex=0;  rgr_usr->Visible=false;
 rgr_fun->ItemIndex=0;  rgr_fun->Visible=false;
 lb_inf->Visible=false;  lb_inf_t->Visible=false;  lb_kol_pol->Visible=false;
ed_f->Visible=false;  ed_f_t->Visible=false;  ed_kol_pol->Visible=false;
 bb_net->Visible=false;  bb_Ok->Visible=false;
 bb_net_test->Enabled=false;  bb_net_test_boch->Enabled=false;
bb_net_test_boch_1->Enabled=false;  bb_razd_dan->Enabled=false;
 bb_Ok->Enabled=false;  bb_no_all->Enabled=false;  bb_all->Enabled=true;
 lb_min->Visible=false;  lb_boch->Visible=false;  lb_boch_1->Visible=false;
ed_boch->Visible=false;  bb_boch->Visible=false;  lb_uch->Visible=false;
 ed_uch->Visible=false;
 fm_main->test_fun=2;
 }
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_razd_danClick(TObject *Sender)
{
   int j,k,ud,t,kol=10,x,z;
   bool pp;
   AnsiString S,S2,slovo,fam;
   bb_razd_dan->Enabled=false; clb_Fam->Enabled=false;
```

```
    ed_f->Text="";
    lb_inf->Visible=false; ed_f->Visible=false; bb_net->Visible=false;
rgr_otb->Visible=false;
    S2="delete from 'DATA\\uch_dan.db'";
    qr_tabl_i->SQL->Clear();  qr_tabl_i->SQL->Add(S2);
    qr_tabl_i->ExecSQL();
    S2="delete from 'DATA\\neiz_ekz.db'";
    qr_tabl_i->SQL->Clear(); qr_tabl_i->SQL->Add(S2);
    qr_tabl_i->ExecSQL();
    i=0;
    for (t=0;t<clb_Fam->Items->Count;t++)
       if (clb_Fam->Checked[t]==true) {fm_main->SS[i]=clb_Fam->Items->
Strings[t]; i++;}
    fm_main->kol_fam=i;
    for (j=0;j<i;j++)
    {
    S="select * from 'DATA\\main.db' where Fam=:F";
         qr_tabl_i->SQL->Clear();  qr_tabl_i->SQL->Add(S);
         qr_tabl_i->ParamByName("F")->AsString=fm_main->SS[j];
         qr_tabl_i->Open();
    ud=0;
    tb_tabl_p_1->TableName="DATA\\uch_dan.db";
    tb_tabl_p_1->Active=true; tb_tabl_p_1->Open();
    tb_tabl_p_2->TableName="DATA\\neiz_ekz.db";
    tb_tabl_p_2->Active=true; tb_tabl_p_2->Open();
    while(!qr_tabl_i->Eof && ud<=999)
    {
     x=0;
     if (qr_tabl_i->Fields[7]->AsInteger==1)
     {
      slovo=qr_tabl_i->Fields[8]->AsString;
      fam=qr_tabl_i->Fields[0]->AsString;
    while(!qr_tabl_i->Eof && ud<=999 && x<kol && qr_tabl_i->Fields[7]->
AsInteger==(x+1) && qr_tabl_i->Fields[8]->AsString==slovo && qr_tabl_i-
>Fields[0]->AsString==fam)
       {
        x++;  qr_tabl_i->Next();
       }
       for(z=0;z<x;z++)
        qr_tabl_i->Prior();
      }
      if(x==kol)
      {
       for (z=0;z<kol;z++)
       {
        tb_tabl_p_1->First();  tb_tabl_p_1->Insert();
        for (k=0;k<31;k++)
         tb_tabl_p_1->Fields[k]->AsString=qr_tabl_i->Fields[k]->AsString;
        qr_tabl_i->Next();
        ud++;
       }
      }
      else
      {
       for (z=0;z<=x;z++)
       {
        tb_tabl_p_2->First(); tb_tabl_p_2->Insert();
        for (k=0;k<31;k++)
         tb_tabl_p_2->Fields[k]->AsString=qr_tabl_i->Fields[k]->AsString;
        qr_tabl_i->Next();
       }
      }
    }
    tb_tabl_p_1->Post(); tb_tabl_p_1->Close();
    if (ud>999)
       {
        while(!qr_tabl_i->Eof)
        {
        tb_tabl_p_2->First(); tb_tabl_p_2->Insert();
        for (k=0;k<31;k++)
            tb_tabl_p_2->Fields[k]->AsString=qr_tabl_i->Fields[k]->AsString;
        qr_tabl_i->Next();
        }
        tb_tabl_p_2->Post(); tb_tabl_p_2->Close();
       }
      tb_tabl_p_2->Close();
    }
    Beep(400,500); lb_inf->Caption="Введіть ширину функції.";
    rgr_otb->ItemIndex=0; bb_razd_dan->Enabled=true;
    clb_Fam->Enabled=true; lb_inf->Visible=true;
    ed_f->Visible=true; rgr_otb->Visible=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_netClick(TObject *Sender)
{
if  (StrToFloat(ed_f->Text)<=0)
    {
fm_main->er="   Ширина функції          повинна бути     більша ніж '0' !!!";
     Beep(400,500);
     Tfm_error *fm_error=new Tfm_error(Application);
    }
else
{
 if(rgr_otb->ItemIndex==0) fm_main->otb=false; else fm_main->otb=true;
 fm_main->f=StrToFloat(ed_f->Text);
 fm_net_otch->qrp_net_otch->Preview();
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::ed_fChange(TObject *Sender)
{ if (ed_f->Text!="") bb_net->Visible=true; }
//---------------------------------------------------------------------------
void __fastcall Tfm_net::FormActivate(TObject *Sender)
{ WindowState=wsMaximized; }
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_net_testClick(TObject *Sender)
{
int i,t,kl;
AnsiString S;
fm_main->tt="DATA\\main.db"; fm_main->test_fun=1;
lb_min->Caption=" "; lb_min->Visible=false;
clb_Fam->Enabled=false; bb_boch->Enabled=false; rgr_pr->Enabled=false;
bb_razd_dan->Enabled=false; bb_all->Enabled=false;
bb_no_all-> Enabled=false; bb_read->Enabled=false;
bb_net_test->Enabled=false; bb_net_test_boch-> Enabled=false;
bb_net_test_boch_1->Enabled=false; ed_boch-> Enabled=false;
ed_kol_pol->Enabled=false; rgr_pr->Enabled=false; rgr_usr->Enabled=false;
rgr_fun->Enabled=false; ed_uch->Enabled=false; ed_f_t->Enabled=false;
i=0;
   for (t=0;t<clb_Fam->Items->Count;t++)
      if (clb_Fam->Checked[t]==true) {fm_main->SS[i]=clb_Fam->Items->
Strings[t]; fm_main->numb[i]=t; i++;}
   fm_main->kol_fam=i;
minim=0;
 for(t=0;t<fm_main->kol_fam;t++)
 {
  kl=0;
  S="select * from 'DATA\\main.db' where Fam=:F";
       qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
       qr_net_test_boch->ParamByName("F")->AsString=fm_main->SS[t];
       qr_net_test_boch->Open(); qr_net_test_boch->Last();
  while(!qr_net_test_boch->Bof)
  {
   kl++;    qr_net_test_boch->Prior();
  }
   if (minim==0) minim=kl; else if (kl<minim) minim=kl;
 }
lb_min->Caption="Для одного з користувачів є тільки
"+IntToStr(minim)+" записів.";
lb_min->Visible=true;
lb_kol_pol->Caption="Для скількох користувачів тестувати мережу?";
clb_Fam->Enabled=true; bb_boch->Enabled=true; rgr_pr->Enabled=true;
bb_razd_dan->Enabled=true; bb_all->Enabled=true;
bb_no_all->Enabled=true; bb_read->Enabled=true;
bb_net_test->Enabled=true; bb_net_test_boch->Enabled=true;
bb_net_test_boch_1->Enabled=true;ed_boch->Enabled=true;
ed_kol_pol->Enabled=true; rgr_pr->Enabled=true; rgr_usr->Enabled=true;
rgr_fun->Enabled=true; ed_uch->Enabled=true; ed_f_t->Enabled=true;
lb_boch->Visible=false; lb_boch_1->Visible=false; ed_boch->Visible=false;
fm_main->et=false; fm_main->och=0;
lb_kol_pol->Visible=true; ed_kol_pol->Visible=true; rgr_pr->Visible=true;
rgr_pr->ItemIndex=0; rgr_usr->Visible=true; rgr_fun->Visible=true;
bb_boch->Visible=false; rgr_usr->Enabled=true;
lb_uch->Visible=true; ed_uch->Visible=true;
ed_uch->Text=""; ed_kol_pol->Text="";
rgr_usr->ItemIndex=0; rgr_fun->ItemIndex=0;
}
//---------------------------------------------------------------------------
```

```
void __fastcall Tfm_net::clb_FamClickCheck(TObject *Sender)
{
   int t,k=0;
   lb_inf->Visible=false; ed_f->Visible=false; bb_net->Visible=false;
rgr_otb->Visible=false; lb_boch->Visible=false; lb_boch_1->Visible=false;
   ed_boch->Visible=false; lb_min->Visible=false; rgr_pr->Visible=false;
rgr_fun->Visible=false; rgr_usr->Visible=false; lb_inf_t->Visible=false;
   lb_kol_pol->Visible=false; lb_uch->Visible=false; ed_f_t->Visible=false;
ed_kol_pol->Visible=false; ed_uch->Visible=false;  bb_Ok->Visible=false;
   if (fm_main->och==1) rgr_pr->Visible=false;
   bb_boch->Visible=false;
   for (t=0;t<clb_Fam->Items->Count;t++)
      if (clb_Fam->Checked[t]==true) k++;
   if(k<2)
   {
    bb_net_test->Enabled=false; bb_net_test_boch->Enabled=false;
bb_net_test_boch_1->Enabled=false; bb_razd_dan->Enabled=false;
    lb_uch->Visible=false; ed_uch->Visible=false; lb_boch->Visible=false;
lb_boch_1->Visible=false; ed_boch->Visible=false; ed_f_t->Visible=false;
    ed_kol_pol->Visible=false; lb_inf_t->Visible=false;
lb_kol_pol->Visible=false; bb_Ok->Visible=false; rgr_pr->Visible=false;
rgr_usr->Visible=false;     rgr_fun->Visible=false; ed_f_t->Text="";
ed_kol_pol->Text="";
   }
    else
   {bb_net_test->Enabled=true; bb_net_test_boch->Enabled=true;
bb_net_test_boch_1->Enabled=true; bb_razd_dan->Enabled=true;
bb_Ok->Enabled=true;}
   if(k==0) bb_no_all->Enabled=false; else bb_no_all->Enabled=true;
   if(k==clb_Fam->Items->Count) bb_all->Enabled=false;
else bb_all->Enabled=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::ed_f_tChange(TObject *Sender)
{if (ed_kol_pol->Text!="" && ed_uch->Text!="" && ((ed_f_t->Text!="") ||
(rgr_fun->ItemIndex==0))) bb_Ok->Visible=true; else bb_Ok->
Visible=false;}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_OkClick(TObject *Sender)
{
fm_main->prizn=rgr_pr->ItemIndex;
fm_main->uch=StrToInt(ed_uch->Text);
if  (rgr_usr->ItemIndex==0)  fm_main->usr=true;
   else if (rgr_usr->ItemIndex==1)  fm_main->usr=false;
if (fm_main->test_fun==1)
{
if  (StrToFloat(ed_f_t->Text)<=0)
   {
 fm_main->er="   Ширина функції        повинна бути     більша ніж '0' !!!";
    Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else if  (StrToInt(ed_kol_pol->Text)<=0 || StrToInt(ed_kol_pol->Text)>21)
   {
    fm_main->er="Кількість користувачів повинна бути більша ніж '0' і
менша ніж '21' !!!";      Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else if  (StrToInt(ed_uch->Text)<=0 || StrToInt(ed_uch->Text)>(minim-1))
   {
    fm_main->er="Кількість  навчальних даних повинна бути більша ніж
'0' і менша ніж " + IntToStr(minim)+ "!!!";      Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else
{
 fm_main->f_t=StrToFloat(ed_f_t->Text);
 fm_main->poz=StrToInt(ed_kol_pol->Text)-1;
 ed_f_t->Visible=false; ed_kol_pol->Visible=false; lb_inf_t->Visible=false;
lb_kol_pol->Visible=false; bb_boch->Visible=false; bb_Ok->Visible=false;
 lb_uch->Visible=false; ed_uch->Visible=false; rgr_pr->Visible=false;
rgr_usr->Visible=false; rgr_fun->Visible=false; lb_min->Visible=false;
 lb_boch->Visible=false; lb_boch_1->Visible=false; ed_boch->Visible=false;
 ed_f_t->Text=""; ed_kol_pol->Text="";
 fm_net_test->qrp_net_test->Preview();
}
}
else if (fm_main->test_fun==2)
{
if  (StrToInt(ed_kol_pol->Text)<=0 || StrToInt(ed_kol_pol->Text)>21)
   {
    fm_main->er="Кількість користувачів повинна бути більша ніж '0' і
менша ніж '21' !!!";  Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else if  (StrToInt(ed_uch->Text)<=0 || StrToInt(ed_uch->Text)>(minim-1))
   {
    fm_main->er="Кількість навчальних  данни повинна бути більша ніж
'0' і менша ніж" + IntToStr(minim)+ "!!!";      Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else
 {
 fm_main->poz=StrToInt(ed_kol_pol->Text)-1;
 ed_kol_pol->Visible=false; lb_kol_pol->Visible=false;
bb_Ok->Visible=false; rgr_pr->Visible=false; rgr_usr->Visible=false;
rgr_fun->Visible=false;  bb_boch->Visible=false; bb_Ok->Visible=false;
lb_min->Visible=false; lb_uch->Visible=false; ed_uch->Visible=false;
lb_boch->Visible=false;  lb_boch_1->Visible=false; ed_boch->Visible=false;
ed_kol_pol->Text="";
 fm_net_test->qrp_net_test->Preview();
}
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_readClick(TObject *Sender)
{ fm_read->qrp_net_read->Preview(); }
//---------------------------------------------------------------------------
void __fastcall Tfm_net::rgr_funClick(TObject *Sender)
{
if (rgr_fun->ItemIndex==1)
{
ed_f_t->Visible=true; lb_inf_t->Visible=true;
lb_inf_t->Caption="Введіть ширину функції.";
fm_main->test_fun=1; bb_Ok->Visible=false;
}
else if (rgr_fun->ItemIndex==0)
{
ed_f_t->Visible=false; ed_f_t->Text=""; lb_inf_t->Visible=false;
fm_main->test_fun=2;
if (ed_kol_pol->Text!="") bb_Ok->Visible=true; else bb_Ok->Visible=false;
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_allClick(TObject *Sender)
{int t;
   for (t=0;t<clb_Fam->Items->Count;t++)
      clb_Fam->Checked[t]=true;
   bb_net_test->Enabled=true; bb_net_test_boch->Enabled=true;
bb_net_test_boch_1->Enabled=true; bb_razd_dan->Enabled=true; bb_Ok-
>Enabled=true;
   bb_no_all->Enabled=true; bb_all->Enabled=false;
   lb_min->Visible=false; rgr_pr->Visible=false; rgr_fun->Visible=false;
rgr_usr->Visible=false; lb_inf_t->Visible=false; lb_kol_pol->Visible=false;
   lb_uch->Visible=false; ed_f_t->Visible=false; ed_kol_pol->Visible=false;
ed_uch->Visible=false; bb_Ok->Visible=false; lb_boch->Visible=false;
 lb_boch_1->Visible=false; ed_boch->Visible=false; bb_boch->Visible=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_no_allClick(TObject *Sender)
{
int t;
   for (t=0;t<clb_Fam->Items->Count;t++)
      clb_Fam->Checked[t]=false;
   lb_uch->Visible=false; ed_uch->Visible=false;
   bb_net_test->Enabled=false; bb_net_test_boch->Enabled=false;
bb_net_test_boch_1->Enabled=false; bb_razd_dan->Enabled=false;
   bb_Ok->Visible=false;
   bb_no_all->Enabled=false; bb_all->Enabled=true;
   lb_inf->Visible=false; ed_f->Visible=false; bb_net->Visible=false;
rgr_otb->Visible=false; lb_boch->Visible=false; lb_boch_1->Visible=false;
   ed_boch->Visible=false;
   rgr_fun->ItemIndex=0; rgr_pr->ItemIndex=0; rgr_usr->ItemIndex=0;
   rgr_pr->Visible=false; rgr_usr->Visible=false; rgr_fun->Visible=false;
ed_f_t->Visible=false; ed_kol_pol->Visible=false; lb_inf_t->Visible=false;
  lb_kol_pol->Visible=false; bb_boch->Visible=false; lb_min->Visible=false;
   ed_f_t->Text=""; ed_kol_pol->Text="";
}
```

```
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_net_test_bochClick(TObject *Sender)
{
met=1; bb_boch->Visible=false; fm_main->och=1;
lb_min->Visible=false; lb_boch->Visible=true; lb_boch_1->Visible=true;
ed_boch->Visible=true; ed_f_t->Visible=false; ed_kol_pol->Visible=false;
lb_inf_t->Visible=false; lb_kol_pol->Visible=false;
rgr_fun->ItemIndex=0; rgr_pr->ItemIndex=0; rgr_usr->ItemIndex=0;
rgr_pr->Visible=true; rgr_usr->Visible=false; rgr_fun->Visible=false;
lb_uch->Visible=false; ed_uch->Visible=false;
ed_f_t->Text=""; ed_kol_pol->Text=""; ed_boch->Text="";
bb_Ok->Visible=false; lb_min->Visible=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::ed_bochChange(TObject *Sender)
{
if (ed_boch->Text!="") bb_boch->Visible=true; else bb_boch->Visible=false;
ed_f_t->Visible=false; ed_kol_pol->Visible=false; lb_inf_t->Visible=false;
lb_kol_pol->Visible=false; rgr_usr->Visible=false; rgr_fun->Visible=false;
lb_min->Visible=false; bb_boch->Enabled=true;
bb_Ok->Visible=false; lb_uch->Visible=false; ed_uch->Visible=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_bochClick(TObject *Sender)
{
int b, ex, mo, io,i,t,kol_pr,kl,dv,sl,nz=0,n_do=0,n_po=0;
double M[1000][20];
AnsiString S, Fam_All[1000];
char *cs;
AnsiString s1,s2,Slovo[25];
S="delete from 'DATA\\main_promeg.db'";
        qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
        qr_net_test_boch->ExecSQL();
tb_main->Open();tb_main->Last(); tb_main_promeg->Open();
while(!tb_main->Bof)
{
tb_main_promeg->Insert();
for(int np=0;np<31;np++)
   tb_main_promeg->Fields[np]->Value=tb_main->Fields[np]->Value;
tb_main_promeg->Fields[31]->Value=nz;
nz++;
tb_main->Prior();
}
tb_main_promeg->Post(); tb_main_promeg->Close();
if(met==1)
{
minim=0;
if  (StrToFloat(ed_boch->Text)<=0)
   {
    fm_main->er="     Значення повинно бути  більше ніж '0' !!!";
    Beep(400,500);
    Tfm_error *fm_error=new Tfm_error(Application);
   }
else
{
ed_f_t->Visible=false; lb_inf_t->Visible=false; lb_kol_pol->Visible=false;
ed_kol_pol->Visible=false; rgr_usr->Visible=false; rgr_fun->Visible=false;
rgr_fun->ItemIndex=0; ed_kol_pol->Text=""; ed_f_t->Text="";
lb_min->Visible=false; clb_Fam->Enabled=false; bb_boch->Enabled=false;
rgr_pr->Enabled=false; bb_razd_dan->Enabled=false; bb_all->Enabled=false;
bb_no_all->Enabled=false; bb_read->Enabled=false;
bb_net_test->Enabled=false; bb_net_test_boch->Enabled=false;
bb_net_test_boch_1->Enabled=false; ed_boch->Enabled=false;
S="delete from 'DATA\\main_boch.db'";
        qr_net_test_boch->SQL->Clear();qr_net_test_boch->SQL->Add(S);
        qr_net_test_boch->ExecSQL();
fm_main->boch=StrToFloat(ed_boch->Text);
i=0;
   for (t=0;t<clb_Fam->Items->Count;t++)
   if (clb_Fam->Checked[t]==true) {Fam_All[i]=clb_Fam->Items->Strings[t];
fm_main->SS[i]=clb_Fam->Items->Strings[t]; fm_main->numb[i]=t; i++;}
   fm_main->kol_fam=i;
fm_main->prizn=rgr_pr->ItemIndex;
if (fm_main->prizn==0) kol_pr=6;
      else if (fm_main->prizn==1) kol_pr=3;
      else if (fm_main->prizn==2) kol_pr=3;
      else if (fm_main->prizn==3) kol_pr=0;
 tb_main_boch->Open();
 for(t=fm_main->kol_fam-1;t>=0;t--)
 {
kl=0; sl=1;
if (fm_main->prizn==3)
 {
 sl=0;
 S="select DISTINCT Slovo from 'DATA\\main_promeg.db' where Fam=:F";
       qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
       qr_net_test_boch->ParamByName("F")->AsString=Fam_All[t];
       qr_net_test_boch->Open();
  while(!qr_net_test_boch->Eof)
  {
  Slovo[sl++]=qr_net_test_boch->Fields[0]->AsString;
  qr_net_test_boch->Next();
  }
 }
  for(int k_sl=0;k_sl<sl;k_sl++)
  {
  ex=0;
  if (fm_main->prizn==3)
  {
   S="select * from 'DATA\\main_promeg.db' where Fam=:F AND
Slovo=:N";
   qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
   qr_net_test_boch->ParamByName("F")->AsString=Fam_All[t];
   qr_net_test_boch->ParamByName("N")->AsString=Slovo[k_sl];
   qr_net_test_boch->Open();
   kol_pr=StrLen(Slovo[k_sl].c_str());
   }
  else
  {
   S="select * from 'DATA\\main_promeg.db' where Fam=:F";
   qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
   qr_net_test_boch->ParamByName("F")->AsString=Fam_All[t];
   qr_net_test_boch->Open();
  }
  qr_net_test_boch->Last();
  for (b=0;b<kol_pr;b++)
    M[t][b]=0;
  while(!qr_net_test_boch->Bof)
  {
   s1=qr_net_test_boch->Fields[8]->AsString;
   cs=s1.c_str();
   for (b=0;b<kol_pr;b++)
 {
    if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
M[t][b]=M[t][b]+qr_net_test_boch->Fields[8+2+StrLen(cs)-(kol_pr)+b]->
AsInteger;
    else if (fm_main->prizn==2) M[t][b]=M[t][b]+qr_net_test_boch->
Fields[8+2+StrLen(cs)-(2*kol_pr)+b]->AsInteger;
 }
   ex++;
   qr_net_test_boch->Prior();
  }
  for (b=0;b<kol_pr;b++)
  M[t][b]=M[t][b]/ex;
  qr_net_test_boch->Last();
  while(!qr_net_test_boch->Bof)
   {
    s1=qr_net_test_boch->Fields[8]->AsString;
    cs=s1.c_str();
    mo=0;
    for (b=0;b<kol_pr;b++)
     if ((fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3) &&
(fabs(qr_net_test_boch->Fields[8+2+StrLen(cs)-(kol_pr)+b]->AsInteger-
M[t][b])<=fm_main->boch*M[t][b]))
                                mo++;
     else if ((fm_main->prizn==2) && (fabs(qr_net_test_boch->
Fields[8+2+StrLen(cs)-(2*kol_pr)+b]->AsInteger-M[t][b])<=fm_main->
boch*M[t][b]))
                                mo++;
    if(mo==kol_pr)
     {
      kl++;  tb_main_boch->First();  n_do=tb_main_bochID->AsInteger;
      while(!tb_main_boch->Eof && qr_net_test_boch->Fields[31]->
AsInteger<n_do)
      {
       tb_main_boch->Next();  n_do=tb_main_bochID->AsInteger;
      }
     tb_main_boch->Insert(); for (io=0;io<32;io++)
```

```
      tb_main_boch->Fields[io]->Value=qr_net_test_boch->Fields[io]->Value;
      }
    qr_net_test_boch->Prior();
   }
   }
  if (minim==0) minim=kl; else if (kl<minim) minim=kl;
}
tb_main_boch->Last(); n_do=tb_main_bochID->AsInteger;
S="select * from 'DATA\\main_boch.db' where ID=:F";
   qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
   qr_net_test_boch->ParamByName("F")->AsInteger=n_do;
   qr_net_test_boch->Open(); tb_main_boch->First();
while(!tb_main_boch->Eof && tb_main_bochID->AsInteger>n_do)
    {
     tb_main_boch->Next();
    }
tb_main_boch->Insert();
for (io=0;io<32;io++)
    tb_main_boch->Fields[io]->Value=qr_net_test_boch->Fields[io]->Value;
tb_main_boch->Last(); tb_main_boch->Delete();
fm_main->tt="DATA\\main_boch.db";
tb_main_boch->First(); tb_main_boch->Close();
lb_min->Caption="Для одного з користувачів є тільки
"+IntToStr(minim)+" записів.";
lb_min->Visible=true;
clb_Fam->Enabled=true; bb_boch->Enabled=true; rgr_pr->Enabled=true;
bb_razd_dan->Enabled=true; bb_all->Enabled=true; bb_no_all->
Enabled=true;
bb_read->Enabled=true; bb_net_test->Enabled=true; bb_net_test_boch->
Enabled=true; bb_net_test_boch_1->Enabled=true; ed_boch->Enabled=true;
}
}
else if (met==2)
{
int i_d, kpr, k_pr, n_d, nk_d, j_d, ln_d, lf_d, imd, kl_d;
double M_d, S_d, T_d;
AnsiString Imya_d, Otch_d;
struct
       {
        AnsiString Slovo;
        TDateTime Data, Time;
        int N_P, N_Nab, K_Povt, N_Povt, All_Slovo, S[20], Och;
       }
       dan[32000], buf;
fm_main->boch=0;
ed_f_t->Visible=false; lb_inf_t->Visible=false; lb_kol_pol->Visible=false;
ed_kol_pol->Visible=false; rgr_usr->Visible=false; rgr_fun->Visible=false;
rgr_fun->ItemIndex=0;
ed_kol_pol->Text=""; ed_f_t->Text="";
lb_min->Visible=false;
clb_Fam->Enabled=false; bb_boch->Enabled=false; rgr_pr->Enabled=false;
bb_razd_dan->Enabled=false; bb_all->Enabled=false; bb_no_all-
>Enabled=false;
bb_read->Enabled=false; bb_net_test->Enabled=false; bb_net_test_boch-
>Enabled=false; bb_net_test_boch_1->Enabled=false;
S="delete from 'DATA\\main_boch.db'";
          qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
          qr_net_test_boch->ExecSQL();
 minim=0;
 i=0;
    for (t=0;t<clb_Fam->Items->Count;t++)
   if (clb_Fam->Checked[t]==true) {Fam_All[i]=clb_Fam->Items->Strings[t];
fm_main->SS[i]=clb_Fam->Items->Strings[t]; fm_main->numb[i]=t; i++;}
    fm_main->kol_fam=i;
 tb_main_boch->Open();
 for(t=fm_main->kol_fam-1;t>=0;t--)
 {
 i_d=0; kl_d=0;
 S="select * from 'DATA\\main_promeg.db' where Fam=:F";
        qr_net_test_boch->SQL->Clear(); qr_net_test_boch->SQL->Add(S);
        qr_net_test_boch->ParamByName("F")->AsString=Fam_All[t];
        qr_net_test_boch->Open(); qr_net_test_boch->First();
  Imya_d=qr_net_test_boch->Fields[1]->AsString;
  Otch_d=qr_net_test_boch->Fields[2]->AsString;
  while(!qr_net_test_boch->Eof)
  {
   dan[i_d].N_P=i_d;
   dan[i_d].Data=qr_net_test_boch->Fields[3]->AsDateTime;
   dan[i_d].Time=qr_net_test_boch->Fields[4]->AsDateTime;
   dan[i_d].N_Nab=qr_net_test_boch->Fields[5]->AsInteger;
   dan[i_d].K_Povt=qr_net_test_boch->Fields[6]->AsInteger;
   dan[i_d].N_Povt=qr_net_test_boch->Fields[7]->AsInteger;
   dan[i_d].Slovo=qr_net_test_boch->Fields[8]->AsString;
  if (qr_net_test_boch->Fields[9]->AsString!="")
dan[i_d].All_Slovo=qr_net_test_boch->Fields[9]->AsInteger;
  s1=qr_net_test_boch->Fields[8]->AsString;
  cs=s1.c_str();
 for(dv=0;dv<StrLen(cs);dv++)
  if (qr_net_test_boch->Fields[10+dv]->AsString!="")
dan[i_d].S[dv]=qr_net_test_boch->Fields[10+dv]->AsInteger;
   dan[i_d].Och=qr_net_test_boch->Fields[30]->AsInteger;
   i_d++;
   qr_net_test_boch->Next();
  }
fm_main->prizn=rgr_pr->ItemIndex;
if (fm_main->prizn==0) {kol_pr=6; kpr=0;}
       else if (fm_main->prizn==1) {kol_pr=3; kpr=0;}
       else if (fm_main->prizn==2) {kol_pr=3; kpr=1;}
n_d=0; nk_d=i_d;
for(k_pr=0;k_pr<kol_pr;k_pr++)
{
    for(int u=0;u<i_d-1;u++)
     for(int u1=0;u1<i_d-1-u;u1++)
      {
      s1=dan[u1].Slovo;  s2=dan[u1+1].Slovo;
      if(dan[u1].S[StrLen(s1.c_str())-kol_pr-
3*kpr+k_pr]>dan[u1+1].S[StrLen(s2.c_str())-kol_pr-3*kpr+k_pr])
       {buf=dan[u1]; dan[u1]=dan[u1+1]; dan[u1+1]=buf;}
      }
D: j_d=n_d; ln_d=j_d+1; lf_d=nk_d;
D1:M_d=0; S_d=0;
for(imd=ln_d;imd<lf_d;imd++)
{
 s1=dan[imd].Slovo;
 M_d=M_d+dan[imd].S[StrLen(s1.c_str())-kol_pr-3*kpr+k_pr];
}
M_d=M_d/(nk_d-1);
S_d=0;
for(imd=ln_d;imd<lf_d;imd++)
{
 s1=dan[imd].Slovo;
 S_d=S_d+pow((dan[imd].S[StrLen(s1.c_str())-kol_pr-3*kpr+k_pr]-M_d),2);
}
S_d=sqrt(S_d/(nk_d-2));
s1=dan[j_d].Slovo;
T_d=(abs(dan[j_d].S[StrLen(s1.c_str())-kol_pr-3*kpr+k_pr]-M_d))/S_d;
if (T_d>1.960) {if (j_d==ln_d-1) n_d++; else if (j_d==lf_d+1) nk_d--; goto
D;}
    else if(T_d<=1.960 && j_d!=nk_d-1) {j_d=nk_d-1; ln_d=n_d; lf_d=j_d-1;
goto D1;}
}
for(int up=n_d;up<nk_d-1;up++)
     for(int up1=n_d;up1<nk_d-1-up;up1++)
      {
      if(dan[up1].N_P>dan[up1+1].N_P)
       {buf=dan[up1]; dan[up1]=dan[up1+1]; dan[up1+1]=buf;}
      }
tb_main_boch->First();
for(int i_d1=nk_d-1;i_d1>=n_d;i_d1--)
{
tb_main_boch->Insert();
tb_main_bochFam->Value=Fam_All[t]; tb_main_bochImya->Value=Imya_d;
tb_main_bochOtch->Value=Otch_d;
tb_main_bochData->Value=dan[i_d1].Data;
tb_main_bochTime->Value=dan[i_d1].Time;
tb_main_bochN_Nab->AsInteger=dan[i_d1].N_Nab;
tb_main_bochK_Povt->AsInteger=dan[i_d1].K_Povt;
tb_main_bochN_Povt->AsInteger=dan[i_d1].N_Povt;
tb_main_bochSlovo->AsString=dan[i_d1].Slovo;
tb_main_bochAll_Slovo->AsInteger=dan[i_d1].All_Slovo;
 s1=dan[i_d1].Slovo;   cs=s1.c_str();
for(dv=0;dv<StrLen(cs);dv++)
 if(IntToStr(dan[i_d1].S[dv])!="") tb_main_boch->Fields[10+dv]->
AsInteger=dan[i_d1].S[dv];
tb_main_bochOch->AsInteger=dan[i_d1].Och;
kl_d++;
}
tb_main_boch->Post();
```

```
if (minim==0) minim=kl_d; else if (kl_d<minim) minim=kl_d;
 }
fm_main->tt="DATA\\main_boch.db";
tb_main_boch->Close();
lb_min->Caption="Для одного з користувачів є тільки
"+IntToStr(minim)+" записів.";
lb_min->Visible=true;
clb_Fam->Enabled=true; bb_boch->Enabled=true; rgr_pr->Enabled=true;
bb_razd_dan->Enabled=true; bb_all->Enabled=true; bb_no_all->
Enabled=true;
bb_read->Enabled=true; bb_net_test->Enabled=true; bb_net_test_boch->
Enabled=true; bb_net_test_boch_1->Enabled=true;
}
fm_main->test_fun=1; fm_main->et=false;
lb_kol_pol->Visible=true; ed_kol_pol->Visible=true; rgr_usr->Visible=true;
rgr_usr->ItemIndex=1; rgr_usr->Enabled=false; rgr_fun->Visible=true;
lb_kol_pol->Caption="Для скількох користувачів тестувати мережу?";
bb_boch->Enabled=false; lb_uch->Visible=true; ed_uch->Visible=true;
ed_uch->Text="";
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::rgr_prClick(TObject *Sender)
{
if(fm_main->och==1)
{
ed_f_t->Visible=false; ed_kol_pol->Visible=false; lb_inf_t->Visible=false;
lb_kol_pol->Visible=false; rgr_usr->Visible=false; rgr_fun->Visible=false;
lb_min->Visible=false; bb_boch->Enabled=true;
bb_Ok->Visible=false; lb_uch->Visible=false; ed_uch->Visible=false;
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net::bb_net_test_boch_1Click(TObject *Sender)
{
met=2;
bb_boch->Visible=true; bb_boch->Enabled=true;
fm_main->och=1;
lb_min->Visible=false; lb_boch->Visible=false; lb_boch_1->Visible=false;
ed_boch->Visible=false; ed_f_t->Visible=false; ed_kol_pol->Visible=false;
lb_inf_t->Visible=false; lb_kol_pol->Visible=false;
rgr_fun->ItemIndex=0; rgr_pr->ItemIndex=0; rgr_usr->ItemIndex=0;
rgr_pr->Visible=true; rgr_usr->Visible=false; rgr_fun->Visible=false;
lb_uch->Visible=false; ed_uch->Visible=false;
ed_f_t->Text=""; ed_kol_pol->Text="";
bb_Ok->Visible=false; lb_min->Visible=false;
}
//---------------------------------------------------------------------------
```

Файл net_otch.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include <math.h>
#include "net_otch.h"
#include "main.h"
#include "net.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_net_otch *fm_net_otch;
int i,j,kol_u,i_as;
float sh_y,sh_n;
char *cs,*cs1;
AnsiString s1,SS_U[25000],Sl_S[25000];
double ss,ak,Sl[25000][6],Sl_N[25000][6],AS[25000][1000];
double Sl_U[25000][6],Sl_N_U[25000][6];
//---------------------------------------------------------------------------
__fastcall Tfm_net_otch::Tfm_net_otch(TComponent* Owner)
    : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_net_otch::qrp_net_otchBeforePrint(TQuickRep *Sender,
      bool &PrintReport)
{
sh_y=0;
sh_n=0;
int c,v,x,z,imax,y;
double max;
AnsiString S_1,S_2,S_3;
for(i=0;i<25000;i++)
for(z=0;z<6;z++)
  Sl_U[i][z]=0;
i=0;
S_2="select * from 'DATA\\uch_dan.db'";
         qr_net_otch->SQL->Clear();  qr_net_otch->SQL->Add(S_2);
         qr_net_otch->Open();
y=0;
while(!qr_net_otch->Eof)
{
 SS_U[i]=qr_net_otch->Fields[0]->AsString;
 s1=qr_net_otch->Fields[8]->AsString;
 cs=s1.c_str();
 for (j=0;j<6;j++)
 {
  Sl_U[i][j]=Sl_U[i][j]+qr_net_otch->Fields[8+2+StrLen(cs)-6+j]->AsInteger;
 }
 y++;
 if (y==10)
 {
  Sl_S[i]=qr_net_otch->Fields[8]->AsString;
  for(j=0;j<6;j++)
     Sl_U[i][j]=Sl_U[i][j]/y;
  y=0;   ss=0;
  for (j=0;j<6;j++)
       ss=ss+pow(Sl_U[i][j],2);
  for (j=0;j<6;j++)
     Sl_N_U[i][j]=Sl_U[i][j]/sqrt(ss);
  i++;
 }
 qr_net_otch->Next();
}
 kol_u=i; i=0;
S_1="select * from 'DATA\\neiz_ekz.db'";
         qr_net_otch->SQL->Clear(); qr_net_otch->SQL->Add(S_1);
         qr_net_otch->Open();
while(!qr_net_otch->Eof)
{
 for (j=0;j<6;j++)
 {
   s1=qr_net_otch->Fields[8]->AsString;
   cs=s1.c_str();
   Sl[i][j]=qr_net_otch->Fields[8+2+StrLen(cs)-6+j]->AsInteger;
 }
 ss=0;
 for (j=0;j<6;j++)
      ss=ss+pow(Sl[i][j],2);
 for (j=0;j<6;j++)
     Sl_N[i][j]=Sl[i][j]/sqrt(ss);
 for(x=0;x<fm_main->kol_fam;x++)
   AS[i][x]=0;
 if(fm_main->otb==false)
   for(c=0;c<kol_u;c++)
   {
     ak=0;
     for(v=0;v<6;v++)
        ak=ak+Sl_N[i][v]*Sl_N_U[c][v];
     ak=exp((ak-1)/pow(fm_main->f,2));
     for(x=0;x<fm_main->kol_fam;x++)
     {
       s1=SS_U[c];  cs=s1.c_str();
       cs1=fm_main->SS[x].c_str();
       if (strcmp(cs,cs1)==0)    AS[i][x]=AS[i][x]+ak;
     }
   }
   else if (fm_main->otb==true)
     for(c=0;c<kol_u;c++)
      {
       s1=Sl_S[c]; cs=s1.c_str();
       s1=qr_net_otch->Fields[8]->AsString;
       cs1=s1.c_str();
       if (strcmp(cs,cs1)==0)
       {
        ak=0;
        for(v=0;v<6;v++)
          ak=ak+Sl_N[i][v]*Sl_N_U[c][v];
        ak=exp((ak-1)/pow(fm_main->f,2));
        for(x=0;x<fm_main->kol_fam;x++)
        {
```

```
          s1=SS_U[c]; cs=s1.c_str();
          cs1=fm_main->SS[x].c_str();
          if (strcmp(cs,cs1)==0)    AS[i][x]=AS[i][x]+ak;
        }
       }
      }
 max=AS[i][0];   imax=0;
 for(z=0;z<fm_main->kol_fam;z++)
   if (AS[i][z]>max) {max=AS[i][z]; imax=z;}
s1=qr_net_otch->Fields[0]->AsString;
cs=s1.c_str();
cs1=fm_main->SS[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y++; else sh_n++;
i++;
qr_net_otch->Next();
}
S_3="select DISTINCT Fam from 'DATA\\uch_dan.db'";
        qr_net_otch_1->SQL->Clear(); qr_net_otch_1->SQL->Add(S_3);
        qr_net_otch_1->Open();
i=0;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net_otch::QRSubDetail1BeforePrint(
     TQRCustomBand *Sender, bool &PrintBand)
{
QRLabel1->Caption=Sl[i][0]; QRLabel2->Caption=Sl[i][1];
QRLabel3->Caption=Sl[i][2]; QRLabel4->Caption=Sl[i][3];
QRLabel5->Caption=Sl[i][4]; QRLabel6->Caption=Sl[i][5];
i++; i_as=0;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net_otch::QRSubDetail2BeforePrint(
     TQRCustomBand *Sender, bool &PrintBand)
{
 QRLabel20->Caption=AS[i-1][i_as];  i_as++;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_net_otch::QRBand1BeforePrint(TQRCustomBand
*Sender,
     bool &PrintBand)
{
QRLabel16->Caption=sh_y; QRLabel17->Caption=sh_n;
QRLabel7->Caption=FloatToStr(100*sh_y/(sh_y+sh_n)) + " %";
QRLabel8->Caption=FloatToStr(100*sh_n/(sh_y+sh_n)) + " %";
}
//---------------------------------------------------------------------------
```

Файл net_test.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include <math.h>
#include "net_test.h"
#include "net.h"
#include "main.h"
#include "analiz.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_net_test *fm_net_test;
int i1,j1,kol_u,hhh,kol_uch_kl,rrr,qqq,www,nnn[1000];
float sh_y,sh_n,sh_y_o,sh_n_o;
AnsiString Fam_Vyb[1000],Fam_All[1000];
char *cs,*cs1;
AnsiString s1,SS_U_1[32000],Sl_S_1[32000];
double ss,ak,eee,Sl_1[32000][20],Sl_N_1[32000][20],AS_1[32000][1000],
AS_1_O[32000][1000];
double Sl_U_1[32000][20],Sl_N_U_1[32000][20];
//---------------------------------------------------------------------------
__fastcall Tfm_net_test::Tfm_net_test(TComponent* Owner)
    : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_net_test::qrp_net_testBeforePrint(TQuickRep *Sender,
     bool &PrintReport)
{
 int i=0,p;
 AnsiString S,S_1;
 int t,kol_pol;
 int KOMB[1000],j,h,d,z,b,kol_pr=0,s=0,sl;
 int ud,k,x,z1,kol=10;
 AnsiString slovo,fam,prz,usred;
 int c1,v1,x1,z11,imax,y,max_y=0,max_y_o=0;
 double fff, fff_o, fun, fun_o;
 bool all=false, all_v=false;
double max;
AnsiString S_11,S_12,S_13,Slovo[100];
kol_pol=fm_main->kol_fam;
for(t=0;t<kol_pol;t++)
   Fam_All[t]=fm_main->SS[t];
hhh=0;
if (fm_main->prizn==0) {prz="6"; kol_pr=6;}
       else if (fm_main->prizn==1) {prz="3 (останні)"; kol_pr=3;}
       else if (fm_main->prizn==2) {prz="3 (4, 5, 6 з кінця)"; kol_pr=3;}
       else if (fm_main->prizn==3)
       {
       for(j=kol_pol-1;j>=0;j--)
{
S="select DISTINCT Slovo from '"+fm_main->tt+"' where Fam=:F";
        qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S);
        qr_net_test_1->ParamByName("F")->AsString=Fam_All[j];
        qr_net_test_1->Open();
if(j==kol_pol-1)
 while(!qr_net_test_1->Eof)
 {
  Slovo[s]=qr_net_test_1->Fields[0]->AsString; s++;  qr_net_test_1->Next();
 }
else
 for(k=0;k<s;k++)
  {
  qr_net_test_1->First();   sl=0;
  while(!qr_net_test_1->Eof && sl==0)
   {
    if(Slovo[k]==qr_net_test_1->Fields[0]->AsString) sl++;
    qr_net_test_1->Next();
   }
   if(sl==0)  Slovo[k]="";
  }
}
for(k=0;k<s;k++)
 if (Slovo[k]!="")
 {
   if (kol_pr==0) kol_pr=StrLen(Slovo[k].c_str());
   else if (StrLen(Slovo[k].c_str())<kol_pr) kol_pr=StrLen(Slovo[k].c_str());
 }
prz=kol_pr;
       }
if (fm_main->usr==true) usred="(з усередненням). ";
else if (fm_main->usr==false) usred="(без усереднення). ";
kol_uch_kl=fm_main->uch;
if (fm_main->test_fun==1)
{
QRLabel11->Caption="Навчальні дані складаються з " +
IntToStr(kol_uch_kl) + " екземпляров кожного класу  " + usred +
"Кількість ознак дорівнює " + prz + ". Ширина функції дорівнює " +
FloatToStr(fm_main->f_t) + ". Відсоток помилок - середній.";
QRLabel23->Caption=" Навчальні дані складаються з " +
IntToStr(kol_uch_kl) + " екземпляров кожного класу  " + usred + "
Кількість ознак дорівнює " + prz + ". Ширина функції дорівнює " +
FloatToStr(fm_main->f_t) + ". Відсоток помилок - середній.";
}
else if (fm_main->test_fun==2)
{
QRLabel11->Caption="Навчальні дані складаються з " +
IntToStr(kol_uch_kl) + " екземпляров кожного класу  " + usred +
"Кількість ознак дорівнює " + prz + ". Ширина функції підбирається.
Відсоток помилок - середній.";
QRLabel23->Caption="Навчальні дані складаються з " +
IntToStr(kol_uch_kl) + " екземпляров кожного класу  " + usred +
"Кількість ознак дорівнює " + prz + ". Ширина функції підбирається.
Відсоток помилок - середній.";
}
if (fm_main->et==true)
{
  QRLabel11->Caption=QRLabel11->Caption + " Використовувалась
багатоетапна мережа.";
  QRLabel23->Caption=QRLabel23->Caption + " Використовувалась
багатоетапна мережа.";
```

```
}
if (fm_main->och==1)
{
 QRLabel11->Caption=QRLabel11->Caption + " З виключенням грубих
помилок (" + fm_main->boch + ").";
 QRLabel23->Caption=QRLabel23->Caption + " З виключенням грубих
помилок (" + fm_main->boch + ").";
}
S="delete from 'DATA\\vyborka.db'";
          qr_net_test->SQL->Clear(); qr_net_test->SQL->Add(S);
          qr_net_test->ExecSQL();
 tb_vyborka->Open();  tb_vyborka->Refresh();
 sh_y=0;  sh_n=0;  sh_y_o=0;  sh_n_o=0;
 S_1="select DISTINCT Fam from '"+fm_main->tt+"'";
        qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S_1);
        qr_net_test_1->Open();
if (fm_main->et==false)
{
 for(i=fm_main->poz;i<(fm_main->poz+1);i++)
 {
  for(z=0;z<=i;z++)
      KOMB[z]=z;
    a1: for(d=KOMB[i-1]+1;d<kol_pol;d++)
    {
      b=i;
      S="delete from 'DATA\\uch_dan.db'";
         qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S);
         qr_net_test_1->ExecSQL();
      S="delete from 'DATA\\neiz_ekz.db'";
         qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S);
         qr_net_test_1->ExecSQL();
      tb_vyborka->First();  tb_vyborka->Insert();
      tb_vyborkaN_Var->AsInteger=hhh+1;
      tb_vyborkaKol_Fam->AsInteger=i+1;
      for(j=0;j<=i;j++)
      {
       Fam_Vyb[j]=Fam_All[KOMB[j]];
       tb_vyborkaFam->AsString=tb_vyborkaFam->AsString + Fam_Vyb[j] +
" (" + IntToStr(fm_main->numb[KOMB[j]]+1) + ")" + " ";
      }
       for (j=0;j<=i;j++)
   {
   S="select * from '"+fm_main->tt+"' where Fam=:F";
        qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S);
        qr_net_test_1->ParamByName("F")->AsString=Fam_Vyb[j];
        qr_net_test_1->Open();
   ud=0;
   tb_tabl_p_1->TableName="DATA\\uch_dan.db";
   tb_tabl_p_1->Active=true;     tb_tabl_p_1->Open();
   tb_tabl_p_2->TableName="DATA\\neiz_ekz.db";
   tb_tabl_p_2->Active=true;     tb_tabl_p_2->Open();
   while(!qr_net_test_1->Eof && ud<=(kol_uch_kl-1))
   {
    x=0;
    if (qr_net_test_1->Fields[7]->AsInteger==1)
    {
     slovo=qr_net_test_1->Fields[8]->AsString;
     fam=qr_net_test_1->Fields[0]->AsString;
     while(!qr_net_test_1->Eof && ud<=(kol_uch_kl-1) && x<kol &&
qr_net_test_1->Fields[7]->AsInteger==(x+1)  && qr_net_test_1->Fields[8]->
AsString==slovo && qr_net_test_1->Fields[0]->AsString==fam)
      {
       x++;  qr_net_test_1->Next();
      }
     for(z1=0;z1<x;z1++)
      qr_net_test_1->Prior();
    }
    if(x==kol)
    {
     for (z1=0;z1<kol;z1++)
     {
      tb_tabl_p_1->First();      tb_tabl_p_1->Insert();
      for (k=0;k<31;k++)
       tb_tabl_p_1->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();
      ud++;
     }
      tb_tabl_p_1->Post();
    }
    else
    {
     for (z1=0;z1<=x;z1++)
     {
      tb_tabl_p_2->First();  tb_tabl_p_2->Insert();
      for (k=0;k<31;k++)
       tb_tabl_p_2->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();
     }
     tb_tabl_p_2->Next();
    }
   }
   if(ud<=(kol_uch_kl-1))
   {
     tb_tabl_p_2->Close();  tb_tabl_p_2->Open();  tb_tabl_p_2->Last();
     for (p=0;p<=(kol_uch_kl-1-ud);p++)
      {
       while(!tb_tabl_p_2->Bof && tb_tabl_p_2->Fields[0]->
AsString!=Fam_Vyb[j])
        tb_tabl_p_2->Prior();  tb_tabl_p_1->First();  tb_tabl_p_1->Insert();
        for (k=0;k<31;k++)
         tb_tabl_p_1->Fields[k]->AsString=tb_tabl_p_2->Fields[k]->AsString;
        tb_tabl_p_2->Delete();
       }
      tb_tabl_p_2->Prior();  tb_tabl_p_2->Close();  tb_tabl_p_1->Post();
   }
   tb_tabl_p_1->Close();
   if (ud>(kol_uch_kl-1))
     {
      while(!qr_net_test_1->Eof)
      {
      tb_tabl_p_2->First();  tb_tabl_p_2->Insert();
      for (k=0;k<31;k++)
       tb_tabl_p_2->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();
      }
      tb_tabl_p_2->Post(); tb_tabl_p_2->Close();
     }
   }
for(i1=0;i1<25000;i1++)
for(z11=0;z11<kol_pr;z11++)
  Sl_U_1[i1][z11]=0;
i1=0;
S_12="select * from 'DATA\\uch_dan.db'";
        qr_net_test_2->SQL->Clear();  qr_net_test_2->SQL->Add(S_12);
        qr_net_test_2->Open();
y=0;
while(!qr_net_test_2->Eof)
{
 SS_U_1[i1]=qr_net_test_2->Fields[0]->AsString;
 s1=qr_net_test_2->Fields[8]->AsString;  cs=s1.c_str();
if (fm_main->usr==true)
{
 for (j1=0;j1<kol_pr;j1++)
 {
  if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_U_1[i1][j1]=Sl_U_1[i1][j1]+qr_net_test_2->Fields[8+2+StrLen(cs)-
(kol_pr)+j1]->AsInteger;
    else if (fm_main->prizn==2)
Sl_U_1[i1][j1]=Sl_U_1[i1][j1]+qr_net_test_2->Fields[8+2+StrLen(cs)-
(2*kol_pr)+j1]->AsInteger;
 }
 y++;
 if (y==10)
 {
  Sl_S_1[i1]=qr_net_test_2->Fields[8]->AsString;
  for(j1=0;j1<kol_pr;j1++)
      Sl_U_1[i1][j1]=Sl_U_1[i1][j1]/y;
  y=0;   ss=0;
  for (j1=0;j1<kol_pr;j1++)
      ss=ss+pow(Sl_U_1[i1][j1],2);
  for (j1=0;j1<kol_pr;j1++)
     Sl_N_U_1[i1][j1]=Sl_U_1[i1][j1]/sqrt(ss);
  i1++;
 }
}
   else if (fm_main->usr==false)
{
 Sl_S_1[i1]=qr_net_test_2->Fields[8]->AsString;
```

```
 for(j1=0;j1<kol_pr;j1++)
 if (fm_main->prizn==0 || fm_main->prizn==1|| fm_main->prizn==3)
Sl_U_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->
AsInteger;
    else if (fm_main->prizn==2) Sl_U_1[i1][j1]=qr_net_test_2->
Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
 ss=0;
 for (j1=0;j1<kol_pr;j1++)
     ss=ss+pow(Sl_U_1[i1][j1],2);
 for (j1=0;j1<kol_pr;j1++)
    Sl_N_U_1[i1][j1]=Sl_U_1[i1][j1]/sqrt(ss);
 i1++;
}
 qr_net_test_2->Next();
}
 kol_u=i1;
if (fm_main->test_fun==1)
{
i1=0;
S_11="select * from 'DATA\\neiz_ekz.db'";
        qr_net_test_2->SQL->Clear();  qr_net_test_2->SQL->Add(S_11);
        qr_net_test_2->Open();
while(!qr_net_test_2->Eof)
{
 for (j1=0;j1<kol_pr;j1++)
 {
   s1=qr_net_test_2->Fields[8]->AsString;     cs=s1.c_str();
   if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->AsInteger;
      else if (fm_main->prizn==2) Sl_1[i1][j1]=qr_net_test_2->
Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
 }
 ss=0;
 for (j1=0;j1<kol_pr;j1++)
     ss=ss+pow(Sl_1[i1][j1],2);
 for (j1=0;j1<kol_pr;j1++)
    Sl_N_1[i1][j1]=Sl_1[i1][j1]/sqrt(ss);
 for(x1=0;x1<=i;x1++)
   {
    AS_1[i1][x1]=0;     AS_1_O[i1][x1]=0;
   }
   for(c1=0;c1<kol_u;c1++)
   {
    ak=0;
    for(v1=0;v1<kol_pr;v1++)
       ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
    ak=exp((ak-1)/pow(fm_main->f_t,2));
    for(x1=0;x1<=i;x1++)
    {
     s1=SS_U_1[c1];  cs=s1.c_str();
     cs1=Fam_Vyb[x1].c_str();
     if (strcmp(cs,cs1)==0)    AS_1[i1][x1]=AS_1[i1][x1]+ak;
    }
   }
 max=AS_1[i1][0];  imax=0;
 for(z11=0;z11<=i;z11++)
   if (AS_1[i1][z11]>max) {max=AS_1[i1][z11]; imax=z11;}
s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y++; else sh_n++;
    for(c1=0;c1<kol_u;c1++)
     {
      s1=Sl_S_1[c1];  cs=s1.c_str();
      s1=qr_net_test_2->Fields[8]->AsString;
      cs1=s1.c_str();
      if (strcmp(cs,cs1)==0)
      {
       ak=0;
       for(v1=0;v1<kol_pr;v1++)
         ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
       ak=exp((ak-1)/pow(fm_main->f_t,2));
       for(x1=0;x1<=i;x1++)
       {
         s1=SS_U_1[c1];  cs=s1.c_str(); cs1=Fam_Vyb[x1].c_str();
         if (strcmp(cs,cs1)==0)    AS_1_O[i1][x1]=AS_1_O[i1][x1]+ak;
       }
      }
     }
 max=AS_1_O[i1][0];   imax=0;
 for(z11=0;z11<=i;z11++)
   if (AS_1_O[i1][z11]>max) {max=AS_1_O[i1][z11]; imax=z11;}
 s1=qr_net_test_2->Fields[0]->AsString;
 cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
 if (strcmp(cs,cs1)==0) sh_y_o++; else sh_n_o++;
 i1++;
 qr_net_test_2->Next();
}
}
else if (fm_main->test_fun==2)
{
 max_y=0;  max_y_o=0;
ff:for(fff_o=0.01,fff=0.01;fff<=1    &&
all_v==false;fff=fff+0.01,fff_o=fff_o+0.01)
{
sh_y=0; sh_n=0; sh_y_o=0; sh_n_o=0;
if(all==true) {fff=fun; fff_o=fun_o; all_v=true;}
i1=0;
S_11="select * from 'DATA\\neiz_ekz.db'";
        qr_net_test_2->SQL->Clear(); qr_net_test_2->SQL->Add(S_11);
        qr_net_test_2->Open();
while(!qr_net_test_2->Eof)
{
 for (j1=0;j1<kol_pr;j1++)
 {
   s1=qr_net_test_2->Fields[8]->AsString;  cs=s1.c_str();
   if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->AsInteger;
      else if (fm_main->prizn==2) Sl_1[i1][j1]=qr_net_test_2-
>Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
 }
 ss=0;
 for (j1=0;j1<kol_pr;j1++)
     ss=ss+pow(Sl_1[i1][j1],2);
 for (j1=0;j1<kol_pr;j1++)
    Sl_N_1[i1][j1]=Sl_1[i1][j1]/sqrt(ss);
 for(x1=0;x1<=i;x1++)
   {
    AS_1[i1][x1]=0;  AS_1_O[i1][x1]=0;
   }
   for(c1=0;c1<kol_u;c1++)
   {
    ak=0;
    for(v1=0;v1<kol_pr;v1++)
       ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
    ak=exp((ak-1)/pow(fff,2));
    for(x1=0;x1<=i;x1++)
    {
     s1=SS_U_1[c1];  cs=s1.c_str();
     cs1=Fam_Vyb[x1].c_str();
     if (strcmp(cs,cs1)==0)    AS_1[i1][x1]=AS_1[i1][x1]+ak;
    }
   }
 max=AS_1[i1][0];   imax=0;
 for(z11=0;z11<=i;z11++)
   if (AS_1[i1][z11]>max) {max=AS_1[i1][z11]; imax=z11;}
s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y++; else sh_n++;
    for(c1=0;c1<kol_u;c1++)
     {
      s1=Sl_S_1[c1];     cs=s1.c_str();
      s1=qr_net_test_2->Fields[8]->AsString;
      cs1=s1.c_str();
      if (strcmp(cs,cs1)==0)
      {
       ak=0;
       for(v1=0;v1<kol_pr;v1++)
         ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
       ak=exp((ak-1)/pow(fff_o,2));
       for(x1=0;x1<=i;x1++)
       {
         s1=SS_U_1[c1];  cs=s1.c_str();  cs1=Fam_Vyb[x1].c_str();
         if (strcmp(cs,cs1)==0)    AS_1_O[i1][x1]=AS_1_O[i1][x1]+ak;
       }
      }
     }
 max=AS_1_O[i1][0];  imax=0;
 for(z11=0;z11<=i;z11++)
```

```
   if (AS_1_O[i1][z11]>max) {max=AS_1_O[i1][z11]; imax=z11;}
 s1=qr_net_test_2->Fields[0]->AsString;
 cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
 if (strcmp(cs,cs1)==0) sh_y_o++; else sh_n_o++;
 i1++;
 qr_net_test_2->Next();
}
          if (sh_y>max_y) {max_y=sh_y; fun=fff;}
          if (sh_y_o>max_y_o) {max_y_o=sh_y_o; fun_o=fff;}
        }
        if (all_v==true)
           {tb_vyborkaFun->AsFloat=fun; tb_vyborkaFun_O->AsFloat=fun_o;
all=false; all_v=false;}
        else { all=true; goto ff;}
}
        tb_vyborkaPrav->AsInteger=sh_y;
        tb_vyborkaOchib->AsInteger=sh_n;
        tb_vyborkaProc_Prav->AsFloat=100*sh_y/(sh_y+sh_n);
        tb_vyborkaProc_Ochib->AsFloat=100*sh_n/(sh_y+sh_n);
        tb_vyborkaPrav_O->AsInteger=sh_y_o;
        tb_vyborkaOchib_O->AsInteger=sh_n_o;
        tb_vyborkaProc_Prav_O->AsFloat=100*sh_y_o/(sh_y_o+sh_n_o);
        tb_vyborkaProc_Ochib_O->AsFloat=100*sh_n_o/(sh_y_o+sh_n_o);
        tb_vyborka->Post();
        hhh++;  sh_y=0;  sh_n=0;  sh_y_o=0;  sh_n_o=0;  KOMB[i]++;
      }
      a2: if (b>0)
      {
       if(KOMB[b-1]<(kol_pol-2-(i-b)))
       {
        KOMB[b-1]++;
        for(z=b;z<=i;z++)
         KOMB[z]=KOMB[z-1]+1;
        if(b<i) b++;
        goto a1;
       }
       else
        {
         b--;  goto a2;
        }
      }
    }
i1=0; tb_vyborka->Close();
S="select Kol_Fam, avg(Proc_Prav), avg(Proc_Ochib), avg(Proc_Prav_O),
avg(Proc_Ochib_O) from 'DATA\\vyborka.db' GROUP BY Kol_Fam";
          qr_net_test->SQL->Clear();  qr_net_test->SQL->Add(S);
          qr_net_test->Open();
QRLabel20->Caption=" из " + IntToStr(kol_pol);
tb_vyborka->Open();
}
else if (fm_main->et==true)
{
 for(i=fm_main->poz;i<(fm_main->poz+1);i++)
 {
  for(z=0;z<=i;z++)
      KOMB[z]=z;
     a11: for(d=KOMB[i-1]+1;d<kol_pol;d++)
     {
       b=i;
       S="delete from 'DATA\\uch_dan.db'";
          qr_net_test_1->SQL->Clear(); qr_net_test_1->SQL->Add(S);
          qr_net_test_1->ExecSQL();
       S="delete from 'DATA\\neiz_ekz.db'";
          qr_net_test_1->SQL->Clear();  qr_net_test_1->SQL->Add(S);
          qr_net_test_1->ExecSQL();
       tb_vyborka->First();  tb_vyborka->Insert();
       tb_vyborkaN_Var->AsInteger=hhh+1;
       tb_vyborkaKol_Fam->AsInteger=i+1;
       for(j=0;j<=i;j++)
       {
        Fam_Vyb[j]=Fam_All[KOMB[j]];
        tb_vyborkaFam->AsString=tb_vyborkaFam->AsString + Fam_Vyb[j] +
" (" + IntToStr(KOMB[j]+1) + ")" + " ";
       }
        for (j=0;j<=i;j++)
   {
   S="select * from '"+fm_main->tt+"' where Fam=:F";
        qr_net_test_1->SQL->Clear();  qr_net_test_1->SQL->Add(S);
        qr_net_test_1->ParamByName("F")->AsString=Fam_Vyb[j];
        qr_net_test_1->Open();
   ud=0;
   tb_tabl_p_1->TableName="DATA\\uch_dan.db";
   tb_tabl_p_1->Active=true;  tb_tabl_p_1->Open();
   tb_tabl_p_2->TableName="DATA\\neiz_ekz.db";
   tb_tabl_p_2->Active=true; tb_tabl_p_2->Open();
   while(!qr_net_test_1->Eof && ud<=(kol_uch_kl-1))
   {
    x=0;
    if (qr_net_test_1->Fields[7]->AsInteger==1)
    {
     slovo=qr_net_test_1->Fields[8]->AsString;
     fam=qr_net_test_1->Fields[0]->AsString;
     while(!qr_net_test_1->Eof && ud<=(kol_uch_kl-1) && x<kol &&
qr_net_test_1->Fields[7]->AsInteger==(x+1) && qr_net_test_1->Fields[8]->
AsString==slovo && qr_net_test_1->Fields[0]->AsString==fam)
     {
      x++;   qr_net_test_1->Next();
     }
     for(z1=0;z1<x;z1++)
      qr_net_test_1->Prior();
    }
    if(x==kol)
    {
     for (z1=0;z1<kol;z1++)
     {
      tb_tabl_p_1->First();  tb_tabl_p_1->Insert();
      for (k=0;k<31;k++)
       tb_tabl_p_1->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();      ud++;
     }
    }
    else
    {
     for (z1=0;z1<=x;z1++)
     {
      tb_tabl_p_2->First();  tb_tabl_p_2->Insert();
      for (k=0;k<31;k++)
       tb_tabl_p_2->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();
     }
    }
   }
   tb_tabl_p_1->Post();  tb_tabl_p_1->Close();
   if (ud>(kol_uch_kl-1))
     {
      while(!qr_net_test_1->Eof)
      {
      tb_tabl_p_2->First();  tb_tabl_p_2->Insert();
      for (k=0;k<31;k++)
      tb_tabl_p_2->Fields[k]->AsString=qr_net_test_1->Fields[k]->AsString;
      qr_net_test_1->Next();
      }
      tb_tabl_p_2->Post();  tb_tabl_p_2->Close();
     }
   }
for(i1=0;i1<25000;i1++)
for(z11=0;z11<kol_pr;z11++)
  Sl_U_1[i1][z11]=0;
i1=0;
S_12="select * from 'DATA\\uch_dan.db'";
          qr_net_test_2->SQL->Clear();  qr_net_test_2->SQL->Add(S_12);
          qr_net_test_2->Open();
y=0;
while(!qr_net_test_2->Eof)
{
 SS_U_1[i1]=qr_net_test_2->Fields[0]->AsString;
 s1=qr_net_test_2->Fields[8]->AsString;
 cs=s1.c_str();
if (fm_main->usr==true)
{
 for (j1=0;j1<kol_pr;j1++)
 {
  if (fm_main->prizn==0 || fm_main->prizn==1  || fm_main->prizn==3)
Sl_U_1[i1][j1]=Sl_U_1[i1][j1]+qr_net_test_2->Fields[8+2+StrLen(cs)-
(kol_pr)+j1]->AsInteger;
    else if (fm_main->prizn==2)
Sl_U_1[i1][j1]=Sl_U_1[i1][j1]+qr_net_test_2->Fields[8+2+StrLen(cs)-
(2*kol_pr)+j1]->AsInteger;
```

```
 }
 y++;
 if (y==10)
 {
  Sl_S_1[i1]=qr_net_test_2->Fields[8]->AsString;
  for(j1=0;j1<kol_pr;j1++)
      Sl_U_1[i1][j1]=Sl_U_1[i1][j1]/y;
  y=0;   ss=0;
  for (j1=0;j1<kol_pr;j1++)
      ss=ss+pow(Sl_U_1[i1][j1],2);
  for (j1=0;j1<kol_pr;j1++)
     Sl_N_U_1[i1][j1]=Sl_U_1[i1][j1]/sqrt(ss);
  i1++;
 }
}
   else if (fm_main->usr==false)
{
  Sl_S_1[i1]=qr_net_test_2->Fields[8]->AsString;
  for(j1=0;j1<kol_pr;j1++)
  if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_U_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->
AsInteger;
    else if (fm_main->prizn==2) Sl_U_1[i1][j1]=qr_net_test_2->
Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
  ss=0;
  for (j1=0;j1<kol_pr;j1++)
      ss=ss+pow(Sl_U_1[i1][j1],2);
  for (j1=0;j1<kol_pr;j1++)
     Sl_N_U_1[i1][j1]=Sl_U_1[i1][j1]/sqrt(ss);
  i1++;
}
 qr_net_test_2->Next();
}
 kol_u=i1;
if (fm_main->test_fun==1)
{
i1=0;
S_11="select * from 'DATA\\neiz_ekz.db'";
        qr_net_test_2->SQL->Clear(); qr_net_test_2->SQL->Add(S_11);
        qr_net_test_2->Open();
while(!qr_net_test_2->Eof)
{
 for (j1=0;j1<kol_pr;j1++)
 {
   s1=qr_net_test_2->Fields[8]->AsString;  cs=s1.c_str();
   if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->AsInteger;
      else if (fm_main->prizn==2) Sl_1[i1][j1]=qr_net_test_2->
Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
 }
  ss=0;
  for (j1=0;j1<kol_pr;j1++)
      ss=ss+pow(Sl_1[i1][j1],2);
  for (j1=0;j1<kol_pr;j1++)
     Sl_N_1[i1][j1]=Sl_1[i1][j1]/sqrt(ss);
  for(x1=0;x1<=i;x1++)
    {
     AS_1[i1][x1]=0; AS_1_O[i1][x1]=0;
    }
    for(c1=0;c1<kol_u;c1++)
    {
      ak=0;
      for(v1=0;v1<kol_pr;v1++)
         ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
      ak=exp((ak-1)/pow(fm_main->f_t,2));
      for(x1=0;x1<=i;x1++)
      {
        s1=SS_U_1[c1]; cs=s1.c_str();
        cs1=Fam_Vyb[x1].c_str();
        if (strcmp(cs,cs1)==0)    AS_1[i1][x1]=AS_1[i1][x1]+ak;
      }
    }
  for(qqq=0;qqq<=i;qqq++)
    nnn[qqq]=qqq;
  for(qqq=0;qqq<=i;qqq++)
    for(www=0;www<=i-qqq;www++)
       if(AS_1[i1][www]<AS_1[i1][www+1])
       {
        eee=AS_1[i1][www];  AS_1[i1][www]=AS_1[i1][www+1];
        AS_1[i1][www+1]=eee;   rrr=nnn[www];
        nnn[www]=nnn[www+1];  nnn[www+1]=rrr;
       }
 imax=nnn[0];  s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y++; else sh_n++;
     for(c1=0;c1<kol_u;c1++)
      {
       s1=Sl_S_1[c1];  cs=s1.c_str();
       s1=qr_net_test_2->Fields[8]->AsString; cs1=s1.c_str();
       if (strcmp(cs,cs1)==0)
       {
        ak=0;
        for(v1=0;v1<kol_pr;v1++)
          ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
        ak=exp((ak-1)/pow(fm_main->f_t,2));
        for(x1=0;x1<=i;x1++)
        {
          s1=SS_U_1[c1];  cs=s1.c_str();
          cs1=Fam_Vyb[x1].c_str();
          if (strcmp(cs,cs1)==0)    AS_1_O[i1][x1]=AS_1_O[i1][x1]+ak;
        }
       }
      }
  for(qqq=0;qqq<=i;qqq++)
    nnn[qqq]=qqq;
  for(qqq=0;qqq<=i;qqq++)
    for(www=0;www<=i-qqq;www++)
       if(AS_1_O[i1][www]<AS_1_O[i1][www+1])
       {
        eee=AS_1_O[i1][www]; AS_1_O[i1][www]=AS_1_O[i1][www+1];
        AS_1_O[i1][www+1]=eee; rrr=nnn[www];
        nnn[www]=nnn[www+1]; nnn[www+1]=rrr;
       }
 imax=nnn[0];  s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
 if (strcmp(cs,cs1)==0) sh_y_o++; else sh_n_o++;
 i1++;
 qr_net_test_2->Next();
}
}
else if (fm_main->test_fun==2)
{
 max_y=0;  max_y_o=0;
ff22:for(fff_o=0.01,fff=0.01;fff<=1 &&
all_v==false;fff=fff+0.01,fff_o=fff_o+0.01)
{
sh_y=0; sh_n=0; sh_y_o=0; sh_n_o=0;
if(all==true) {fff=fun; fff_o=fun_o; all_v=true;}
i1=0;
S_11="select * from 'DATA\\neiz_ekz.db'";
        qr_net_test_2->SQL->Clear();  qr_net_test_2->SQL->Add(S_11);
        qr_net_test_2->Open();
while(!qr_net_test_2->Eof)
{
 for (j1=0;j1<kol_pr;j1++)
 {
   s1=qr_net_test_2->Fields[8]->AsString;     cs=s1.c_str();
   if (fm_main->prizn==0 || fm_main->prizn==1 || fm_main->prizn==3)
Sl_1[i1][j1]=qr_net_test_2->Fields[8+2+StrLen(cs)-(kol_pr)+j1]->AsInteger;
      else if (fm_main->prizn==2) Sl_1[i1][j1]=qr_net_test_2->
Fields[8+2+StrLen(cs)-(2*kol_pr)+j1]->AsInteger;
 }
  ss=0;
  for (j1=0;j1<kol_pr;j1++)
      ss=ss+pow(Sl_1[i1][j1],2);
  for (j1=0;j1<kol_pr;j1++)
     Sl_N_1[i1][j1]=Sl_1[i1][j1]/sqrt(ss);
  for(x1=0;x1<=i;x1++)
    {
     AS_1[i1][x1]=0; AS_1_O[i1][x1]=0;
    }
    for(c1=0;c1<kol_u;c1++)
    {
      ak=0;
      for(v1=0;v1<kol_pr;v1++)
         ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
      ak=exp((ak-1)/pow(fff,2));
      for(x1=0;x1<=i;x1++)
```

```
    {
     s1=SS_U_1[c1];  cs=s1.c_str(); cs1=Fam_Vyb[x1].c_str();
     if (strcmp(cs,cs1)==0)    AS_1[i1][x1]=AS_1[i1][x1]+ak;
    }
   }
 for(qqq=0;qqq<=i;qqq++)
  nnn[qqq]=qqq;
 for(qqq=0;qqq<=i;qqq++)
  for(www=0;www<=i-qqq;www++)
     if(AS_1[i1][www]<AS_1[i1][www+1])
     {
      eee=AS_1[i1][www]; AS_1[i1][www]=AS_1[i1][www+1];
      AS_1[i1][www+1]=eee;  rrr=nnn[www];
      nnn[www]=nnn[www+1];  nnn[www+1]=rrr;
     }
 imax=nnn[0];
s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y++; else sh_n++;
    for(c1=0;c1<kol_u;c1++)
     {
      s1=Sl_S_1[c1];  cs=s1.c_str();
      s1=qr_net_test_2->Fields[8]->AsString;  cs1=s1.c_str();
      if (strcmp(cs,cs1)==0)
      {
       ak=0;
       for(v1=0;v1<kol_pr;v1++)
         ak=ak+Sl_N_1[i1][v1]*Sl_N_U_1[c1][v1];
       ak=exp((ak-1)/pow(fff_o,2));
       for(x1=0;x1<=i;x1++)
       {
         s1=SS_U_1[c1];  cs=s1.c_str();
         cs1=Fam_Vyb[x1].c_str();
         if (strcmp(cs,cs1)==0)    AS_1_O[i1][x1]=AS_1_O[i1][x1]+ak;
       }
      }
     }
 for(qqq=0;qqq<=i;qqq++)
  nnn[qqq]=qqq;
 for(qqq=0;qqq<=i;qqq++)
  for(www=0;www<=i-qqq;www++)
     if(AS_1_O[i1][www]<AS_1_O[i1][www+1])
     {
      eee=AS_1_O[i1][www]; AS_1_O[i1][www]=AS_1_O[i1][www+1];
      AS_1_O[i1][www+1]=eee; rrr=nnn[www];
      nnn[www]=nnn[www+1];  nnn[www+1]=rrr;
     }
 imax=nnn[0];
s1=qr_net_test_2->Fields[0]->AsString;
cs=s1.c_str();  cs1=Fam_Vyb[imax].c_str();
if (strcmp(cs,cs1)==0) sh_y_o++; else sh_n_o++;
i1++;
qr_net_test_2->Next();
}
        if (sh_y>max_y) {max_y=sh_y; fun=fff;}
        if (sh_y_o>max_y_o) {max_y_o=sh_y_o; fun_o=fff;}
     }
     if (all_v==true)
        {tb_vyborkaFun->AsFloat=fun; tb_vyborkaFun_O->AsFloat=fun_o;
all=false; all_v=false;}
     else { all=true; goto ff22;}
}
     tb_vyborkaPrav->AsInteger=sh_y;  tb_vyborkaOchib->AsInteger=sh_n;
     tb_vyborkaProc_Prav->AsFloat=100*sh_y/(sh_y+sh_n);
     tb_vyborkaProc_Ochib->AsFloat=100*sh_n/(sh_y+sh_n);
     tb_vyborkaPrav_O->AsInteger=sh_y_o;
     tb_vyborkaOchib_O->AsInteger=sh_n_o;
     tb_vyborkaProc_Prav_O->AsFloat=100*sh_y_o/(sh_y_o+sh_n_o);
     tb_vyborkaProc_Ochib_O->AsFloat=100*sh_n_o/(sh_y_o+sh_n_o);
     tb_vyborka->Post();
     hhh++;  sh_y=0;  sh_n=0;  sh_y_o=0;   sh_n_o=0;  KOMB[i]++;
    }
    a22: if (b>0)
    {
     if(KOMB[b-1]<(kol_pol-2-(i-b)))
     {
      KOMB[b-1]++;
      for(z=b;z<=i;z++)
       KOMB[z]=KOMB[z-1]+1;
      if(b<i) b++;
      goto a11;
     }
     else
      {
       b--;
       goto a22;
      }
     }
  }
i1=0;
tb_vyborka->Close();
S="select Kol_Fam, avg(Proc_Prav), avg(Proc_Ochib), avg(Proc_Prav_O),
avg(Proc_Ochib_O) from 'DATA\\vyborka.db' GROUP BY Kol_Fam";
        qr_net_test->SQL->Clear();  qr_net_test->SQL->Add(S);
        qr_net_test->Open();
QRLabel20->Caption=" из " + IntToStr(kol_pol);
tb_vyborka->Open();
}
}
//---------------------------------------------------------------------------
 void __fastcall Tfm_net_test::QRSubDetail2BeforePrint(
     TQRCustomBand *Sender, bool &PrintBand)
{
if (fm_main->test_fun==1) {QRDBRichText1->
Height=14*tb_vyborkaKol_Fam->AsInteger; QRLabel27->Caption="";}
   else if (fm_main->test_fun==2) QRDBRichText1->
Height=14*tb_vyborkaKol_Fam->AsInteger+22;
QRSubDetail2->Height=QRDBRichText1->Height+10;
QRDBText3->Height=QRDBRichText1->Height+9;
QRDBText6->Height=QRDBRichText1->Height+9;
QRDBText11->Height=QRDBRichText1->Height+9;
QRDBText12->Height=QRDBRichText1->Height+9;
}
//---------------------------------------------------------------------------
```

Файл new_nab.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "new_nab.h"
#include "nastr.h"
#include "main.h"
#include "error.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_new_nab *fm_new_nab;
bool s;
//---------------------------------------------------------------------------
__fastcall Tfm_new_nab::Tfm_new_nab(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::FormClose(TObject *Sender,
     TCloseAction &Action)
{
Action=caFree;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::FormCreate(TObject *Sender)
{
ed_New_Slovo->Clear(); tb_nastr->Open();
tb_nastr->Last(); ed_Nab_New->Text=tb_nastrN_Nab->Value+1;
ed_Nab_New->ReadOnly=true; lb_All->Visible=false;
bb_Save_Nab->Enabled=false; s=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::bb_AddClick(TObject *Sender)
{
if (ed_New_Slovo->Text!="")
{
   fm_main->k++;  lst_bx_Nab->Items->Add(ed_New_Slovo->Text);
   ed_New_Slovo->Clear();
}
if (fm_main->k==10)
{
   lb_All->Visible=true;  bb_Add->Enabled=false;
   ed_New_Slovo->Visible=false;  lb_N_S->Visible=false;
```

```
   bb_Save_Nab->Enabled=true;
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::bb_Save_NabClick(TObject *Sender)
{
int i;
tb_nastr->Open(); tb_nastr->Insert();
tb_nastrN_Nab->AsInteger=StrToInt(ed_Nab_New->Text);
for (i=0;i<10;i++)
   tb_nastr->Fields[i+1]->AsString=lst_bx_Nab->Items->Strings[i];
tb_nastrV_Nab->AsBoolean=false; tb_nastrK_Povt->AsInteger=1;
tb_nastrK_Slov->AsInteger=10; tb_nastr->Post();
bb_Save_Nab->Enabled=false; bb_Cancel->Enabled=false; s=true;
Close();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::FormCloseQuery(TObject *Sender,
     bool &CanClose)
{
if (s==false)
{
   CanClose=false;
   fm_main->er="Необхідно зберегти набір.";  Beep(400,500);
   Tfm_error *fm_error=new Tfm_error(Application);
}
if (fm_main->k<10)
{
   CanClose=false;
   fm_main->er="Необхідно ввести всі 10 слів в новий набор.";
   Beep(400,500);
   Tfm_error *fm_error=new Tfm_error(Application);
}
fm_main->z=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_new_nab::bb_CancelClick(TObject *Sender)
{
fm_main->k=10; s=true;
Close();
}
//---------------------------------------------------------------------------
```

Файл print.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <math.h>
#pragma hdrstop
#include "print.h"
#include "analiz.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_print *fm_print;
int k,kol;
//---------------------------------------------------------------------------
__fastcall Tfm_print::Tfm_print(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_print::qrp_printBeforePrint(TQuickRep *Sender,
     bool &PrintReport)
{
bool n=false;
AnsiString SS_1,SS_2;
k=0;
kol=0;
tb_print->Open(); tb_print->First();
while(!tb_print->Eof)
{
tb_print->Edit(); tb_printN->AsBoolean=n;
if (n==false) n=true; else if (n==true) n=false;
if (tb_printN->AsBoolean==true) kol++;
tb_print->Next();
}
tb_print->Close(); tb_print->Open();
SS_2="select * from 'DATA\\main_print_i.db' where Print=true And N=true";
        qr_print_2->SQL->Clear();  qr_print_2->SQL->Add(SS_2);
        qr_print_2->Open();
SS_1="select * from 'DATA\\main_print_i.db' where Print=true And
N=false";
        qr_print_1->SQL->Clear();  qr_print_1->SQL->Add(SS_1);
        qr_print_1->Open();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_print::QRSubDetail1AfterPrint(TQRCustomBand
*Sender,
     bool BandPrinted)
{
k++;
if (k<kol) qr_print_2->Next(); else qr_print_2->Close();
}
//---------------------------------------------------------------------------
void __fastcall Tfm_print::QRDBText3Print(TObject *sender,
     AnsiString &Value)
{}
//---------------------------------------------------------------------------
```

Файл prizn.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <math.h>
#pragma hdrstop
#include "prizn.h"
#include "main.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_prizn *fm_prizn;
AnsiString S1[1000];
int i,r,t,kol_z;
//---------------------------------------------------------------------------
__fastcall Tfm_prizn::Tfm_prizn(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_prizn::FormClose(TObject *Sender, TCloseAction
&Action)
{
Action=caFree;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_prizn::FormCreate(TObject *Sender)
{
 rgr_och->ItemIndex=0;  strgr_prizn->Visible=false;
 clb_Fam->Clear();  tb_main->Open();  tb_main->First();
 kol_z=0;  i=0;
 while (!tb_main->Eof)
 {
 r=0;
 if (tb_mainFam->AsString!="")
 {
  for (t=0;t<i && r!=1;t++)
 if (S1[t]==tb_mainFam->AsString) r=1;
 if (r!=1){S1[i]=tb_mainFam->AsString;
          clb_Fam->Items->Add(tb_mainFam->AsString);   i++;}
 }
 kol_z++;  tb_main->Next();
 }
 clb_Fam->Sorted=true;  bb_no_all->Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_prizn::bb_allClick(TObject *Sender)
{
int t;
   for (t=0;t<clb_Fam->Items->Count;t++)
      clb_Fam->Checked[t]=true;
   bb_no_all->Enabled=true;     bb_all->Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_prizn::bb_no_allClick(TObject *Sender)
{
int t;
   for (t=0;t<clb_Fam->Items->Count;t++)
      clb_Fam->Checked[t]=false;
   bb_no_all->Enabled=false;     bb_all->Enabled=true;
}
//---------------------------------------------------------------------------
```

```
void __fastcall Tfm_prizn::clb_FamClickCheck(TObject *Sender)
{
int t,k=0;
for (t=0;t<clb_Fam->Items->Count;t++)
      if (clb_Fam->Checked[t]==true) k++;
if (k==kol_z)  {bb_all->Enabled=false; bb_no_all->Enabled=true;}
else if (k==0)  {bb_all->Enabled=true; bb_no_all->Enabled=false;}
else {bb_all->Enabled=true; bb_no_all->Enabled=true;}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_prizn::bb_priznClick(TObject *Sender)
{AnsiString S_1,Slovo[100],Fam_All[1000];
int t,i=0,j,s=0,k,sl,ns=1,max=0,kol,dl,g,p;
double M[1000][20],S[1000][20],R;
bb_prizn->Enabled=false; bb_all->Enabled=false;
bb_no_all->Enabled=false; clb_Fam->Enabled=false;
if (rgr_och->ItemIndex==0) fm_main->tt="DATA\\main.db";
  else if (rgr_och->ItemIndex==1) fm_main->tt="DATA\\main_boch.db";
 for (t=0;t<clb_Fam->Items->Count;t++)
      if (clb_Fam->Checked[t]==true) {Fam_All[i]=clb_Fam->Items->
Strings[t]; i++;}
fm_main->kol_fam=i;
for(j=fm_main->kol_fam-1;j>=0;j--)
{
S_1="select DISTINCT Slovo from '"+fm_main->tt+"' where Fam=:F";
        qr_prizn->SQL->Clear();   qr_prizn->SQL->Add(S_1);
        qr_prizn->ParamByName("F")->AsString=Fam_All[j];
        qr_prizn->Open();
if(j==fm_main->kol_fam-1)
 while(!qr_prizn->Eof)
 {
  Slovo[s]=qr_prizn->Fields[0]->AsString; s++;  qr_prizn->Next();
 }
else
 for(k=0;k<s;k++)
  {
  qr_prizn->First();
  sl=0;
  while(!qr_prizn->Eof && sl==0)
   {
    if(Slovo[k]==qr_prizn->Fields[0]->AsString) sl++;
    qr_prizn->Next();
   }
   if(sl==0)  Slovo[k]="";
  }
}
for(k=0;k<s;k++)
{
 if (Slovo[k]!="")
 {
    strgr_prizn->Cells[0][ns++]=Slovo[k];
    if (StrLen(Slovo[k].c_str())>max) max=StrLen(Slovo[k].c_str());
 }
}
strgr_prizn->Cells[0][0]="Ñëîâî";
for(k=1;k<=max;k++)
    strgr_prizn->Cells[k][0]=IntToStr(k)+" ñèìâîë";
ns=1;
for(k=0;k<s;k++)
if (Slovo[k]!="")
{
for(j=fm_main->kol_fam-1;j>=0;j--)
{
for(dl=0;dl<20;dl++)
{M[j][dl]=0; S[j][dl]=0;}
kol=0;
S_1="select * from '"+fm_main->tt+"' where Fam=:F and Slovo=:S";
        qr_prizn->SQL->Clear();  qr_prizn->SQL->Add(S_1);
        qr_prizn->ParamByName("F")->AsString=Fam_All[j];
        qr_prizn->ParamByName("S")->AsString=Slovo[k];
        qr_prizn->Open();
while(!qr_prizn->Eof)
{
for(dl=0;dl<StrLen(Slovo[k].c_str());dl++)
   M[j][dl]=M[j][dl]+qr_prizn->Fields[10+dl]->AsInteger;
kol++; qr_prizn->Next();
}
for(dl=0;dl<StrLen(Slovo[k].c_str());dl++)
   M[j][dl]=M[j][dl]/kol;
qr_prizn->First(); kol=0;
while(!qr_prizn->Eof)
{
for(dl=0;dl<StrLen(Slovo[k].c_str());dl++)
   S[j][dl]=S[j][dl]+pow((qr_prizn->Fields[10+dl]->AsInteger-M[j][dl]),2);
kol++; qr_prizn->Next();
}
for(dl=0;dl<StrLen(Slovo[k].c_str());dl++)
   S[j][dl]=sqrt(S[j][dl]/(kol-1));
}
for(dl=0;dl<StrLen(Slovo[k].c_str());dl++)
{
R=0; p=0;
for(j=0;j<(fm_main->kol_fam-1);j++)
for(g=j;g<fm_main->kol_fam;g++)
{
 R=R+(S[j][dl]+S[g][dl])/fabs(M[j][dl]+M[g][dl]); p++;
}
R=R/p; strgr_prizn->Cells[dl+1][ns]=R;
}
ns++;
}
bb_prizn->Enabled=true; bb_all->Enabled=true; bb_no_all->Enabled=true;
clb_Fam->Enabled=true; strgr_prizn->Visible=true;
}
//---------------------------------------------------------------------------
```

Файл read.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "read.h"
#include "main.h"
#include "net.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_read *fm_read;
//---------------------------------------------------------------------------
__fastcall Tfm_read::Tfm_read(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
```

Файл scor.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "scor.h"
#include "main.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
Tfm_scor *fm_scor;
//---------------------------------------------------------------------------
__fastcall Tfm_scor::Tfm_scor(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_scor::FormClose(TObject *Sender, TCloseAction
&Action)
{
Action=caFree;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_scor::FormCreate(TObject *Sender)
{
int i=0,j,sl;
float k,kol,sum,ochib,sr_vr;
char *f;
AnsiString S,Fam[1000],fm;
S="delete from 'DATA\\scor.db'";
       qr_scor->SQL->Clear();  qr_scor->SQL->Add(S);
       qr_scor->ExecSQL();
S="select DISTINCT Fam from 'DATA\\main.db'";
        qr_scor->SQL->Clear(); qr_scor->SQL->Add(S);
        qr_scor->Open();
while(!qr_scor->Eof)
 {
```

```
   Fam[i]=qr_scor->Fields[0]->AsString;  i++;  qr_scor->Next();
 }
for(j=i-1;j>=0;j--)
{
   S="select * from 'DATA\\main.db' where Fam=:F";
        qr_scor->SQL->Clear(); qr_scor->SQL->Add(S);
        qr_scor->ParamByName("F")->AsString=Fam[j];
        qr_scor->Open();
   sum=0;  kol=0; ochib=0;
   while (!qr_scor->Eof)
   {
    fm=qr_scor->Fields[8]->AsString;
    f=fm.c_str();  sl=StrLen(f);
    for(k=10;k<(sl+10);k++)
          {sum=sum+qr_scor->Fields[k]->AsInteger; kol++;}
    ochib=ochib+qr_scor->Fields[30]->AsInteger;
    qr_scor->Next();
   }
tb_scor->Open(); tb_scor->Insert();
tb_scorFam->AsString=Fam[j]; tb_scorSumm->AsInteger=sum;
tb_scorKol->AsInteger=kol; sr_vr=sum/kol;
tb_scorSr_vr->AsFloat=sr_vr;
tb_scorScor->AsFloat=60/(0.001*sum/kol);
tb_scorKol_ochib->AsInteger=ochib;
tb_scorOchib_Proc->AsFloat=100*ochib/(kol+ochib); sum=0;
qr_scor->First();
while (!qr_scor->Eof)
   {
    fm=qr_scor->Fields[8]->AsString;  f=fm.c_str();
    sl=StrLen(f); for(k=10;k<(sl+10);k++)
      sum=sum+100*abs(sr_vr-qr_scor->Fields[k]->AsInteger)/sr_vr;
    qr_scor->Next();
   }
tb_scorNeravnom->AsFloat=sum/kol; tb_scor->Post();
}
tb_scor->Close(); tb_scor->Open();
}
//---------------------------------------------------------------------------
```

Файл tabl.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "tabl.h"
#include "main.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "ToolEdit"
#pragma link "ToolEdit"
#pragma link "ToolEdit"
#pragma link "ToolEdit"
#pragma resource "*.dfm"
Tfm_tabl *fm_tabl;
bool vi=false,vp=false;
//---------------------------------------------------------------------------
__fastcall Tfm_tabl::Tfm_tabl(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_tabl::FormClose(TObject *Sender, TCloseAction
&Action)
{
Action=caFree;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_tabl::FormCreate(TObject *Sender)
{
f_ed_tabl_i->Text=""; f_ed_tabl_p->Text=""; bb_Tabl->Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_tabl::bb_TablClick(TObject *Sender)
{
   int i,p;
   bool pp;
   AnsiString S;
   lb_inf->Caption="Йде процес переписування.";
    bb_Tabl->Enabled=false;  f_ed_tabl_i->Enabled=false;
   f_ed_tabl_p->Enabled=false;
   S="select * from '"+f_ed_tabl_i->Text+"'";
   qr_tabl_i->SQL->Clear();  qr_tabl_i->SQL->Add(S);
   qr_tabl_i->Open();
   tb_tabl_p->TableName=f_ed_tabl_p->Text;  tb_tabl_p->Active=true;
   tb_tabl_p->Open();
   while(!qr_tabl_i->Eof)
   {
    tb_tabl_p->First();   tb_tabl_p->Insert();
     for (i=0;i<31;i++)
       tb_tabl_p->Fields[i]->AsString=qr_tabl_i->Fields[i]->AsString;
    qr_tabl_i->Next();
   }
   tb_tabl_p->Post();  tb_tabl_p->Close();  Beep(400,500);
   vi=false;   vp=false;
   lb_inf->Caption="";
   bb_Tabl->Enabled=true; f_ed_tabl_i->Enabled=true;
   f_ed_tabl_p->Enabled=true;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_tabl::f_ed_tabl_iChange(TObject *Sender)
{
if (f_ed_tabl_i->Text!="") vi=true; else vi=false;
if (vp==true && vi==true) bb_Tabl->Enabled=true; else bb_Tabl->
Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_tabl::f_ed_tabl_pChange(TObject *Sender)
{
if (f_ed_tabl_p->Text!="") vp=true; else vp=false;
if (vi==true && vp==true) bb_Tabl->Enabled=true; else bb_Tabl->
Enabled=false;
}
//---------------------------------------------------------------------------
```

Файл test.cpp:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <string.h>
#include <stdlib.h>
#include <dos.h>
#pragma hdrstop
#include "test.h"
#include "main.h"
#include "error.h"
#include "analiz.h"
#include "nastr.h"
#include "new_nab.h"
#include "print.h"
#include "tabl.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "TimerLst"
#pragma resource "*.dfm"
#define N 10
Tfm_test *fm_test;
int str_kol,i,j=0,sv,kp,nkp,ks,nks,nn,m,och;
TDateTime T1,T2,T3,T;
unsigned short hour, minut, sec, msec;
char *c;
bool e=false;
AnsiString Str_old[N],Str_n,Str[N], Str1="Введіть слово - ";
//---------------------------------------------------------------------------
__fastcall Tfm_test::Tfm_test(TComponent* Owner)
   : TForm(Owner)
{}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::FormClose(TObject *Sender, TCloseAction
&Action)
{Action=caFree; }
//---------------------------------------------------------------------------
void __fastcall Tfm_test::bb_testClick(TObject *Sender)
{
int z;
if (ed_Fam->Text== "")
        {
         fm_main->er="Ви забули ввести своє прізвище!!! Введіть.";
         Beep(400,500);
         Tfm_error *fm_error=new Tfm_error(Application);
         goto A1;
        }
```

```
if (ed_Imya->Text== "")
        {
         fm_main->er="Ви забули ввести своє ім'я!!! Введіть.";
         Beep(400,500);
         Tfm_error *fm_error=new Tfm_error(Application);
         goto A1;
        }
if (ed_Otch->Text== "")
        {
         fm_main->er="Ви забули ввести своє по-батькові!!! Введіть.";
         Beep(400,500);
         Tfm_error *fm_error=new Tfm_error(Application);
         goto A1;
        }
if (de_Test->Text== "  .  .  ")
        {
         fm_main->er="Ви забули ввести дату!!! Введіть.";
         Beep(400,500);
         Tfm_error *fm_error=new Tfm_error(Application);
         goto A1;
        }
rx_tmr_lst->Active=true; bb_test->Enabled=false;
fm_main->N1->Enabled=false; fm_main->N2->Enabled=false;
fm_main-> N3->Enabled=false; lb_test->Visible=true; ed_test->Visible=true;
for (z=0;z<N;z++)
   Str_old[z]="";
j=0; i=0; e=false; nkp=0; nks=0;
ed_test->SetFocus();
A1:
}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::FormActivate(TObject *Sender)
{
AnsiString S;
int i;
WindowState=wsMaximized;
S="select N_Nab, K_Povt, K_Slov, Slovo_1, Slovo_2, Slovo_3, Slovo_4,
Slovo_5, Slovo_6, Slovo_7, Slovo_8, Slovo_9, Slovo_10 from
'DATA\\nabor.db' where V_Nab=:V";
qr_nastr->SQL->Clear(); qr_nastr->SQL->Add(S);
qr_nastr->ParamByName("V")->AsBoolean=true;
qr_nastr->Open();
nn=qr_nastr->Fields[0]->Value; kp=qr_nastr->Fields[1]->Value;
ks=qr_nastr->Fields[2]->Value;
m=kp*ks;
for (i=0;i<N;i++)
   Str[i]=qr_nastr->Fields[3+i]->Value;
}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::FormCreate(TObject *Sender)
{ lb_test->Visible=false; lb_och->Visible=false; ed_test->Visible=false;
Randomize(); }
//---------------------------------------------------------------------------
void __fastcall Tfm_test::ed_testKeyPress(TObject *Sender, char &Key)
{
tb_main->Open(); tb_main->Edit();
if (Key!=c[i])
   {Key=0;i=0;lb_och->Visible=true; ed_test->Text="";
lb_och->Caption="Була допущена помилка. Повторіть ввод слова.";
T1=Time(); sv=0; Beep(400,500);och++;}
         else
                {
                 lb_och->Visible=false;  rx_tmr_lst->Active=false;
                 rx_tmr_lst->Active=true;  lb_och->Caption="";
                 T2=Time();  T3=T2-T1;  T1=T2;
                 DecodeTime(T3,hour, minut, sec, msec);
                 tb_main->Fields[10+i]->AsInteger=
msec+sec*1000+minut*60000+hour*3600000;
                 i++;
                 sv=sv+msec+sec*1000+minut*60000+hour*3600000;
                }
if (i==str_kol)
        {
         tb_mainAll_Slovo->AsInteger=sv;  tb_mainOch->AsInteger=och;
         e=false; nkp++;
         de_Test->SetFocus(); ed_test->SetFocus();
        }
}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::ed_testEnter(TObject *Sender)
{
int k;
if (e==false)
{
 if(j==m)
   {
    bb_test->Enabled=true;
    fm_main->N1->Enabled=true; fm_main->N2->Enabled=true; fm_main->
N3->Enabled=true;
    lb_test->Caption="Тест окончен.";  rx_tmr_lst->Active=false;
    tb_main->Open(); tb_main->Post();
    ed_test->Visible=false; i=0;
    ed_Fam->Clear(); ed_Imya->Clear(); ed_Otch->Clear(); de_Test->Clear();
    lb_N_K_P->Caption="";
    goto E;
   }
 S:
if (nkp<kp && nkp>0) goto P;
if (nks==ks && nkp==kp)
      {
       bb_test->Enabled=true;
       fm_main->N1->Enabled=true; fm_main->N2->Enabled=true;
fm_main->N3->Enabled=true;
       lb_test->Caption="Тест окончен.";  rx_tmr_lst->Active=false;
       tb_main->Open();  tb_main->Post();  ed_test->Visible=false;  i=0;
    ed_Fam->Clear(); ed_Imya->Clear(); ed_Otch->Clear(); de_Test->Clear();
       lb_N_K_P->Caption="";
       goto E;
      }
 Str_n=Str[random(N)];
 for(k=0;k<j;k++)
  if (Str_n==Str_old[k]) goto S;
 Str_old[j]=Str_n;  nks++;  nkp=0;
 lb_test->Caption=Str1+Str_n;  c=Str_n.c_str();  str_kol=strlen(c);  j++;
 P:
 e=true;  i=0;  sv=0;  T1=Time();
 tb_main->Open();  tb_main->Insert();
 tb_mainFam->AsString=ed_Fam->Text.c_str();
 tb_mainImya->AsString=ed_Imya->Text.c_str();
 tb_mainOtch->AsString=ed_Otch->Text.c_str();
 tb_mainData->Value=de_Test->Date;
 tb_mainTime->Value=Time();  tb_mainSlovo->AsString=Str_n;
 tb_mainN_Nab->AsInteger=nn;  tb_mainK_Povt->AsInteger=kp;
 tb_mainN_Povt->AsInteger=nkp+1;  tb_main->Post();
 lb_N_K_P->Caption="Кількість повторень -"+IntToStr(kp) + " Номер
повторення - " + StrToInt(nkp+1);
 och=0;
 E:
}
}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::ed_testChange(TObject *Sender)
{
if (i==0) ed_test->Text="";
}
//---------------------------------------------------------------------------
void __fastcall Tfm_test::FormCloseQuery(TObject *Sender, bool
&CanClose)
{ fm_main->N1->Enabled=true; fm_main->N2->Enabled=true; fm_main->
N3->Enabled=true; }
//---------------------------------------------------------------------------
void __fastcall Tfm_test::RxTimerEvent1Timer(TObject *Sender)
{
i=0;
lb_och->Visible=true; ed_test->Text="";
lb_och->Caption="Пройшло занадто багато часу. Повторіть ввод слова.";
T1=Time(); sv=0; Beep(400,500);
}
//---------------------------------------------------------------------------
```

Додаток В. Система автентифікації користувачів інформаційних систем за рукописним почерком «АРП»

Система «АРП» розроблена на мові Delphi з використанням, для обробки та зберігання інформації, програми Database Desktop (програма для роботи з базами даних) та SQL-запитів [200]. Програма працює під керуванням ОС Windows. Як засіб формування та динамічного передавання характеристик РП користувача в комп'ютер використовується графічний планшет, як один з варіантів пристрою з сенсорним екраном. У зв'язку з цим в Delphi був додатково встановлено компонент JanHWinTab (сторінка Jan Hlavenka) – компонент для роботи с ГП (компонент використовує інтерфейс Wintab).

Початкові коди програми розміщені в наступних основних файлах:

- файл формування проекта: Project1.dpr;
- 7 файлів форм: Unit1.dfm, Unit2.dfm, Unit3.dfm, Unit4.dfm, Unit5.dfm, Unit6.dfm, Unit7.dfm;
- 7 файлів реалізації модулів: Unit1.pas, Unit2.pas, Unit3.pas, Unit4.pas, Unit5.pas, Unit6.pas, Unit7.pas.

Для зберігання кожного навчального зразка РП для кожного користувача *використовуються таблиці, які знаходяться в окремих файлах відповідних папок*. В процесі роботи системи «АКП», для зберігання характеристик написання кожного символа, *використовуються окремі таблиці*, взалежності від кількості зафіксованих в них КТ.

Призначення файлу формування проекта основних файлів реалізацій модулів:

Project1.dpr – використовується для зберігання інформації о формах та модулях системи. В ньому знаходяться оператори ініціалізації та запуску программи на виконання.

Unit2.pas – містить код реалізації модуля головної (батьківської) форми проекта. З цієї форми викликаються всі інші (дочірні) форми. Крім того, при виборі відповідного пункта меню, виконується архівування (резервне копіювання) зразків РП користувачів, які були накопичені в БДНЗ за поточний день.

Unit1.pas – містить КРДМ, в якому виконується введення з ГП КФ (слово-пароль) та супутньої інформації (прізвище, ім'я, по-батькові). Слово-пароль, який було введено, відображається, а потім обробляється.

Unit3.pas – містить КРДМ, в якому виконується перегляд накопичених в базі даних НЗ. Також виконється моделювання всіх етапів первинної обробки зразків.

Unit4.pas – містить КРДМ, в якому виконується архівування (резервне копіювання) зразків РП користувачів, які були накопичені в БДНЗ.

Unit5.pas – містить КРДМ, в якому виконується перегляд наступних статистичних характеристик зразків РП, накопичених в БДНЗ: розподіл значень кількості точок в зображенні слова; розподіл значень кількості КТ в зображенні слова; розподіл значень кількості КТ в зображенні символа; розподіл значень кількості зображень символів (до об'єднання). Цю інформіцію можна побачить і в табличному вигляді, і в вигляді графіків. Також виконується перегляд кількості накопичених НЗ РП кожного користувача і визначення того, скільки з них придатні

для використання їх в якості навчальних даних.

Unit6.pas – містить КРДМ, в якому виконується видалення поорожніх зразків (створених під час якихось збоїв) з БДНЗ.

Unit7.pas – містить КРДМ, в якому виконується вибір параметрів розпізнавання написаного слова-пароля; реальне розділення зображення КФ на зображення символів (запис в різні файли); розпізнавання написаного слова-пароля. Можливе розпізнавання одного зразка або необхідної кількості зразків, тим самим тестуючи роботу системи.

Лістинг файлу формування проекта та основних файлів реалізацій модулів:

Файл Project1.dpr:

```
program Project1;
uses
  Forms,
  Unit1 in 'Unit1.pas' {fm_test},
  Unit2 in 'Unit2.pas' {fm_main},
  Unit3 in 'Unit3.pas' {fm_read},
  Unit4 in 'Unit4.pas' {fm_setup},
  Unit5 in 'Unit5.pas' {fm_tochki},
  Unit6 in 'Unit6.pas' {fm_obrab},
  Unit7 in 'Unit7.pas' {fm_raspozn};
{$R *.res}
begin
  Application.Initialize;
  Application.CreateForm(Tfm_main, fm_main);
  Application.Run;
end.
```

Файл Unit2.pas:

```
unit Unit2;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, Menus, StdCtrls;
type
  Tfm_main = class(TForm)
    MainMenu1: TMainMenu;  ///Меню системи
    test: TMenuItem;
    analiz: TMenuItem;
    an_reed: TMenuItem;
    arch: TMenuItem;
    ar_arch: TMenuItem;
    ar_setup: TMenuItem;
    an_tochki: TMenuItem;
    obrab: TMenuItem;
    an_raspozn: TMenuItem;
    exit: TMenuItem;
    procedure testClick(Sender: TObject);
    procedure an_reedClick(Sender: TObject);
    procedure ar_archClick(Sender: TObject);
    procedure ar_setupClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure an_tochkiClick(Sender: TObject);
    procedure obrabClick(Sender: TObject);
    procedure an_raspoznClick(Sender: TObject);
    procedure exitClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_main: Tfm_main;
  S:string;
implementation
uses Unit1, Unit3, Unit4, Unit5, Unit6, Unit7;
{$R *.dfm}
procedure Tfm_main.testClick(Sender: TObject);
begin
/////  Створення чергової дочірньої форми
Application.CreateForm(Tfm_test, fm_test);
end;
procedure Tfm_main.an_reedClick(Sender: TObject);
begin
Application.CreateForm(Tfm_read, fm_read);
end;
procedure Tfm_main.ar_archClick(Sender: TObject);
var i,iff:integer;
StartInfo:TStartupInfo;
ProcInfo:TProcessInformation;
p,path,poisk:AnsiString;
f1,f2:TextFile;
SR: TSearchRec;
D1,D2:TDateTime;
Label mm1,mm2,mm3,mm4,mm5,mm6,mm7,mm8,mm9;
begin
///////  Архівування НЗ, які були накопичені за поточній день {
ChDir(S+'\');
fm_main.arch.Enabled:=false;
poisk:=FileSearch('D:\Program Files\WinRAR_27\winrar.exe','d:\');
p:=DateTimeToStr(Now); AssignFile(f1, S+'\path.txt'); reset(f1);
read(f1,path); CloseFile(f1);
for i:=1 to Length(p) do begin
if ((Copy(p,i,1)=':') or (Copy(p,i,1)=' ')) then
begin
p:=Copy(p,1,(i-1))+'_'+Copy(p,(i+1),(Length(p)-i))
end;
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path+'\WinRAR.exe a -tn1d -sfx
Dannie\data_bmp_' +p+ ' data_bmp\*.*')),
nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm1: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm1;
CloseHandle(ProcInfo.hProcess);
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path+'\WinRAR.exe a -tn1d -sfx
Dannie\data_txt_' +p+ '
data_txt\*.*')),nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm2: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm2;
CloseHandle(ProcInfo.hProcess);
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path + '\WinRAR.exe a -tn1d -
xbaza.* -xmain.* -sfx Dannie\data_db_'+p+ '
data_db\*.*')),nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm3: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm3;
CloseHandle(ProcInfo.hProcess);
```

```
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path+'\WinRAR.exe a -tn1d
Dannie\data_bmp_' +p+ '.rar data_bmp\*.*')),
nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm4: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm4;
CloseHandle(ProcInfo.hProcess);
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path+'\WinRAR.exe a -tn1d
Dannie\data_txt_' +p+ '.rar
data_txt\*.*')),nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm5: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm5;
CloseHandle(ProcInfo.hProcess);
end;
FillChar(StartInfo,Sizeof(StartInfo),#0);
StartInfo.cb:=Sizeof(StartInfo);
if not CreateProcess(nil,StrLower(PChar(path+'\WinRAR.exe a -tn1d -
xbaza.* -xmain.* Dannie\data_db_'+p+ '.rar
data_db\*.*')),nil,nil,false,CREATE_NEW_CONSOLE or
HIGH_PRIORITY_CLASS,nil,nil,StartInfo,ProcInfo)
then ShowMessage(IntToStr(GetLastError))
else begin
mm6: if WaitForSingleObject(ProcInfo.hProcess,20000)=WAIT_TIMEOUT
then goto mm6;
CloseHandle(ProcInfo.hProcess);
end;
///// } Архівування Н3, які були накопичені за поточній день
//////////////////// Видалення заархівованих Н3 {
iff:=FindFirst('data_db\*.db',faAnyFile, SR);
while iff=0 do
 begin
 D1:=FileDateToDateTime(SR.Time);  D2:=Date();
 if ((D1>D2) and (SR.Name<>'MAIN.DB') and (SR.Name<>'BAZA.DB'))
then
 begin
 DeleteFile('data_db\'+SR.Name);
 end;
 iff := FindNext(SR);
 end;
FindClose(SR); iff:=FindFirst('data_txt\*.log',faAnyFile, SR);
while iff=0 do
 begin
 D1:=FileDateToDateTime(SR.Time);  D2:=Date();
 if (D1>D2) then
 begin
 DeleteFile('data_txt\'+SR.Name);
 end;
 iff := FindNext(SR);
 end;
FindClose(SR);
iff:=FindFirst(S+'\data_bmp\*.bmp',faAnyFile, SR);
while iff=0 do
 begin
 D1:=FileDateToDateTime(SR.Time);  D2:=Date();
 if (D1>D2) then
 begin
 DeleteFile('data_bmp\'+SR.Name);
 end;
 iff := FindNext(SR);
 end;
FindClose(SR);
ShowMessage('Архівування закінчено!!!');
fm_main.arch.Enabled:=true;
////////////////////// } Видалення заархівованих Н3
end;
procedure Tfm_main.ar_setupClick(Sender: TObject);
begin
Application.CreateForm(Tfm_setup, fm_setup);
end;
procedure Tfm_main.FormCreate(Sender: TObject);
begin
GetDir(0, S);
end;
procedure Tfm_main.an_tochkiClick(Sender: TObject);
begin
Application.CreateForm(Tfm_tochki, fm_tochki);
end;
procedure Tfm_main.obrabClick(Sender: TObject);
begin
Application.CreateForm(Tfm_obrab, fm_obrab);
end;
procedure Tfm_main.an_raspoznClick(Sender: TObject);
begin
Application.CreateForm(Tfm_raspozn, fm_raspozn);
end;
procedure Tfm_main.exitClick(Sender: TObject);
begin
Close();
end;
end.
```

Файл Unit1.pas:

```
unit Unit1;
interface
uses
  Windows,DateUtils, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms, Dialogs, StdCtrls, JH_WinTab,JH_WinTab_Const, ExtCtrls,
Buttons, FileCtrl, ExtDlgs, DB, DBTables, DBCtrls, TeeProcs, TeEngine,
Chart, DbChart, Grids, Outline, DirOutln, ComCtrls;
type
  Tfm_test = class(TForm)
    JanHWinTab: TJanHWinTab;
    ed_xy: TEdit;  ed_press: TEdit;  DriveComboBox1: TDriveComboBox;
    DirectoryListBox1: TDirectoryListBox;  bt_er: TButton;  bt_start: TButton;
    ed_f: TEdit; ed_fam: TEdit;  ed_imya: TEdit; ed_otch: TEdit;
    Label1: TLabel;  Label2: TLabel;  Label3: TLabel;
    ds_tb_main: TDataSource;  tb_main: TTable;  tb_mainID: TIntegerField;
    tb_mainFam: TStringField;  tb_mainImya: TStringField;
    tb_mainOtch: TStringField;  tb_mainContext: TIntegerField;
    tb_mainStatus: TIntegerField;  tb_mainCursor: TIntegerField;
    tb_mainButtons: TIntegerField;  tb_mainPosition_X: TIntegerField;
    tb_mainPosition_Y: TIntegerField; tb_mainPressure: TIntegerField;
    tb_mainTangPress: TIntegerField;
    tb_mainOrientation_orAzimuth: TIntegerField;
    tb_mainOrientation_orAltitude: TIntegerField;
    tb_mainOrientation_orTwist: TIntegerField;
    tb_mainRotation_roPitch: TIntegerField;
    tb_mainRotation_roRoll: TIntegerField;
    tb_mainRotation_roYaw: TIntegerField; tb_mainData: TDateField;
    tb_mainTime: TTimeField; tb_mainTime_ms: TIntegerField;
    bt_finish: TButton; bt_save: TButton;  bt_del: TButton;  Image1: TImage;
    stb_inf: TStatusBar;  tb_baza: TTable;  ds_tb_baza: TDataSource;
    tb_bazaFam: TStringField;  tb_bazaImya: TStringField;
    tb_bazaOtch: TStringField;  tb_bazaData: TDateField;
    tb_bazaKolich: TIntegerField;
    procedure Timer1Timer(Sender: TObject);
    procedure JanHWinTabPacket(Sender: TObject; PacketSerial, Handle:
HWND; Packet: TTabletPacket);
    procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X, Y:
Integer);
    procedure FormCreate(Sender: TObject);
    procedure bt_erClick(Sender: TObject);
    procedure ed_famChange(Sender: TObject);
    procedure ed_imyaChange(Sender: TObject);
    procedure ed_otchChange(Sender: TObject);
    procedure bt_finishKeyDown(Sender: TObject; var Key: Word; Shift:
TShiftState);
    procedure bt_saveClick(Sender: TObject);
    procedure bt_delClick(Sender: TObject);
    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
    procedure DriveComboBox1Change(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure bt_startClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
  end;
var
  fm_test: Tfm_test;Press_:integer; Log:System.text;Color_:TColor;
  Brush_:TBrushStyle;
  nom, nom_zap, p :integer;
  ris,reg,press:boolean;
  path_b,path_2:string;
  type dannie=record
ID:Integer; Fam:String;  Imya:String; Otch:String; Context:Integer;
Status:Integer; Cursor:Integer; Buttons:Integer; Position_X:Integer;
Position_Y:Integer; Pressure:Integer; TangPress:Integer;
Orientation_orAzimuth:Integer; Orientation_orAltitude:Integer;
Orientation_orTwist:Integer; Rotation_roPitch:Integer;
Rotation_roRoll:Integer; Rotation_roYaw:Integer;
Data:TDateTime; Time:TDateTime; Time_ms:Integer;
end;
//////  Структура, в котрій зберігаються дані про точки
var Pac: array [0..32000] of dannie; implementation
uses Unit2, Unit3;
{$R *.dfm}
procedure Tfm_test.Timer1Timer(Sender: TObject);
begin
ed_xy.text:=JanHWinTab.GetInfoAsString(WTI_DDCTXS,
CTX_OUTORGX,0);
end;
procedure Tfm_test.JanHWinTabPacket(Sender: TObject; PacketSerial,
Handle: HWND; Packet: TTabletPacket);
///// Функція, викликається під час передавання пакета даних з ГП  на
комп’ютер
var diam,d_ch:integer;
A:TPoint;
begin
GetCursorPos(A);
if(Image1.Visible=true)     then
begin
  ed_press.Text:='Pressure= '+inttostr(Packet.pkPressure);
  Press_:=Packet.pkPressure;
if (Packet.pkPressure<>0) or (nom=1) then
  begin
Pac[nom_zap].ID:=nom_zap; Pac[nom_zap].Fam:=ed_fam.Text;
Pac[nom_zap].Imya:=ed_imya.Text; Pac[nom_zap].Otch:=ed_otch.Text;
Pac[nom_zap].Context:=Packet.pkContext;
Pac[nom_zap].Status:=Packet.pkStatus;
Pac[nom_zap].Cursor:=Packet.pkCursor;
Pac[nom_zap].Buttons:=Packet.pkButtons;
Pac[nom_zap].Position_X:=Packet.pkPosition.X;
Pac[nom_zap].Position_Y:=Packet.pkPosition.Y;
Pac[nom_zap].Pressure:=Packet.pkPressure;
Pac[nom_zap].TangPress:=Packet.pkTangPress;
Pac[nom_zap].Orientation_orAzimuth:=Packet.pkOrientation.orAzimuth;
Pac[nom_zap].Orientation_orAltitude:=Packet.pkOrientation.orAltitude;
Pac[nom_zap].Orientation_orTwist:=Packet.pkOrientation.orTwist;
Pac[nom_zap].Rotation_roPitch:=Packet.pkRotation.roPitch;
Pac[nom_zap].Rotation_roRoll:=Packet.pkRotation.roRoll;
Pac[nom_zap].Rotation_roYaw:=Packet.pkRotation.roYaw;
Pac[nom_zap].Data:=Date(); Pac[nom_zap].Time:=Time();
Pac[nom_zap].Time_ms:=MilliSecondOfTheSecond(Now);
if ((Pac[nom_zap].Pressure<>0) and (Pac[nom_zap-1].Pressure<>0) and
(nom_zap<>0)) then press:=true else press:=false;
ed_xy.Text:='X= '+inttostr(Packet.pkPosition.X)+'
Y='+inttostr(Packet.pkPosition.Y)+ '
XE='+inttostr(round(Pac[nom_zap].Position_X*Image1.Width/8000)) + '
YE=' +inttostr(Image1.Height-
round(Pac[nom_zap].Position_Y*Image1.Height/6000));
if (reg=false) then
begin
A:=fm_test.Image1.ClientToScreen(Point(round(Pac[nom_zap].Position_X*I
mage1.ClientWidth/8000),Image1.ClientHeight-
round(Pac[nom_zap].Position_Y*Image1.ClientHeight/6000)));
if ((Pac[nom_zap].Pressure<>0) and (Pac[nom_zap-1].Pressure<>0) and
(nom_zap<>0)) then
Image1.Canvas.LineTo(A.X,A.Y)
else
Image1.Canvas.MoveTo(A.X,A.Y);
end;
if (Pac[nom_zap].Pressure<>0) then ris:=true;
nom_zap:=nom_zap+1;
end;
if (Packet.pkPressure=0) and (nom=1) then nom:=0 else if
(Packet.pkPressure<>0) then nom:=1;
end;
end;
procedure Tfm_test.Image1MouseMove(Sender: TObject; Shift: TShiftState;
X,  Y: Integer);
///////   Функція, яка викликається під час руху мишки по області,
///////   на  котрій відображається зображення, що створюється
var diam,d_ch:integer;
  begin
if Press_=0 then p:=0 else p:=1;
if Press_>0 then begin
diam:=100; d_ch:=round((1023-Press_)/10); diam:=diam-d_ch;
end;
if(reg=true) then
begin
if (press=true) then
Image1.Canvas.LineTo(X,Y)
else
Image1.Canvas.MoveTo(X,Y);
end;
end;
procedure Tfm_test.FormCreate(Sender: TObject);
///////    Функція, що викликається під час створення даної форми
var path_:string; i,kol_povt:integer;
povt:boolean;
begin
ed_xy.Visible:=false; ed_press.Visible:=false; ed_f.Visible:=false;
press:=false; ris:=false; reg:=true; stb_inf.Panels[1].Width:=110;
stb_inf.Panels[1].Text:='Напишить "ізація".';
stb_inf.Panels[2].Text:='Планшет.'; nom:=1;
Image1.Visible:=false; Image1.Top:=0; stb_inf.Visible:=false;
bt_start.Enabled:=false; bt_save.Visible:=false; bt_del.Visible:=false;
bt_er.Visible:=false; bt_finish.Width:=0; bt_er.Visible:=false;
  Image1.Canvas.Brush.Color:=clWhite;
  Image1.Canvas.Rectangle(0,0,(Image1.Width),(Image1.Height));
end;
procedure Tfm_test.bt_erClick(Sender: TObject);
///////    Очистка області зображення, без зберігання даних
var i:integer;
begin
Image1.Canvas.pen.Color:=clWhite; Image1.Canvas.Brush.Color:=clWhite;
Image1.Canvas.Brush.Style:=bsSolid;
Image1.Canvas.Rectangle(0,0,(Image1.width),(Image1.height));
ris:=false;
for i:=0 to (nom_zap-1) do
begin
Pac[i].ID:=0; Pac[i].Fam:=''; Pac[i].Imya:=''; Pac[i].Otch:='';
Pac[i].Context:=0; Pac[i].Status:=0; Pac[i].Cursor:=0; Pac[i].Buttons:=0;
Pac[i].Position_X:=0; Pac[i].Position_Y:=0; Pac[i].Pressure:=0;
Pac[i].TangPress:=0; Pac[i].Orientation_orAzimuth:=0;
Pac[i].Orientation_orAltitude:=0; Pac[i].Orientation_orTwist:=0;
Pac[i].Rotation_roPitch:=0; Pac[i].Rotation_roRoll:=0;
Pac[i].Rotation_roYaw:=0; Pac[i].Data:=0; Pac[i].Time:=0;
Pac[i].Time_ms:=0;
end;
end;
//////////  Перевірка чи можна починати ввод слова-пароля (чи введено
ім'я, //////  по-батькове та прізвище)  {
procedure Tfm_test.ed_famChange(Sender: TObject);
begin
if (ed_fam.Text<>'') and (ed_imya.Text<>'') and (ed_otch.Text<>'') then
bt_start.Enabled:=true else bt_start.Enabled:=false;
end;
procedure Tfm_test.ed_imyaChange(Sender: TObject);
begin
if (ed_fam.Text<>'') and (ed_imya.Text<>'') and (ed_otch.Text<>'') then
bt_start.Enabled:=true else bt_start.Enabled:=false;
end;
procedure Tfm_test.ed_otchChange(Sender: TObject);
begin
if (ed_fam.Text<>'') and (ed_imya.Text<>'') and (ed_otch.Text<>'') then
bt_start.Enabled:=true else bt_start.Enabled:=false;
end;
////////  Перевірка чи можна починати ввод слова-пароля (чи введено ім'я,
//////  по-батькове та прізвище)
procedure Tfm_test.bt_finishKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
//////////////////   Функція, яка викликається під час якої-небудь клавіши
/////// клавиатури під час написання слова-пароля на ГП
```

```
  label e;
begin
if((Key=ord('M')) and (ssCtrl in Shift)) then
     begin stb_inf.Panels[2].Text:='Мишка.'; reg:=true; end;
if((Key=ord('P')) and (ssCtrl in Shift)) then
     begin stb_inf.Panels[2].Text:='Планшет.'; reg:=false; end;
if((Key=ord('S')) and (ssCtrl in Shift)) then
     begin if(ed_xy.Visible=true) then ed_xy.Visible:=false else
ed_xy.Visible:=true; if(ed_press.Visible=true) then ed_press.Visible:=false
else ed_press.Visible:=true; if(ed_f.Visible=true) then ed_f.Visible:=false else
ed_f.Visible:=true; end;
if(Key=VK_ESCAPE) then
     begin    bt_er.Click; Image1.Canvas.pen.Color:=clBlack;
ed_xy.Visible:=false; ed_press.Visible:=false; ed_f.Visible:=false; end;
if(Key=VK_BACK) then
     begin    bt_del.Click; Image1.Canvas.pen.Color:=clBlack;
Image1.Visible:=false; stb_inf.Visible:=false; ed_xy.Visible:=false;
ed_press.Visible:=false; ed_f.Visible:=false; end;
if(Key=VK_SPACE) then begin
stb_inf.Visible:=false; ed_xy.Visible:=false; ed_press.Visible:=false;
ed_f.Visible:=false;
if(ris=true) then
begin
Image1.Visible:=false; bt_save.Visible:=true; bt_del.Visible:=true;
ed_xy.Text:='X=       Y=   '; ed_press.Text:='Pressure=';
ed_f.Text:='LogFile='; bt_save.Click; bt_start.Click;
goto e;
end;
if (ris=false) then
begin
ed_fam.Visible:=true; ed_imya.Visible:=true; ed_otch.Visible:=true;
DriveComboBox1.Visible:=true; DirectoryListBox1.Visible:=true;
bt_start.Visible:=true; Label1.Visible:=true; Label2.Visible:=true;
Label3.Visible:=true; bt_save.Visible:=false; bt_del.Visible:=false;
Image1.Canvas.pen.Color:=clWhite; Image1.Canvas.Brush.Color:=clWhite;
Image1.Canvas.Brush.Style:=bsSolid;
Image1.Canvas.Rectangle(0,0,(Image1.width),(Image1.height));
Image1.Canvas.pen.Color:=clBlack; Image1.Visible:=false;
end;
e:;
end;
end;
procedure Tfm_test.bt_saveClick(Sender: TObject);
///////   Функція зберігання в базі даних зразка, що отримано
var i,kol_povt:integer;
povt:boolean;
begin
with tb_main do
 begin
TableName := path_2; CreateTable;
end;
Rewrite(Log); Append(Log);
Image1.Picture.SaveToFile(path_b);
tb_main.Open(); tb_main.Insert();
ris:=false;
for i:=0 to (nom_zap-1) do
begin
Writeln(Log,'Packet.pkContext='+inttostr(Pac[i].Context)+
'Packet.pkStatus='+inttostr(Pac[i].Status)+
'Packet.pkCursor='+inttostr(Pac[i].Cursor)+
'Packet.pkButtons='+inttostr(Pac[i].Buttons)+
'Packet.pkPosition.X='+inttostr(Pac[i].Position_X)+
'Packet.pkPosition.Y='+inttostr(Pac[i].Position_Y)+
'Packet.pkPressure='+inttostr(Pac[i].Pressure)+
'Packet.pkTangPress='+inttostr(Pac[i].TangPress)+
'Packet.pkOrientation.orAzimuth='+inttostr(Pac[i].Orientation_orAzimuth)+
'Packet.pkOrientation.orAltitude='+inttostr(Pac[i].Orientation_orAltitude)+
'Packet.pkOrientation.orTwist='+inttostr(Pac[i].Orientation_orTwist)+
'Packet.pkRotation.roPitch='+inttostr(Pac[i].Rotation_roPitch)+
'Packet.pkRotation.roRoll='+inttostr(Pac[i].Rotation_roRoll)+
'Packet.pkRotation.roYaw='+
inttostr(Pac[i].Rotation_roYaw)+'  Date='+DateTimeToStr(Pac[i].Data)+
'Time='+DateTimeToStr(Pac[i].Time)+
'Milliseconds='+IntToStr(Pac[i].Time_ms));
tb_main.Append();
tb_mainID.AsInteger:=Pac[i].ID; tb_mainFam.AsString:=Pac[i].Fam;
tb_mainImya.AsString:=Pac[i].Imya; tb_mainOtch.AsString:=Pac[i].Otch;
tb_mainContext.AsInteger:=Pac[i].Context;
tb_mainStatus.AsInteger:=Pac[i].Status;
tb_mainCursor.AsInteger:=Pac[i].Cursor;
tb_mainButtons.AsInteger:=Pac[i].Buttons;
tb_mainPosition_X.AsInteger:=Pac[i].Position_X;
tb_mainPosition_Y.AsInteger:=Pac[i].Position_Y;
tb_mainPressure.AsInteger:=Pac[i].Pressure;
tb_mainTangPress.AsInteger:=Pac[i].TangPress;
tb_mainOrientation_orAzimuth.AsInteger:=Pac[i].Orientation_orAzimuth;
tb_mainOrientation_orAltitude.AsInteger:=Pac[i].Orientation_orAltitude;
tb_mainOrientation_orTwist.AsInteger:=Pac[i].Orientation_orTwist;
tb_mainRotation_roPitch.AsInteger:=Pac[i].Rotation_roPitch;
tb_mainRotation_roRoll.AsInteger:=Pac[i].Rotation_roRoll;
tb_mainRotation_roYaw.AsInteger:=Pac[i].Rotation_roYaw;
tb_mainData.Value:=Pac[i].Data; tb_mainTime.Value:=Pac[i].Time;
tb_mainTime_ms.Value:=Pac[i].Time_ms;
tb_main.Next();
end;
CloseFile(Log);
tb_main.Close();
ed_f.Text:='LogFile='; ed_fam.Visible:=true; ed_imya.Visible:=true;
ed_otch.Visible:=true; ed_xy.Visible:=true; ed_press.Visible:=true;
ed_f.Visible:=true; DriveComboBox1.Visible:=true;
DirectoryListBox1.Visible:=true; bt_start.Visible:=true;
Label1.Visible:=true; Label2.Visible:=true; Label3.Visible:=true;
bt_save.Visible:=false; bt_del.Visible:=false; povt:=false;
tb_baza.Open();
while tb_baza.Eof<>true do
begin
if((tb_bazaFam.AsString=ed_fam.Text) and
(tb_bazaImya.AsString=ed_imya.Text) and
(tb_bazaOtch.AsString=ed_otch.Text) and (tb_bazaData.Value=Date())) then
begin  tb_baza.Edit(); tb_bazaKolich.AsInteger:=tb_bazaKolich.AsInteger+1;
povt:=true; kol_povt:=tb_bazaKolich.AsInteger; end;
tb_baza.Next();
end;
if(povt<>true) then
begin
tb_baza.Insert();
tb_bazaFam.AsString:=ed_fam.Text; tb_bazaImya.AsString:=ed_imya.Text;
tb_bazaOtch.AsString:=ed_otch.Text; tb_bazaData.Value:=Date();
tb_bazaKolich.AsInteger:=1; kol_povt:=tb_bazaKolich.AsInteger;
tb_baza.Next();
end;
tb_baza.Close(); stb_inf.Panels[0].Width:=210;
stb_inf.Panels[0].Text:='Сьогодні ви написали '+IntToStr(kol_povt)+
'разів(и).';
bt_er.Click;
end;
procedure Tfm_test.bt_delClick(Sender: TObject);
var i:integer;
begin
ris:=false;
for i:=0 to (nom_zap-1) do
begin
Pac[i].ID:=0; Pac[i].Fam:=''; Pac[i].Imya:=''; Pac[i].Otch:='';
Pac[i].Context:=0; Pac[i].Status:=0; Pac[i].Cursor:=0; Pac[i].Buttons:=0;
Pac[i].Position_X:=0; Pac[i].Position_Y:=0; Pac[i].Pressure:=0;
Pac[i].TangPress:=0; Pac[i].Orientation_orAzimuth:=0;
Pac[i].Orientation_orAltitude:=0; Pac[i].Orientation_orTwist:=0;
Pac[i].Rotation_roPitch:=0; Pac[i].Rotation_roRoll:=0;
Pac[i].Rotation_roYaw:=0; Pac[i].Data:=0;
Pac[i].Time:=0; Pac[i].Time_ms:=0;
end;
ed_fam.Visible:=true; ed_imya.Visible:=true; ed_otch.Visible:=true;
ed_xy.Visible:=true; ed_press.Visible:=true; ed_f.Visible:=true;
DriveComboBox1.Visible:=true; DirectoryListBox1.Visible:=true;
bt_start.Visible:=true; Label1.Visible:=true; Label2.Visible:=true;
Label3.Visible:=true; bt_save.Visible:=false; bt_del.Visible:=false;
bt_er.Click;
end;
procedure Tfm_test.FormCloseQuery(Sender: TObject; var CanClose:
Boolean);
begin
if (ris=true) then begin CanClose:=false;
ShowMessage('Інформація, що введена, не збережена та не видалена. Для
виконання однієї з цих дій нажміть пробіл, а потім одну з
запропонованих кнопок.');   end else CanClose:=true;
end;
procedure Tfm_test.DriveComboBox1Change(Sender: TObject);
begin
```

```
DirectoryListBox1.Drive:=DriveComboBox1.Drive;
end;
procedure Tfm_test.FormClose(Sender: TObject; var Action: TCloseAction);
begin
Action:=caFree;
end;
procedure Tfm_test.bt_startClick(Sender: TObject);
//// Функція, що викликається, під час натискання на кнопку формы
/////“Почати    ввод” (написання на ГП слова пароля)
var path_:string; i,kol_povt:integer;   str:AnsiString; povt:boolean;
begin
stb_inf.Visible:=true;
stb_inf.Panels[2].Text:='';
stb_inf.Panels[2].Text:='Для очистки вікна натиснить "Esc". Для
зберігання даних-"Пробіл". Для повернення в головне вікно без
збереження даних-"BackSpace".';
povt:=false;
tb_baza.Open();
while tb_baza.Eof<>true do
begin
if((tb_bazaFam.AsString=ed_fam.Text) and
(tb_bazaImya.AsString=ed_imya.Text) and
(tb_bazaOtch.AsString=ed_otch.Text) and (tb_bazaData.Value=Date())) then
begin povt:=true; kol_povt:=tb_bazaKolich.AsInteger; end;
tb_baza.Next();
end;
if(povt<>true) then
begin
kol_povt:=0;
end;
tb_baza.Close();
stb_inf.Panels[0].Width:=210;
stb_inf.Panels[0].Text:='Сьогодні вы написали '+IntToStr(kol_povt)+'
раз(а).';
Image1.Canvas.pen.Color:=clBlack;
Image1.Visible:=true; Label1.Visible:=false; Label2.Visible:=false;
Label3.Visible:=false; ed_fam.Visible:=false; ed_imya.Visible:=false;
ed_otch.Visible:=false; ed_xy.Enabled:=false; ed_press.Enabled:=false;
ed_f.Enabled:=false; ed_xy.Visible:=false; ed_press.Visible:=false;
ed_f.Visible:=false; DriveComboBox1.Visible:=false;
DirectoryListBox1.Visible:=false;
bt_start.Visible:=false; bt_er.Visible:=false; p:=0;
path_:=DirectoryListBox1.Directory+'\'+'Data_txt\'+ed_fam.Text+'_'+
DateTimeToStr(Now)+'.log';
for i:=3 to Length(path_) do begin
if Copy(path_,i,1)=':' then
begin
path_:=Copy(path_,1,(i-1))+'_'+Copy(path_,(i+1),(Length(path_)-i))
end;
end;
path_b:=DirectoryListBox1.Directory+'\'+'Data_bmp\'+ed_fam.Text+'_'+
DateTimeToStr(Now)+'.bmp';
for i:=3 to Length(path_b) do begin
if Copy(path_b,i,1)=':' then
begin
path_b:=Copy(path_b,1,(i-1))+'_'+Copy(path_b,(i+1),(Length(path_b)-i))
end;
end;
path_2:=DirectoryListBox1.Directory+'\'+'Data_db\'+ed_fam.Text+'_'+
DateTimeToStr(Now);
for i:=3 to Length(path_2) do begin
if Copy(path_2,i,1)=':' then
begin
path_2:=Copy(path_2,1,(i-1))+'_'+Copy(path_2,(i+1),(Length(path_2)-i))
end;
end;
  AssignFile(Log,path_);
ed_f.Text:='LogFile='+path_;
nom_zap:=0;
bt_finish.SetFocus;
end;
end.
```

Файл Unit3.pas:

```
unit Unit3;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, FileCtrl, StdCtrls, ExtCtrls, DB, DBTables,math;
type
  Tfm_read = class(TForm)
    Image1: TImage;    DriveComboBox2: TDriveComboBox;
    FileListBox1: TFileListBox;  bt_ok: TButton; bt_finish: TButton;
    DirectoryListBox2: TDirectoryListBox;
    tb_main: TTable;  tb_mainID: TIntegerField;  tb_mainFam: TStringField;
    tb_mainImya: TStringField;  tb_mainOtch: TStringField;
    tb_mainContext: TIntegerField;   tb_mainStatus: TIntegerField;
    tb_mainCursor: TIntegerField;  tb_mainButtons: TIntegerField;
    tb_mainPosition_X: TIntegerField;  tb_mainPosition_Y: TIntegerField;
    tb_mainPressure: TIntegerField;  tb_mainTangPress: TIntegerField;
    tb_mainOrientation_orAzimuth: TIntegerField;
    tb_mainOrientation_orAltitude: TIntegerField;
    tb_mainOrientation_orTwist: TIntegerField;
    tb_mainRotation_roPitch: TIntegerField;
    tb_mainRotation_roRoll: TIntegerField;
    tb_mainRotation_roYaw: TIntegerField; tb_mainData: TDateField;
    tb_mainTime: TTimeField; tb_mainTime_ms: TIntegerField;
    ds_tb_main: TDataSource; chb_k: TCheckBox; chb_r: TCheckBox;
    chb_p: TCheckBox;  chb_s: TCheckBox; chb_t: TCheckBox;
    ed_t: TEdit;  chb_x: TCheckBox; ed_kol_t: TEdit; ed_x: TEdit;
    chb_povt: TCheckBox; chb_bl: TCheckBox; ed_bl: TEdit;
    chb_ramka: TCheckBox; chb_m: TCheckBox;  ed_m: TEdit;
    chb_pov: TCheckBox;  ed_pov: TEdit; chb_sdv_x: TCheckBox;
    ed_sdv_x: TEdit; chb_sdv_y: TCheckBox; ed_sdv_y: TEdit;
    chb_ramka_1: TCheckBox; chb_pov_1: TCheckBox;
    ed_pov_1: TEdit; chb_sdv_y_1: TCheckBox;  chb_simv_m: TCheckBox;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DriveComboBox2Change(Sender: TObject);
    procedure DirectoryListBox2Change(Sender: TObject);
    procedure FileListBox1Change(Sender: TObject);
    procedure FileListBox1Click(Sender: TObject);
    procedure bt_okClick(Sender: TObject);
    procedure bt_finishKeyDown(Sender: TObject; var Key: Word; Shift:
TShiftState);
    procedure FormCreate(Sender: TObject);
    procedure ed_xKeyDown(Sender: TObject; var Key: Word; Shift:
TShiftState);
    procedure ed_kol_tChange(Sender: TObject);
    procedure ed_xChange(Sender: TObject);
    procedure ed_kol_tKeyPress(Sender: TObject; var Key: Char);
    procedure ed_xKeyPress(Sender: TObject; var Key: Char);
    procedure ed_sdv_xKeyPress(Sender: TObject; var Key: Char);
    procedure ed_povKeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_read: Tfm_read;
type dannie=record
ID:Integer; Fam:String; Imya:String; Otch:String; Context:Integer;
Status:Integer; Cursor:Integer; Buttons:Integer; Position_X:Integer;
Position_Y:Integer; Pressure:Integer; TangPress:Integer;
Orientation_orAzimuth:Integer; Orientation_orAltitude:Integer;
Orientation_orTwist:Integer; Rotation_roPitch:Integer;
Rotation_roRoll:Integer; Rotation_roYaw:Integer; Data:TDateTime;
Time:TDateTime; Time_ms:Integer; N_simv:Integer;
end;
type kontr=record
ID:Integer; Fam:String; Imya:String; Otch:String; Context:Integer;
Status:Integer; Cursor:Integer; Buttons:Integer; Position_X:Integer;
Position_Y:Integer; Pressure:Integer; TangPress:Integer;
Orientation_orAzimuth:Integer; Orientation_orAltitude:Integer;
Orientation_orTwist:Integer; Rotation_roPitch:Integer;
Rotation_roRoll:Integer; Rotation_roYaw:Integer; Data:TDateTime;
Time:TDateTime; Time_ms:Integer; type_t:integer; N_simv:Integer;
end;
var Pac1: array [0..32000] of dannie;
var Kontr_T: array [0..32000] of kontr;
var KKontr_T: array [0..32000] of integer;
nom_zap:integer;
implementation
uses Unit1, Unit2, Unit4;
{$R *.dfm}
procedure Tfm_read.FormClose(Sender: TObject; var Action: TCloseAction);
begin
ChDir(S+'\');
```

```
Action:=caFree;
end;
procedure Tfm_read.DriveComboBox2Change(Sender: TObject);
begin
DirectoryListBox2.Drive:=DriveComboBox2.Drive;
end;
procedure Tfm_read.DirectoryListBox2Change(Sender: TObject);
begin
FileListBox1.Directory:=DirectoryListBox2.Directory;
end;
procedure Tfm_read.FileListBox1Change(Sender: TObject);
begin
if (FileListBox1.FileName<>'') then bt_ok.Enabled:=true else
bt_ok.Enabled:=false;
end;
procedure Tfm_read.FileListBox1Click(Sender: TObject);
begin
if (FileListBox1.FileName<>'') then bt_ok.Enabled:=true else
bt_ok.Enabled:=false;
end;
procedure Tfm_read.bt_okClick(Sender: TObject);
var
pr,x,y,dx1,dx2,dy1,dy2,dx,dy,kt,i,p,ni,nii,j,j1,kz,w,nol,u,tt,ut,ut1,ut2,utt,ut12,
uttk, nvt, utt1:integer;
min_x_r, max_x_r, min_y_r, max_y_r, simv, min_rr, nmin_rr, jj, XS, YS, nn,
MM, XL, YL, XR, YR, ij, ww, ED_X11, ED_Y11, net, ij1, nnk, nin, jni, jni1,
jni3, nnkt, jn:integer;
min_x_r1: array [0..100] of integer;
max_x_r1: array [0..100] of integer;
min_y_r1: array [0..100] of integer;
max_y_r1: array [0..100] of integer;
trr: array [0..100] of integer;
kx: array [0..9] of integer;
ky: array [0..9] of integer;
kp: array [0..9] of integer;
ss,ss1,ss2,ss3,ss4:Extended;
sss, ekt:boolean;
x11,x12,x21,x22,x0,y11,y12,y21,y22,y0,a1,a2,b1,b2,minx1,minx2,maxx1,ma
xx2,miny1,miny2,maxy1,maxy2,tt1,per,per1,per11,per12,per21,per22,sssss,
PR2, PR1, ED_M11:real;
Label xv, xv1, rrr, rrr_1, rrr_2, rrr_3, utttt, jjj, enen, enen1, slt, jni2, jni4, nt,
nt1, ee, mu12, uttt5;
begin
Image1.Canvas.pen.Color:=clWhite; Image1.Canvas.Brush.Color:=clWhite;
Image1.Canvas.Brush.Style:=bsSolid;
Image1.Canvas.Rectangle(0,0,(Image1.width),(Image1.height));
Image1.Canvas.pen.Color:=clBlack; FileListBox1.Visible:=false;
DriveComboBox2.Visible:=false; DirectoryListBox2.Visible:=false;
bt_ok.Visible:=false; Image1.Visible:=true;
pr:=0; x:=0; y:=0; dx1:=0; dx2:=0; dy1:=0; dy2:=0; kt:=0; i:=0; nom_zap:=0;
p:=0; kz:=0;
tb_main.TableName:=FileListBox1.FileName; tb_main.Open();
while tb_main.Eof<>true do
begin
////////////////// Створення масиву точок, якщо виконувати згладжування {
if (chb_s.Checked=true)
then
begin
if (tb_mainPressure.AsInteger=0) then
 begin
 if (kz>9) then
  begin
  for j1:=kz-5 to kz-2 do
  begin
  Pac1[i].ID:=i;  Pac1[i].Position_X:=kx[j1];  Pac1[i].Position_Y:=ky[j1];
  Pac1[i].Pressure:=kp[j1];  i:=i+1;  nom_zap:=nom_zap+1;
  end;
  kz:=0;
  end
 else if(kz>0) then
  begin
  for j1:=0 to kz-1 do
  begin
  Pac1[i].ID:=i; Pac1[i].Position_X:=kx[j1];  Pac1[i].Position_Y:=ky[j1];
  Pac1[i].Pressure:=kp[j1];  i:=i+1;  nom_zap:=nom_zap+1;
  end;
  kz:=0;
  end;
 Pac1[i].ID:=tb_mainID.AsInteger;
 Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
 Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
 Pac1[i].Pressure:=tb_mainPressure.AsInteger;
 i:=i+1;
 nom_zap:=nom_zap+1;
 end
else
if(kz<=4) then
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
i:=i+1;
nom_zap:=nom_zap+1;
kz:=kz+1;
end
else if ((kz>4) and (kz<9)) then
begin
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
kz:=kz+1;
end
else if (kz>=9) then
begin
kx[9]:=tb_mainPosition_X.AsInteger;
ky[9]:=tb_mainPosition_Y.AsInteger;
kp[9]:=tb_mainPressure.AsInteger;
Pac1[i].Position_X:=0; Pac1[i].Position_Y:=0; Pac1[i].Pressure:=0;
for j1:=0 to 9 do
 begin
 Pac1[i].Position_X:=Pac1[i].Position_X+kx[j1];
 Pac1[i].Position_Y:=Pac1[i].Position_Y+ky[j1];
 Pac1[i].Pressure:=Pac1[i].Pressure+kp[j1];
 if (j1<>0) then
  begin
  kx[j1-1]:=kx[j1];  ky[j1-1]:=ky[j1];  kp[j1-1]:=kp[j1];
  end;
 end;
Pac1[i].ID:=i;
Pac1[i].Position_X:=round(Pac1[i].Position_X/10);
Pac1[i].Position_Y:=round(Pac1[i].Position_Y/10);
Pac1[i].Pressure:=round(Pac1[i].Pressure/10);
i:=i+1;
nom_zap:=nom_zap+1; if (kz=9) then kz:=kz+1;
end;
end
////////////////// }  Створення масиву точок, якщо виконувати згладжування
////////////////// Створення масиву точок, якщо не виконувати згладжування
{
else
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
i:=i+1;
nom_zap:=nom_zap+1;
end;
/////////////// }  Створення масиву точок, якщо не виконувати
згладжування
tb_main.Next();
end;
tb_main.Close();
////////////////// Видалення повторів {  ////////////////////////////
if(chb_povt.Checked=true) then
begin
 for j:=0 to i-1 do
 begin
  if ((Pac1[j].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].Position_X=Pac1[j+1].Position_X) and
(Pac1[j].Position_Y=Pac1[j+1].Position_Y) and (j<(i-1))) then
  begin
   u:=j+2;
   while ((Pac1[u].Pressure<>0) and
```

```
(Pac1[j].Position_X=Pac1[u].Position_X) and
(Pac1[j].Position_Y=Pac1[u].Position_Y) and (u<=(i-1))) do
     begin
       u:=u+1;
     end;
     nol:=u-1;
     begin
       i:=i-nol+j;
       nom_zap:=nom_zap-nol+j;
       for u:=j+1 to i-1 do
       begin
         Pac1[u]:=Pac1[u+nol-j];
       end;
     end;
   end;
  end;
end;
////////////////  }  Видалення повторів  ////////////////////////////
////////////////  Видалення випадкових точок {  ////////////////////////////
if(chb_t.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (((Pac1[j].Pressure=0) and (j<(i-1))) or (j=0)) then
    begin
      if(Pac1[j].Pressure=0) then jn:=j else jn:=j-1;
      slt:nol:=jn;
      for u:=jn+1 to (jn+1+StrToInt(ed_t.Text)) do
        if (u=i) then begin nol:=u-1; break; end else if (Pac1[u].Pressure=0) then
nol:=u;
      if(nol<>jn) then
      begin
        i:=i-nol+jn;
        nom_zap:=nom_zap-nol+jn;
        for u:=jn+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-jn];
        end;
        if (jn<(i-1)) then goto slt;
      end;
    end;
  end;
end;
////////////////  }  Видалення випадкових точок  ////////////////////////////
////////////////  Видалення хвостів за кількістю точок  {
tt:=StrToInt(ed_kol_t.Text);
if ((chb_x.Checked=true) and (StrToInt(ed_kol_t.Text)<>0)) then
begin
xv:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;
      for u:=j+1 to (j+1+tt) do
        if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
         else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then nol:=u;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
////////////////  }  Видалення хвостів за кількістю точок  ///////////////
////////////////  Видалення хвостів за дліною  {  ////////////////////////////
tt1:=StrToFloat(ed_x.Text);
if ((chb_x.Checked=true) and (StrToFloat(ed_x.Text)<>0)) then
begin
xv1:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;
      u:=j+1;
      PR2:=0;
      while (PR2<=tt1) do
      begin
        if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
         else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then begin nol:=u;  end;
        u:=u+1;
        PR2:=PR2+sqrt(sqr(Pac1[u].Position_X-
Pac1[u-1].Position_X)+sqr(Pac1[u].Position_Y-Pac1[u-1].Position_Y));
      end;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
////////////////  }  Видалення хвостів за дліною  ////////////////////////////
////////////////  Визначення меж букв  {  ////////////////////////////
begin
  simv:=0;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r1[simv]:=Pac1[j].Position_X;
      max_x_r1[simv]:=Pac1[j].Position_X;
      min_y_r1[simv]:=Pac1[j].Position_Y;
      max_y_r1[simv]:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
      if (Pac1[j].Pressure<>0) then
      begin
        if (Pac1[j].Position_X<min_x_r1[simv]) then
min_x_r1[simv]:=Pac1[j].Position_X;
        if (Pac1[j].Position_X>max_x_r1[simv]) then
max_x_r1[simv]:=Pac1[j].Position_X;
        if (Pac1[j].Position_Y<min_y_r1[simv]) then
min_y_r1[simv]:=Pac1[j].Position_Y;
        if (Pac1[j].Position_Y>max_y_r1[simv]) then
max_y_r1[simv]:=Pac1[j].Position_Y;
      end
      else
      begin
        trr[simv]:=Pac1[j].ID;
        if((j<>0) and (j<>(i-1))) then
        begin
          simv:=simv+1;
          min_x_r1[simv]:=Pac1[j+1].Position_X;
          max_x_r1[simv]:=Pac1[j+1].Position_X;
          min_y_r1[simv]:=Pac1[j+1].Position_Y;
          max_y_r1[simv]:=Pac1[j+1].Position_Y;
        end;
      end;
  end;
rrr_3: if (simv>5) then
  begin
    min_rr:=min_x_r1[1]-max_x_r1[0];
    nmin_rr:=1;
    for j:=1 to simv do
    begin
      if ((min_x_r1[j]-max_x_r1[j-1])<min_rr) then begin min_rr:=min_x_r1[j]-
```

```
max_x_r1[j-1]; nmin_rr:=j; end;
    end;
    max_x_r1[nmin_rr-1]:=Max(max_x_r1[nmin_rr-1],max_x_r1[nmin_rr]);
    min_x_r1[nmin_rr-1]:=Min(min_x_r1[nmin_rr-1],min_x_r1[nmin_rr]);
    max_y_r1[nmin_rr-1]:=Max(max_y_r1[nmin_rr-1],max_y_r1[nmin_rr]);
    min_y_r1[nmin_rr-1]:=Min(min_y_r1[nmin_rr-1],min_y_r1[nmin_rr]);
    for j:=nmin_rr to simv do
    begin
      max_x_r1[j]:=max_x_r1[j+1];  min_x_r1[j]:=min_x_r1[j+1];
      max_y_r1[j]:=max_y_r1[j+1];  min_y_r1[j]:=min_y_r1[j+1];
      trr[j-1]:=trr[j];
    end;
    simv:=simv-1;
    goto rrr_3;
  end
end;
  nn:=Pac1[0].ID;   ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    Pac1[ij].N_simv:=jj;
        end;
  nn:=trr[jj];
  end;
////////////////  }  Визначення меж букв  ////////////////////////////
////////////////  Зсув  {  ////////////////////////////
if((chb_sdv_x.Checked=true) or (chb_sdv_y.Checked=true)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  if ((chb_sdv_x.Checked=true) and (StrToInt(ed_sdv_x.Text)=777)) then
ed_sdv_x.Text:=IntToStr(round((8000-(max_x_r-min_x_r))/2)-min_x_r);
  if ((chb_sdv_y.Checked=true) and (StrToInt(ed_sdv_y.Text)=777)) then
ed_sdv_y.Text:=IntToStr(round((6000-(max_y_r-min_y_r))/2)-min_y_r);
  for j:=0 to i-1 do
  begin
    if(chb_sdv_x.Checked=true) then
Pac1[j].Position_X:=Pac1[j].Position_X+StrToInt(ed_sdv_x.Text);
    if(chb_sdv_y.Checked=true) then
Pac1[j].Position_Y:=Pac1[j].Position_Y+StrToInt(ed_sdv_y.Text);
  end;
end;
////////////////  }  Зсув  ////////////////////////////
////////////////  Масштабування  {  ////////////////////////////
if((chb_m.Checked=true) and (StrToFloat(ed_m.Text)>0)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  if (StrToFloat(ed_m.Text)=777) then
  ed_m.Text:=FloatToStr(
  Min (Min ((0-((max_x_r-min_x_r)/2+  min_x_r))/(min_x_r-((max_x_r-
min_x_r)/2+min_x_r)),(8000-((max_x_r-min_x_r)/2+min_x_r)/(max_x_r-
((max_x_r-min_x_r)/2+min_x_r)))), Min ((0-((max_y_r-
min_y_r)/2+min_y_r))/(min_y_r-((max_y_r-min_y_r)/2+min_y_r)),(6000-
((max_y_r-min_y_r)/2+min_y_r)/(max_y_r-((max_y_r-
min_y_r)/2+min_y_r))))  )  );
  for j:=0 to i-1 do
  begin
    Pac1[j].Position_X:=round((Pac1[j].Position_X-((max_x_r-
min_x_r)/2+min_x_r))*StrToFloat(ed_m.Text)+((max_x_r-
min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((Pac1[j].Position_Y-((max_y_r-
min_y_r)/2+min_y_r))*StrToFloat(ed_m.Text)+((max_y_r-
min_y_r)/2+min_y_r));
  end;
end;
////////////////  }  Масштабування  ////////////////////////////
////////////////  Поворот  {  ////////////////////////////
if(chb_pov.Checked=true) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  for j:=0 to i-1 do
  begin
    XS:=Pac1[j].Position_X;
    YS:=Pac1[j].Position_Y;
    Pac1[j].Position_X:=round((XS-((max_x_r-
min_x_r)/2+min_x_r))*cos(StrToFloat(ed_pov.Text)*3.14159265358979323
85/180.0)+(YS-((max_y_r-
min_y_r)/2+min_y_r))*sin(StrToFloat(ed_pov.Text)*PI/180.0))+
round(((max_x_r-min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((YS-((max_y_r-min_y_r)/2+ min_y_r))*
cos(StrToFloat(ed_pov.Text)*3.14159265358979323 85/180.0)-(XS-
((max_x_r-min_x_r)/2+min_x_r))*sin(StrToFloat(ed_pov.Text)*
PI/180.0))+round(((max_y_r-min_y_r)/2+min_y_r));
  end;
end;
////////////////  }  Поворот  ////////////////////////////
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
////////////////  Посимвольний поворот  {  ////////////////////////////
if(chb_pov_1.Checked=true) then
begin
  nn:=Pac1[0].ID;//0;
  ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
```

```
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      ij1:=ij;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
if (StrToFloat(ed_pov_1.Text)=777) then
ed_pov_1.Text:=FloatToStr(180.0*(arctan((YR-YL)/(XR-
XL)))/3.1415926535897932385);
j:=nn;
while (j<=trr[jj]-1) do
        begin
          ij:=ij1;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          XS:=Pac1[ij].Position_X;  YS:=Pac1[ij].Position_Y;
          Pac1[ij].Position_X:=round((XS-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*cos(StrToFloat(ed_pov_1.Text)*
3.1415926535897932385/180.0)+(YS-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*sin(StrToFloat(ed_pov_1.Text)*PI/180.0))+r
ound(((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]));
          Pac1[ij].Position_Y:=round((YS-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*cos(StrToFloat(ed_pov_1.Text)*
3.1415926535897932385/180.0)-(XS-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*sin(StrToFloat(ed_pov_1.Text)*PI/180.0))+r
ound(((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]));
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
//////////////////  }  Посимвольний поворот  ////////////////////////////
//////////////////  Посимвольний зсув по Y  {  ////////////////////////////
if(chb_sdv_y_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  MM:=max_y_r1[0];
  for jj:=0 to simv do
  begin
   MM:=Max(MM,max_y_r1[jj]);
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
     Pac1[ij].Position_Y:=Pac1[ij].Position_Y+(MM-max_y_r1[jj]);
     if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
//////////////////  }  Посимвольний зсув по Y  ////////////////////////////
//////////////////  Посимвольне масштабування  {  ////////////////////////////
if(chb_simv_m.Checked=true) then
begin
  nn:=Pac1[0].ID;
  ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      ij1:=ij;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
```

```
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  ED_X11:=round((8000-(max_x_r1[jj]-min_x_r1[jj]))/2)-min_x_r1[jj];
  ED_Y11:=round((6000-(max_y_r1[jj]-min_y_r1[jj]))/2)-min_y_r1[jj];
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=ij1;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
           Pac1[ij].Position_X:=Pac1[ij].Position_X+ED_X11;
           Pac1[ij].Position_Y:=Pac1[ij].Position_Y+ED_Y11;
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
////////////////////////////////////////////////////////////////////
  nn:=Pac1[0].ID;   ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      ij1:=ij;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  ED_M11:=
  Min (
  Min ((0-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))/(min_x_r1[jj]-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj])),(8000-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj])/(max_x_r1[jj]-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj])))), Min ((0-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))/(min_y_r1[jj]-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj])),(6000-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj])/(max_y_r1[jj]-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))))   );
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=ij1;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
           Pac1[ij].Position_X:=round((Pac1[ij].Position_X-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*ED_M11+((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]));
           Pac1[ij].Position_Y:=round((Pac1[ij].Position_Y-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*ED_M11+((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]));
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольне масштабування  ////////////////////////////
////////////////  Загальна рамка  {  ////////////////////////////
if(chb_ramka.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
Image1.Canvas.Brush.Style:=bsClear;
Image1.Canvas.Pen.Color:=clFuchsia;
Image1.Canvas.Rectangle(round(min_x_r*Image1.Width/8000),
Image1.Height-round(min_y_r*Image1.Height/6000), round(max_x_r*
Image1.Width/8000),Image1.Height-round(max_y_r*Image1.Height/6000));
end;
////////////////  }  Загальна рамка  ////////////////////////////
////////////////  Посимвольна рамка  {  ////////////////////////////
if(chb_ramka_1.Checked=true) then
begin
  Image1.Canvas.Brush.Style:=bsClear; Image1.Canvas.Pen.Color:=clLime;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
```

```
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
      end;
  end;
  Image1.Canvas.Rectangle(round(min_x_r1[jj]*Image1.Width/8000),
Image1.Height-round(min_y_r1[jj]*Image1.Height/6000),
round(max_x_r1[jj]*Image1.Width/8000),Image1.Height-
round(max_y_r1[jj]*Image1.Height/6000));
  nn:=trr[jj];
  end;
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
  Image1.Canvas.MoveTo(round(XL*Image1.Width/8000),Image1.Height-
round(YL*Image1.Height/6000));
  Image1.Canvas.LineTo(round(XR*Image1.Width/8000),Image1.Height-
round(YR*Image1.Height/6000));
end;
////////////////  }  Посимвольна рамка  ////////////////////////////
//////////  Визначення та  вивід на экран точок і КТ  {
nnk:=-1; jj:=-1;
for j:=0 to i-1 do
begin
Image1.Canvas.Pen.Color:=clBlack;
////////////////  Визначення та  вивід на экран точкі  {  ///////////////////////
if ((Pac1[j].Pressure<>0) and (pr<>0) and (Pac1[j].ID<>0)) then
Image1.Canvas.LineTo(round(Pac1[j].Position_X*Image1.Width/8000),
Image1.Height-round(Pac1[j].Position_Y*Image1.Height/6000))
else
Image1.Canvas.MoveTo(round(Pac1[j].Position_X*Image1.Width/8000),
Image1.Height-round(Pac1[j].Position_Y*Image1.Height/6000));
////////////////  }  Визначення та  вивід на экран точкі  ///////////////////////
////////  Визначення начальної або кінцевої КТ  {
ekt:=false;
if ((nnk=-1) or (Pac1[j].N_simv<>jj)) then
begin
  jj:=jj+1;
  nin:=nnk+1;
  while (nin<=(i-1)) do
        begin
         if ((nin=0) or ((Pac1[nin].N_simv=Pac1[j].N_simv) and     (Pac1[nin-
1].N_simv<Pac1[j].N_simv))) then begin nn:=nin; nnkt:=kt; end;
         if ((nin=(i-1))or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin+1].N_simv>Pac1[j].N_simv))) then begin nnk:=nin; break; end;
          nin:=nin+1;
        end;
end;
if ((chb_k.Checked=true) and (Pac1[j].Pressure<>0) and ((j=0) or (j=i-1) or
(Pac1[j-1].Pressure=0) or (Pac1[j+1].Pressure=0)))  then
begin
if (nnkt<>kt) then
begin
 net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and ((Kontr_T[net].type_t=2) or (Kontr_T[net].type_t=3)))
then
   begin
       for ni:=net to kt-2 do
      begin
       Kontr_T[ni]:=Kontr_T[ni+1];
      end;
      kt:=kt-1;
   end
   else if ((net<>kt) and (Kontr_T[net].type_t=1)) then goto nt;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=1;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt: end;
////////////////  }  Визначення начальної або кінцевої КТ
////////////////  Визначення кутової КТ  {  ////////////////////
 if ( (chb_r.Checked=true) and (ekt=false) and
 (Pac1[j].Pressure<>0) and (Pac1[j-1].Pressure<>0) and
(Pac1[j+1].Pressure<>0) and (Pac1[j].N_simv=Pac1[j-1].N_simv)  and
(Pac1[j].N_simv=Pac1[j+1].N_simv)
 and
(j<>(i-1)) and (j<>0) and ((Sign(Pac1[j].Position_X-
Pac1[j-1].Position_X)<>Sign(Pac1[j+1].Position_X-Pac1[j].Position_X)) or
 (Sign(Pac1[j].Position_Y-Pac1[j-
1].Position_Y)<>Sign(Pac1[j+1].Position_Y-Pac1[j].Position_Y))))
then
 begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and (Kontr_T[net].type_t=3)) then
   begin
       for ni:=net to kt-2 do
      begin
       Kontr_T[ni]:=Kontr_T[ni+1];
      end;
      kt:=kt-1;
   end
   else if ((net<>kt) and ((Kontr_T[net].type_t=1) or
(Kontr_T[net].type_t=2))) then goto nt1;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=2;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt1: end;
////////////////  }  Визначення кутової КТ  ////////////////////
////////////////  Визначення  КТ  перетинання  {  ////////////
if ((chb_p.Checked=true) and (Pac1[j].Pressure<>0)) then
begin
if (ekt=false) then
begin
 for ni:=nn to (j-1) do
 begin
  if ( (Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y=Pac1[ni].Position_Y) and  (Pac1[ni].Pressure<>0) )  then
   begin break;   end;
 end;
 if(ni<>j) then
 begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
   if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
 end
 else
 begin
 for ni:=nn to (nnk-1) do
 begin
      if ((Pac1[ni].Pressure<>0) and (Pac1[ni+1].Pressure<>0)  and
(Pac1[j].ID<>Pac1[ni].ID) and (Pac1[j].ID<>Pac1[ni+1].ID))  then
       begin
         if(Pac1[ni].Position_X=Pac1[ni+1].Position_X) then
         begin
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
         end
         else if(Pac1[ni].Position_Y=Pac1[ni+1].Position_Y) then
         begin
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
```

```
          if((Pac1[j].Position_Y=Pac1[ni].Position_Y) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1)) then
           begin break; end;
         end
         else
         begin
          x11:=Pac1[ni].Position_X;  x12:=Pac1[ni+1].Position_X;
          y11:=Pac1[ni].Position_Y;  y12:=Pac1[ni+1].Position_Y;
          if (x11=0) then
          begin
           b1:=y11;  a1:=(y12-b1)/x12;
          end
          else
          begin
            b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
          end;
          minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
          maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
          miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
          maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
          if((Pac1[j].Position_Y=(a1*Pac1[j].Position_X+b1)) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
          end;
       end;
  end;
  if(ni<>nnk) then
  begin
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
    if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
  end;
  end;
  end;
  for ni:=nn+1 to (j-2) do
  begin
  if ((Pac1[ni].Pressure<>0) and (Pac1[ni-1].Pressure<>0) and          (Pac1[j-
1].Pressure<>0) )  then
  begin
    x11:=Pac1[ni].Position_X; x12:=Pac1[ni-1].Position_X;
    y11:=Pac1[ni].Position_Y; y12:=Pac1[ni-1].Position_Y;
    x21:=Pac1[j].Position_X;  y21:=Pac1[j].Position_Y;
    if (j<>(nn)) then begin x22:=Pac1[j-1].Position_X; y22:=
Pac1[j-1].Position_Y end  else begin x22:=Pac1[j].Position_X;
y22:=Pac1[j].Position_Y; end;
    if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
    if (x11=0) then
    begin
      b1:=y11;  a1:=(y12-b1)/x12;
    end
    else
    begin
      b1:=(x11*y12-x12*y11)/(x11-x12);   a1:=(y11-b1)/x11;
    end;
    if (x21=0) then
    begin
      b2:=y21;  a2:=(y22-b2)/x22;
    end
    else
    begin
      b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
    end;
    if (a1<>a2) then
      begin
        y0:=(b2*a1-b1*a2)/(a1-a2);  x0:=(y0-b1)/a1;
        minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
        maxx1:=Max(x11,x12);  maxx2:=Max(x21,x22);
        miny1:=Min(y11,y12);   miny2:=Min(y21,y22);
        maxy1:=Max(y11,y12);  maxy2:=Max(y21,y22);
        if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then
          begin goto enen1;   end;
      end;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21;  a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      x0:=x11;  y0:=a2*x0+b2;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then goto enen1;
    end
    else if ((x11<>x12) and (x21=x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
      end;
      x0:=x21;  y0:=a1*x0+b1;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then    goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11=y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      y0:=y11;  x0:=(y0-b2)/a2;
      minx1:=Min(x11,x12); minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then   goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
      end;
      y0:=y21; x0:=(y0-b1)/a1;
      minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then      goto enen1;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      y0:=y21; x0:=x11;
      miny1:=Min(y11,y12);  minx1:=Min(x21,x22);
      maxy1:=Max(y11,y12); maxx1:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1;
```

```
    end
    else if ((x11<>x12) and (x21=x22) and (y11=y12) and (y21<>y22)) then
    begin
      y0:=y11;
      x0:=x21;
      miny1:=Min(y21,y22);  minx1:=Min(x11,x12);
      maxy1:=Max(y21,y22); maxx1:=Max(x11,x12);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then  goto enen1;
    end;
    goto enen;
    enen1:
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=x0) and (Kontr_T[net].Position_Y=y0)) then
break;
    if (net<>kt) then begin goto enen; end;
end;
    jni3:=j;
jni4:    if ((sqrt(sqr(round(x0)-Pac1[jni3].Position_X)+sqr(round(y0)-
Pac1[jni3].Position_Y)))>=(sqrt(sqr(round(x0)-
Pac1[jni3-1].Position_X)+sqr(round(y0)-Pac1[jni3-1].Position_Y)))) then
    begin jni:=jni3; end else begin jni:=jni3-1; end;
    jni1:=1;
jni2:    net:=round(sin(0));
    for net:=nnkt to kt-1 do
      if (Kontr_T[net].ID=Pac1[jni].ID) then break;
    if ((net=kt) or (nnkt=kt)) then
    begin
      Kontr_T[kt].ID:=Pac1[jni].ID; Kontr_T[kt].Position_X:=round(x0);
Kontr_T[kt].Position_Y:=round(y0); if (jni3=j) then begin
Kontr_T[kt].Pressure:=round((Pac1[j].Pressure+Pac1[j-1].Pressure)/2); end
else if (jni3=ni) then begin
Kontr_T[kt].Pressure:=round((Pac1[ni].Pressure+Pac1[ni-1].Pressure)/2);
end; Kontr_T[kt].type_t:=3;  Kontr_T[kt].N_simv:=Pac1[jni].N_simv;
      kt:=kt+1; p:=p+1; goto enen;
    end;
    if (jni1=1) then begin jni1:=jni+1; if (jni=jni3) then begin jni:=jni-1; end
else if (jni=jni3-1) then begin jni:=jni+1; end; goto jni2; end
    else if((jni1=2) and (jni3=j)) then begin jni3:=ni; goto jni4; end else goto
enen;
enen:
  end;
  end;
end;
pr:=Pac1[j].Pressure;
end;
////////////////  }  Визначення КТ  перетинання  //////////////
/////  Видалення кутових КТ, які знаходяться поруч (по дліні)   {
if(chb_bl.Checked=true) then
begin
  ut:=0; ut1:=0; ut2:=0; ij:=0; nvt:=1;
  uttt5: while (ut<=(kt-1)) do
          begin
            if (Kontr_T[ut].type_t=2) then break;
            ut:=ut+1;
          end;
  if(ut=kt) then goto ee;
if(nvt=1) then begin ut1:=ut; nvt:=2; ut:=ut+1; goto uttt5; end else ut2:=ut;
  if (Kontr_T[ut1].N_simv<>Kontr_T[ut2].N_simv) then begin ut1:=ut2;
nvt:=2; ut:=ut1+1; goto uttt5; end;
        ut12:=ut1;
        mu12: while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=Kontr_T[ut12].ID) then break;
        ij:=ij+1;
        end;
        if(ut12=ut1) then begin utt:=ij; utt1:=ij; ut12:=ut2; ij:=ij+1; goto mu12;
end else uttk:=ij;
  PR1:=0;
  while (utt1<uttk) do
  begin
     if(Pac1[utt1].Pressure=0) then break;
     PR1:=PR1+sqrt(sqr(Pac1[utt1].Position_X-Pac1[utt1+1].Position_X)+
sqr(Pac1[utt1].Position_Y-Pac1[utt1+1].Position_Y));
  utt1:=utt1+1;
  end;
    if(utt1<>uttk) then begin ut1:=ut2; nvt:=2; ut:=ut1+1; goto uttt5; end;
    if ((PR1<=StrToFloat(ed_bl.Text)) and (PR1>0)) then
    begin
      if ((Sign(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)<>Sign(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)) and ((Sign(Pac1[utt].Position_Y-Pac1[utt-
1].Position_Y)= Sign(Pac1[utt+1].Position_Y-Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_Y<>Pac1[utt-1].Position_Y) then
per11:=180*(arctan(abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)/abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_Y<>Pac1[utt].Position_Y) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)/abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))/3.1415926535897932385 else per12:=90;
      end;
      if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)=
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)<>Sign(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_X<>Pac1[utt-1].Position_X) then
per11:=180*(arctan(abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)/abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_X<>Pac1[utt].Position_X) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)/abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)))/3.1415926535897932385 else per12:=90;
      end;
      per:=180-per11-per12;
      if ((Sign(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)<>Sign(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)) and ((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)=Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_Y<>Pac1[uttk-1].Position_Y) then
per21:=180*(arctan(abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)/abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_Y<>Pac1[uttk].Position_Y) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)/abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))/3.1415926535897932385 else per22:=90;
      end;
      if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-1].Position_X)=
Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)<>Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_X<>Pac1[uttk-1].Position_X) then
per21:=180*(arctan(abs(Pac1[uttk].Position_Y-Pac1[uttk-
1].Position_Y)/abs(Pac1[uttk].Position_X-Pac1[uttk-
1].Position_X)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_X<>Pac1[uttk].Position_X) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)/abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)))/3.1415926535897932385 else per22:=90;
      end;
      per1:=180-per21-per22;
      if (per<=per1) then
      begin
        for u:=ut2 to kt-2 do
        begin
          Kontr_T[u]:=Kontr_T[u+1];
        end;
      end
       else
       begin
         for u:=ut1 to kt-2 do
        begin
          Kontr_T[u]:=Kontr_T[u+1];
        end;
        ut1:=ut2-1;
       end;
```

```
      kt:=kt-1;
    end
    else begin ut1:=ut2; ut:=ut1+1; end;
    nvt:=2;     ij:=ij+1;
    goto uttt5;
ee: end;
//  }  Видалення кутових КТ, які знаходяться поруч (по длині)
/////////////////  Вивод на экран КТ  {  ////////////////////////
for ww:=0 to simv do
begin
  KKontr_T[ww]:=0;
end;
for w:=0 to kt-1 do
begin
if (Kontr_T[w].type_t=1) then Image1.Canvas.pen.Color:=clRed
else if (Kontr_T[w].type_t=2) then Image1.Canvas.pen.Color:=clBlue
else if (Kontr_T[w].type_t=3) then Image1.Canvas.pen.Color:=clGreen;
Image1.Canvas.Ellipse(round(Kontr_T[w].Position_X*Image1.Width/8000)-
3, Image1.Height-round(Kontr_T[w].Position_Y*Image1.Height/6000)-3,
round(Kontr_T[w].Position_X*Image1.Width/8000)+3,Image1.Height-
round(Kontr_T[w].Position_Y*Image1.Height/6000)+3);
Image1.Canvas.TextOut(round(Kontr_T[w].Position_X*Image1.Width/8000)
-27,Image1.Height-round(Kontr_T[w].Position_Y*Image1.Height/6000)-27,
IntToStr(w)+'  '+ IntToStr(Kontr_T[w].Position_X)+'  '+
IntToStr(Kontr_T[w].Position_Y));
Image1.Canvas.pen.Color:=clBlack;
nn:=Pac1[0].ID;
for ww:=0 to simv do
begin
  if ((Kontr_T[w].ID>=nn) and (trr[ww]>=Kontr_T[w].ID)) then
  KKontr_T[ww]:=KKontr_T[ww]+1;
  nn:=trr[ww];
end;
end;
/////////////////  }  Вивод на экран КТ  ////////////////////////////
/////////////////  }  Визначення та вивод на экран точок і КТ
bt_finish.SetFocus;
end;
procedure Tfm_read.bt_finishKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
  var i:integer;
begin
if(Key=VK_ESCAPE) then
begin
chb_m.State:=cbUnchecked; chb_pov.State:=cbUnchecked;
chb_pov_1.State:=cbUnchecked; chb_sdv_x.State:=cbUnchecked;
chb_sdv_y.State:=cbUnchecked;
ed_m.Text:=''; ed_pov.Text:=''; ed_pov_1.Text:=''; ed_sdv_x.Text:='';
ed_sdv_y.Text:='';
DriveComboBox2.Visible:=true; DirectoryListBox2.Visible:=true;
FileListBox1.Visible:=true; Image1.Visible:=false;
Image1.Canvas.pen.Color:=clWhite; Image1.Canvas.Brush.Color:=clWhite;
Image1.Canvas.Brush.Style:=bsSolid;
Image1.Canvas.Rectangle(0,0,(Image1.width),(Image1.height));
for i:=0 to (nom_zap-1) do
begin
Pac[i].ID:=0; Pac[i].Fam:=''; Pac[i].Imya:=''; Pac[i].Otch:='';
Pac[i].Context:=0; Pac[i].Status:=0; Pac[i].Cursor:=0; Pac[i].Buttons:=0;
Pac[i].Position_X:=0; Pac[i].Position_Y:=0; Pac[i].Pressure:=0;
Pac[i].TangPress:=0; Pac[i].Orientation_orAzimuth:=0;
Pac[i].Orientation_orAltitude:=0; Pac[i].Orientation_orTwist:=0;
Pac[i].Rotation_roPitch:=0; Pac[i].Rotation_roRoll:=0;
Pac[i].Rotation_roYaw:=0; Pac[i].Data:=0; Pac[i].Time:=0;
Pac[i].Time_ms:=0;
end;
end;
end;
procedure Tfm_read.FormCreate(Sender: TObject);
begin
GetDir(0, S);
bt_finish.Width:=0;
end;
procedure Tfm_read.ed_xKeyDown(Sender: TObject; var Key: Word; Shift:
TShiftState);
begin
if(Key=VK_RETURN) then
begin
bt_ok.Click;
end;
end;
procedure Tfm_read.ed_kol_tChange(Sender: TObject);
begin
if (StrToInt(ed_kol_t.Text)<>0) then ed_x.Text:=IntToStr(0);
end;
procedure Tfm_read.ed_xChange(Sender: TObject);
begin
if (StrToFloat(ed_x.Text)<>0) then ed_kol_t.Text:=IntToStr(0);
end;
procedure Tfm_read.ed_kol_tKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9'])  then Key:=#0;
end;
procedure Tfm_read.ed_xKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.'])  then Key:=#0;
end;
procedure Tfm_read.ed_sdv_xKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','-'])  then Key:=#0;
end;
procedure Tfm_read.ed_povKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.','-'])  then Key:=#0;
end;
end.
```

Файл Unit4.pas:

```
unit Unit4;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, FileCtrl;
type
  Tfm_setup = class(TForm)
    DriveComboBox2: TDriveComboBox;
    DirectoryListBox2: TDirectoryListBox;
    FileListBox1: TFileListBox;
    bt_ok: TButton;
    Label1: TLabel;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure bt_okClick(Sender: TObject);
    procedure DriveComboBox2Change(Sender: TObject);
    procedure DirectoryListBox2Change(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_setup: Tfm_setup;
implementation
uses Unit1, Unit2, Unit3;
{$R *.dfm}
procedure Tfm_setup.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Action:=caFree;
end;
procedure Tfm_setup.bt_okClick(Sender: TObject);
var path:AnsiString;
f1:TextFile;
begin
///////// Перевірка чи правильно вказана папка, в котрій знаходится
///////// файл winrar.exe
if (FileSearch('winrar.exe',DirectoryListBox2.Directory)<>'') then
begin
/////    Зберігання в файлі path.txt вказаного місцязнаходження
///// файла winrar.exe
AssignFile(f1, S+'\path.txt'); Rewrite(f1); path:=DirectoryListBox2.Directory;
write(f1,path); CloseFile(f1); Close();
end
else ShowMessage('В цій папці немає WinRar.exe.');
end;
procedure Tfm_setup.DriveComboBox2Change(Sender: TObject);
begin
DirectoryListBox2.Drive:=DriveComboBox2.Drive;
end;
```

```
procedure Tfm_setup.DirectoryListBox2Change(Sender: TObject);
begin
FileListBox1.Directory:=DirectoryListBox2.Directory;
end;
procedure Tfm_setup.FormCreate(Sender: TObject);
begin
FileListBox1.Enabled:=false;
end;
end
```

Файл Unit5.pas:

```
unit Unit5;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, StdCtrls, Grids, FileCtrl,math, ExtCtrls, TeeProcs,
TeEngine, Chart, Series;
type
  Tfm_tochki = class(TForm)
    ds_tb_main: TDataSource;  tb_main: TTable; tb_mainID: TIntegerField;
    tb_mainFam: TStringField; tb_mainImya: TStringField;
    tb_mainOtch: TStringField; tb_mainContext: TIntegerField;
    tb_mainStatus: TIntegerField; tb_mainCursor: TIntegerField;
    tb_mainButtons: TIntegerField; tb_mainPosition_X: TIntegerField;
    tb_mainPosition_Y: TIntegerField; tb_mainPressure: TIntegerField;
    tb_mainTangPress: TIntegerField;
    tb_mainOrientation_orAzimuth: TIntegerField;
    tb_mainOrientation_orAltitude: TIntegerField;
    tb_mainOrientation_orTwist: TIntegerField;
    tb_mainRotation_roPitch: TIntegerField;
    tb_mainRotation_roRoll: TIntegerField;
    tb_mainRotation_roYaw: TIntegerField; tb_mainData: TDateField;
    tb_mainTime: TTimeField; tb_mainTime_ms: TIntegerField;
    bt_ok: TButton; str_gr: TStringGrid;
    DriveComboBox2: TDriveComboBox;
    DirectoryListBox2: TDirectoryListBox;
    str_gr_1: TStringGrid; ch_graph: TChart; ed_min: TEdit; ed_max: TEdit;
    Label1: TLabel; Label2: TLabel; Label3: TLabel; str_gr_kt: TStringGrid;
    Series1: TFastLineSeries; Series2: TFastLineSeries;
    Series3: TFastLineSeries; Series4: TFastLineSeries;
    Series5: TFastLineSeries; Series6: TFastLineSeries;
    Series7: TFastLineSeries; Series8: TFastLineSeries;
    Series9: TFastLineSeries; Series10: TFastLineSeries;
    Series11: TFastLineSeries; Series12: TFastLineSeries;
    Series13: TFastLineSeries; Series14: TFastLineSeries;
    Series15: TFastLineSeries;
    chb_k: TCheckBox; chb_bl: TCheckBox; chb_r: TCheckBox;
    ed_bl: TEdit; chb_p: TCheckBox; chb_s: TCheckBox;
    chb_t: TCheckBox; ed_t: TEdit; chb_x: TCheckBox; ed_kol_t: TEdit;
    ed_x: TEdit; chb_povt: TCheckBox; chb_ramka: TCheckBox;
    chb_ramka_1: TCheckBox; chb_m: TCheckBox; ed_m: TEdit;
    chb_pov: TCheckBox;  ed_pov: TEdit; chb_sdv_x: TCheckBox;
    ed_sdv_x: TEdit; chb_sdv_y: TCheckBox; ed_sdv_y: TEdit;
    chb_sdv_y_1: TCheckBox; chb_simv_m: TCheckBox;
    ed_pov_1: TEdit; chb_pov_1: TCheckBox;  str_gr_t: TStringGrid;
    Label4: TLabel; Label5: TLabel;  Label6: TLabel;
    Label7: TLabel;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure bt_okClck(Sender: TObject);
    procedure DriveComboBox2Change(Sender: TObject);
    procedure str_grClick(Sender: TObject);
    procedure str_gr_1Click(Sender: TObject);
    procedure ed_minKeyPress(Sender: TObject; var Key: Char);
    procedure ed_maxKeyPress(Sender: TObject; var Key: Char);
    procedure FormCreate(Sender: TObject);
    procedure str_gr_ktClick(Sender: TObject);
    procedure str_gr_tClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_tochki: Tfm_tochki;
  kol:integer;
implementation
uses Unit1, Unit2, Unit3, Unit4;
{$R *.dfm}
procedure Tfm_tochki.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Action:=caFree;
end;
procedure Tfm_tochki.bt_okClck(Sender: TObject);
/////    Функція, яка викликається при натисканні на кнопку форми “ОК”
var ii,jj,iff,kol1,kolf:integer;
dr: array [0..32000] of AnsiString;
d:AnsiString;
SR: TSearchRec;
pr,x,y,dx1,dx2,dy1,dy2,dx,dy,kt,kt1,i,ni,p,nii,j,j1,kz,w,nol,u,tt,kol_t,sii, simv,
simv_s, net, min_rr, nmin_rr, nn, ij, min_x_r, max_x_r, min_y_r, max_y_r,
ED_X1, ED_Y1, XS, YS, XL, YL, XR, YR, ij1, MM, ED_X11, ED_Y11,
nnk, nin, nnkt, jni, jni1, jni3, ut,ut1,ut2, nvt, ut12, utt, utt1, uttk,
ww,jn:integer;
kx: array [0..9] of integer;
ky: array [0..9] of integer;
kp: array [0..9] of integer;
min_x_r1: array [0..100] of integer;
max_x_r1: array [0..100] of integer;
min_y_r1: array [0..100] of integer;
max_y_r1: array [0..100] of integer;
trr: array [0..100] of integer;
ss,ss1,ss2:Extended;
sss, ekt:boolean;
x11,x12,x21,x22,x0,y11,y12,y21,y22,y0,a1,a2,b1,b2,minx1,minx2,maxx1,ma
xx2,miny1,miny2,maxy1,maxy2,tt1,per,per1,sssss, PR2, PR1, ED_M1,
ED_P1, ED_M11, per11,per12,per21,per22:real;
Label xv, xv1,k1, slt2, nt, endfn, rrr_3, nt1, enen, enen1, jni2, jni4, uttt5, ee,
mu12,slt;
begin
///////   Ініціалізація таблиць
str_gr.ColCount:=301; str_gr_1.ColCount:=301; str_gr_t.ColCount:=3001;
str_gr_kt.ColCount:=31;
jj:=0;
for ii:=0 to kol do
begin
str_gr.Cells[1,ii]:=''; str_gr.Cells[2,ii]:=''; str_gr_1.Cells[1,ii]:='';
str_gr_1.Cells[2,ii]:=''; str_gr_t.Cells[1,ii]:=''; str_gr_t.Cells[2,ii]:='';
str_gr_kt.Cells[1,ii]:=''; str_gr_kt.Cells[2,ii]:='';
end;
for ii:=2 to 300 do
begin
str_gr.Cells[ii+1,0]:=IntToStr(ii); str_gr_1.Cells[ii+1,0]:=IntToStr(ii);
end;
for ii:=2 to 3000 do
begin
str_gr_t.Cells[ii+1,0]:=IntToStr(ii);
end;
for ii:=1 to 30 do
begin
str_gr_kt.Cells[ii+2,0]:=IntToStr(ii);
end;
kol:=0;
ii:=0;
///////   Пошук даних для аналіза
d:=DirectoryListBox2.Directory;
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
 str_gr.RowCount:=ii+2;  str_gr.Cells[0,ii+1]:=SR.Name;
 str_gr_1.RowCount:=ii+2;  str_gr_1.Cells[0,ii+1]:=SR.Name;
 str_gr_kt.RowCount:=ii+2;  str_gr_kt.Cells[0,ii+1]:=SR.Name;
 str_gr_t.RowCount:=ii+2;  str_gr_t.Cells[0,ii+1]:=SR.Name;
 for sii:=1 to 301 do
  begin
  str_gr.Cells[sii,ii+1]:='';   str_gr_1.Cells[sii,ii+1]:='';
  if (sii<=31) then str_gr_kt.Cells[sii,ii+1]:='';
end;
for sii:=1 to 3001 do
  begin
  str_gr_t.Cells[sii,ii+1]:='';
end;
str_gr.Cells[1,0]:='Всього зразків';  str_gr.Cells[2,0]:='Відібрано зразків';
str_gr_1.Cells[1,0]:='Всього зразків';
```

```
str_gr_1.Cells[2,0]:='Відібрано зразків';
str_gr_kt.Cells[1,0]:='Всього зразків';
str_gr_kt.Cells[2,0]:='Відібрано зразків';
str_gr_t.Cells[1,0]:='Всього зразків';
str_gr_t.Cells[2,0]:='Відібрано зразків';
 if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
 ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
 str_gr.Refresh();  str_gr_1.Refresh();  str_gr_kt.Refresh();  str_gr_t.Refresh();
FindClose(SR);
kol:=ii;
kol1:=0;
k1: for jj:=kol1 to kol do
begin
d:=dr[jj];
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
 str_gr.RowCount:=ii+2;  str_gr.Cells[0,ii+1]:=SR.Name;
 str_gr_1.RowCount:=ii+2;  str_gr_1.Cells[0,ii+1]:=SR.Name;
 str_gr_kt.RowCount:=ii+2;  str_gr_kt.Cells[0,ii+1]:=SR.Name;
 str_gr_t.RowCount:=ii+2;  str_gr_t.Cells[0,ii+1]:=SR.Name;
if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
 ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
FindClose(SR);
end;
kol1:=kol;
kol:=ii-1;
if (kol1<>kol) then goto k1;
for ii:=0 to kol do
begin
kolf:=0;
if (Length(dr[ii])=3) then begin iff:=FindFirst(dr[ii] +'*.db',faAnyFile, SR);
end
else begin iff:=FindFirst(dr[ii] +'\*.db',faAnyFile, SR); end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr<>faDirectory)) then
 begin
 kolf:=kolf+1;
 str_gr.Cells[1,ii+1]:=IntToStr(kolf);  str_gr_1.Cells[1,ii+1]:=IntToStr(kolf);
str_gr_kt.Cells[1,ii+1]:=IntToStr(kolf); str_gr_t.Cells[1,ii+1]:=IntToStr(kolf);
pr:=0; x:=0; y:=0; dx1:=0; dx2:=0; dy1:=0; dy2:=0; kt:=0; i:=0; nom_zap:=0;
p:=0; kz:=0;
if (Length(d)=3) then
begin tb_main.TableName:=dr[ii]+SR.Name; end
else begin tb_main.TableName:=dr[ii] +'\'+SR.Name; end;
str_gr.Refresh(); str_gr_1.Refresh(); str_gr_t.Refresh(); str_gr_kt.Refresh();
tb_main.Open();
while tb_main.Eof<>true do
begin
////////////////// Створення масива точок, якщо виконувати згладжування  {
if (chb_s.Checked=true)
then
begin
if (tb_mainPressure.AsInteger=0) then
  begin
  if (kz>9) then
    begin
    for j1:=kz-5 to kz-2 do
    begin
    Pac1[i].ID:=i; Pac1[i].Position_X:=kx[j1]; Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1];  i:=i+1; nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end
  else if(kz>0) then
    begin
    for j1:=0 to kz-1 do
    begin
    Pac1[i].ID:=i; Pac1[i].Position_X:=kx[j1];  Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1]; i:=i+1;  nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end;
  Pac1[i].ID:=tb_mainID.AsInteger;
  Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
  Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
  Pac1[i].Pressure:=tb_mainPressure.AsInteger;
  i:=i+1;
  nom_zap:=nom_zap+1;
  end
else
if(kz<=4) then
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
i:=i+1;
nom_zap:=nom_zap+1;
kz:=kz+1;
end
else if ((kz>4) and (kz<9)) then
begin
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
kz:=kz+1;
end
else if (kz>=9) then
begin
kx[9]:=tb_mainPosition_X.AsInteger;
ky[9]:=tb_mainPosition_Y.AsInteger;
kp[9]:=tb_mainPressure.AsInteger;
Pac1[i].Position_X:=0; Pac1[i].Position_Y:=0; Pac1[i].Pressure:=0;
for j1:=0 to 9 do
  begin
  Pac1[i].Position_X:=Pac1[i].Position_X+kx[j1];
  Pac1[i].Position_Y:=Pac1[i].Position_Y+ky[j1];
  Pac1[i].Pressure:=Pac1[i].Pressure+kp[j1];
  if (j1<>0) then
    begin
    kx[j1-1]:=kx[j1];  ky[j1-1]:=ky[j1];  kp[j1-1]:=kp[j1];
    end;
  end;
Pac1[i].ID:=i;
Pac1[i].Position_X:=round(Pac1[i].Position_X/10);
Pac1[i].Position_Y:=round(Pac1[i].Position_Y/10);
Pac1[i].Pressure:=round(Pac1[i].Pressure/10);
i:=i+1;
nom_zap:=nom_zap+1;
if (kz=9) then kz:=kz+1;
end;
end
////////////////// }  Створення масива точок, якщо виконувати згладжування
////////////////// Створення масива точок, якщо не виконувати згладжування
}
else
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
i:=i+1;
nom_zap:=nom_zap+1;
end;
/////////////// }  Створення масива точок, якщо не виконувати
згладжування
tb_main.Next();
end;
tb_main.Close();
```

```
//////////////// Видалення повторів  {  ////////////////////////////
if(chb_povt.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].Position_X=Pac1[j+1].Position_X) and
(Pac1[j].Position_Y=Pac1[j+1].Position_Y) and (j<(i-1))) then
    begin
      u:=j+2;
      while ((Pac1[u].Pressure<>0) and
(Pac1[j].Position_X=Pac1[u].Position_X) and
(Pac1[j].Position_Y=Pac1[u].Position_Y) and (u<=(i-1))) do
      begin
        u:=u+1;
      end;
      nol:=u-1;
      begin
        i:=i-nol+j;
        nom_zap:=nom_zap-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
end;
//////////////// } Видалення повторів ////////////////////////////
//////////////// Видалення випадкових точок  {  ////////////////////////////
if(chb_t.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (((Pac1[j].Pressure=0) and (j<(i-1))) or (j=0)) then
    begin
      if(Pac1[j].Pressure=0) then jn:=j else jn:=j-1;
      slt:nol:=jn;
      for u:=jn+1 to (jn+1+StrToInt(ed_t.Text)) do
        if (u=i) then begin nol:=u-1; break; end else if (Pac1[u].Pressure=0) then
nol:=u;
      if(nol<>jn) then
      begin
        i:=i-nol+jn;
        nom_zap:=nom_zap-nol+jn;
        for u:=jn+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-jn];
        end;
        if (jn<(i-1)) then goto slt;
      end;
    end;
  end;
end;
//////////////// } Видалення випадкових точок ////////////////////////////
//////////////// Видалення хвостів за кількістю точок { ////////////
tt:=StrToInt(ed_kol_t.Text);
if ((chb_x.Checked=true) and (StrToInt(ed_kol_t.Text)<>0)) then
begin
xv:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;
      for u:=j+1 to (j+1+tt) do
        if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
          else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then nol:=u;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
//////////////// } Видалення хвостів за кількістю точок //////////////////
//////////////// Видалення хвостів за длиною  {  ////////////////////////////
tt1:=StrToFloat(ed_x.Text);
if ((chb_x.Checked=true) and (StrToFloat(ed_x.Text)<>0)) then
begin
xv1:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;  u:=j+1; PR2:=0;
      while (PR2<=tt1) do
      begin
        if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
          else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then begin nol:=u;  end;
        u:=u+1;  PR2:=PR2+sqrt(sqr(Pac1[u].Position_X-Pac1[u-
1].Position_X)+ sqr(Pac1[u].Position_Y-Pac1[u-1].Position_Y));
      end;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
//////////////// } Видалення хвостів за длиною ////////////////////////////
//////////////// Визначення меж букв {  ////////////////////////////
begin
  simv:=0;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r1[simv]:=Pac1[j].Position_X;
      max_x_r1[simv]:=Pac1[j].Position_X;
      min_y_r1[simv]:=Pac1[j].Position_Y;
      max_y_r1[simv]:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
      if (Pac1[j].Pressure<>0) then
      begin
        if (Pac1[j].Position_X<min_x_r1[simv]) then
min_x_r1[simv]:=Pac1[j].Position_X;
        if (Pac1[j].Position_X>max_x_r1[simv]) then
max_x_r1[simv]:=Pac1[j].Position_X;
        if (Pac1[j].Position_Y<min_y_r1[simv]) then
min_y_r1[simv]:=Pac1[j].Position_Y;
        if (Pac1[j].Position_Y>max_y_r1[simv]) then
max_y_r1[simv]:=Pac1[j].Position_Y;
      end
      else
      begin
        trr[simv]:=Pac1[j].ID;
        if((j<>0) and (j<>(i-1))) then
        begin
          simv:=simv+1;
          min_x_r1[simv]:=Pac1[j+1].Position_X;
          max_x_r1[simv]:=Pac1[j+1].Position_X;
          min_y_r1[simv]:=Pac1[j+1].Position_Y;
          max_y_r1[simv]:=Pac1[j+1].Position_Y;
        end;
      end;
```

```
  end;
simv_s:=simv;
if (simv<5) then goto endfn;
rrr_3: if (simv>5) then
  begin
    min_rr:=min_x_r1[1]-max_x_r1[0];
    nmin_rr:=1;
    for j:=1 to simv do
    begin
      if ((min_x_r1[j]-max_x_r1[j-1])<min_rr) then begin min_rr:=min_x_r1[j]-
max_x_r1[j-1]; nmin_rr:=j; end;
    end;
    max_x_r1[nmin_rr-1]:=Max(max_x_r1[nmin_rr-1],max_x_r1[nmin_rr]);
    min_x_r1[nmin_rr-1]:=Min(min_x_r1[nmin_rr-1],min_x_r1[nmin_rr]);
    max_y_r1[nmin_rr-1]:=Max(max_y_r1[nmin_rr-1],max_y_r1[nmin_rr]);
    min_y_r1[nmin_rr-1]:=Min(min_y_r1[nmin_rr-1],min_y_r1[nmin_rr]);
    for j:=nmin_rr to simv do
    begin
      max_x_r1[j]:=max_x_r1[j+1];  min_x_r1[j]:=min_x_r1[j+1];
      max_y_r1[j]:=max_y_r1[j+1];  min_y_r1[j]:=min_y_r1[j+1];
      trr[j-1]:=trr[j];
    end;
    simv:=simv-1;
    goto rrr_3;
  end;
end;
  nn:=Pac1[0].ID;
  ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    Pac1[ij].N_simv:=jj;
        end;
  nn:=trr[jj];
  end;
////////////////  }  Визначення меж букв  ////////////////////////////
////////////////  Зсув{  ////////////////////////////
if((chb_sdv_x.Checked=true) or (chb_sdv_y.Checked=true)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  ED_X1:=StrToInt(ed_sdv_x.Text);  ED_Y1:=StrToInt(ed_sdv_y.Text);
  if ((chb_sdv_x.Checked=true) and (ED_X1=777)) then
ED_X1:=round((8000-(max_x_r-min_x_r))/2)-min_x_r;
  if ((chb_sdv_y.Checked=true) and (ED_Y1=777)) then
ED_Y1:=round((6000-(max_y_r-min_y_r))/2)-min_y_r;
  for j:=0 to i-1 do
  begin
    if(chb_sdv_x.Checked=true) then
Pac1[j].Position_X:=Pac1[j].Position_X+ED_X1;
    if(chb_sdv_y.Checked=true) then
Pac1[j].Position_Y:=Pac1[j].Position_Y+ED_Y1;
  end;
end;
////////////////  }  Зсув  ////////////////////////////
////////////////  Масштабування{  ////////////////////////////
if((chb_m.Checked=true) and (StrToFloat(ed_m.Text)>0)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  ED_M1:=StrToFloat(ed_m.Text);
  if (ED_M1=777) then
  ED_M1:=
  Min (  Min ((0-((max_x_r-min_x_r)/2+min_x_r))/(min_x_r-((max_x_r-
min_x_r)/2+min_x_r)),(8000-((max_x_r-min_x_r)/2+min_x_r)/(max_x_r-
((max_x_r-min_x_r)/2+min_x_r)))), Min ((0-((max_y_r-min_y_r)/2+
min_y_r))/(min_y_r-((max_y_r-min_y_r)/2+min_y_r)),(6000-((max_y_r-
min_y_r)/2+min_y_r)/(max_y_r-((max_y_r-min_y_r)/2+min_y_r))))   );
  for j:=0 to i-1 do
  begin
    Pac1[j].Position_X:=round((Pac1[j].Position_X-((max_x_r-min_x_r)/2+
min_x_r))*ED_M1+((max_x_r-min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((Pac1[j].Position_Y-((max_y_r-min_y_r)/2+
min_y_r))*ED_M1+((max_y_r-min_y_r)/2+min_y_r));
  end;
end;
////////////////  }  Масштабування  ////////////////////////////
////////////////  Поворот  {  ////////////////////////////
if(chb_pov.Checked=true) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  for j:=0 to i-1 do
  begin
    XS:=Pac1[j].Position_X; YS:=Pac1[j].Position_Y;
    Pac1[j].Position_X:=round((XS-((max_x_r-
min_x_r)/2+min_x_r))*cos(StrToFloat(ed_pov.Text)*
3.14159265358979323385/180.0)+(YS-((max_y_r-min_y_r)/2+ min_y_r))*
sin(StrToFloat(ed_pov.Text)*PI/180.0))+round(((max_x_r-
min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((YS-((max_y_r-
min_y_r)/2+min_y_r))*cos(StrToFloat(ed_pov.Text)*
3.14159265358979323385/180.0)-(XS-((max_x_r-min_x_r)/2+ min_x_r))*
sin(StrToFloat(ed_pov.Text)*PI/180.0))+round(((max_y_r-
min_y_r)/2+min_y_r));
  end;
end;
////////////////  }  Поворот  ////////////////////////////
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
////////////////  Посимвольний поворот  {  ////////////////////////////
```

```
if(chb_pov_1.Checked=true) then
begin
ED_P1:=StrToFloat(ed_pov_1.Text);
if (ED_P1=777) then ED_P1:=180.0*(arctan((YR-YL)/(XR-
XL)))/3.1415926535897932385;
 nn:=Pac1[0].ID;  ij:=0;
 for jj:=0 to simv do
 begin
 for j:=nn to trr[jj]-1 do
 begin
   while (ij<=(i-1)) do
   begin
   if (Pac1[ij].ID>=j) then break;
   ij:=ij+1;
   end;
   if (Pac1[ij].Pressure<>0) then
   begin
     min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
     min_y_r1[jj]:=Pac1[ij].Position_Y; max_y_r1[jj]:=Pac1[ij].Position_Y;
     ij1:=ij;
     break;
   end;
 end;
 for j:=nn to trr[jj]-1 do
 begin
      while (ij<=(i-1)) do
      begin
      if (Pac1[ij].ID>=j) then break;
      ij:=ij+1;
      end;
      if (Pac1[ij].Pressure<>0) then
      begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
      end;
 end;
j:=nn;
while (j<=trr[jj]-1) do
       begin
        ij:=ij1;
         while (ij<=(i-1)) do
         begin
           if (Pac1[ij].ID>=j) then break;
           ij:=ij+1;
         end;
         XS:=Pac1[ij].Position_X;
         YS:=Pac1[ij].Position_Y;
         Pac1[ij].Position_X:=round((XS-((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]))*cos(ED_P1*3.1415926535897932385/180.0)+(YS-
((max_y_r1[jj]-min_y_r1[jj])/2+ min_y_r1[jj]))*sin(ED_P1*PI/180.0))+
round(((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]));
         Pac1[ij].Position_Y:=round((YS-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*cos(ED_P1*3.1415926535897932385/180.0)
-(XS-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))* sin(ED_P1*PI/180.0))+
round(((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]));
         if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
       end;
 nn:=trr[jj];
 end;
end;
//////////////////  }  Посимвольний поворот  ////////////////////////////
//////////////////  Посимвольний зсув по Y  {  ////////////////////////////
if(chb_sdv_y_1.Checked=true) then
begin
 nn:=Pac1[0].ID;
 for jj:=0 to simv do
 begin
 for j:=nn to trr[jj]-1 do
 begin
   ij:=0;
   while (ij<=(i-1)) do
   begin
   if (Pac1[ij].ID>=j) then break;
   ij:=ij+1;
   end;
   if (Pac1[ij].Pressure<>0) then
   begin
     min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
     min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
     break;
   end;
 end;
 for j:=nn to trr[jj]-1 do
 begin
      ij:=0;
      while (ij<=(i-1)) do
      begin
      if (Pac1[ij].ID>=j) then break;
      ij:=ij+1;
      end;
      if (Pac1[ij].Pressure<>0) then
      begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
      end;
 end;
 nn:=trr[jj];
 end;
 MM:=max_y_r1[0];
 for jj:=0 to simv do
 begin
  MM:=Max(MM,max_y_r1[jj]);
 end;
 nn:=Pac1[0].ID;
 for jj:=0 to simv do
 begin
j:=nn;
while (j<=trr[jj]-1) do
       begin
        ij:=0;
         while (ij<=(i-1)) do
         begin
           if (Pac1[ij].ID>=j) then break;
           ij:=ij+1;
         end;
         Pac1[ij].Position_Y:=Pac1[ij].Position_Y+(MM-max_y_r1[jj]);
         if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
       end;
 nn:=trr[jj];
 end;
end;
//////////////////  }  Посимвольний зсув по Y  ////////////////////////////
//////////////////  Посимвольне масштабування {  ////////////////////////////
if(chb_simv_m.Checked=true) then
begin
 nn:=Pac1[0].ID;
 for jj:=0 to simv do
 begin
 for j:=nn to trr[jj]-1 do
 begin
  ij:=0;
  while (ij<=(i-1)) do
  begin
  if (Pac1[ij].ID>=j) then break;
  ij:=ij+1;
  end;
  if (Pac1[ij].Pressure<>0) then
  begin
    min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
    min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
    break;
  end;
 end;
 for j:=nn to trr[jj]-1 do
 begin
      ij:=0;
```

```
        while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=j) then break;
        ij:=ij+1;
        end;
        if (Pac1[ij].Pressure<>0) then
        begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
        end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_X11:=round((8000-(max_x_r1[jj]-min_x_r1[jj]))/2)-min_x_r1[jj];
  ED_Y11:=round((6000-(max_y_r1[jj]-min_y_r1[jj]))/2)-min_y_r1[jj];
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
           Pac1[ij].Position_X:=Pac1[ij].Position_X+ED_X11;
           Pac1[ij].Position_Y:=Pac1[ij].Position_Y+ED_Y11;
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
    nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
        ij:=0;
        while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=j) then break;
        ij:=ij+1;
        end;
        if (Pac1[ij].Pressure<>0) then
        begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
        end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_M11:=
  Min (
  Min ((0-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))/(min_x_r1[jj]-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj])),(8000-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj])/(max_x_r1[jj]-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))),
  Min ((0-((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]))/(min_y_r1[jj]-
((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj])),(6000-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj])/(max_y_r1[jj]-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))))   );
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
           Pac1[ij].Position_X:=round((Pac1[ij].Position_X-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*ED_M11+((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]));
           Pac1[ij].Position_Y:=round((Pac1[ij].Position_Y-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*ED_M11+((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]));
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольне масштабування  ////////////////////////////
////////////////  Загальна рамка  {  ////////////////////////////
if(chb_ramka.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
end;
////////////////  }  Загальна рамка  ////////////////////////////
////////////////  Посимвольна рамка  {  ////////////////////////////
if(chb_ramka_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
```

```
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
end;
////////////////  }  Посимвольна рамка  ////////////////////////////
////////////////  Визначення КТ  {  ////////////////////////////
nnk:=-1; jj:=-1;
for j:=0 to i-1 do
begin
dx:=x-Pac1[j].Position_X; dy:=y-Pac1[j].Position_Y;
//////////  Визначення начальної або кінцевої КТ {
ekt:=false;
if ((nnk=-1) or (Pac1[j].N_simv<>jj)) then
begin
   jj:=jj+1;
   nin:=nnk+1;
   while (nin<=(i-1)) do
         begin
           if ((nin=0) or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin-1].N_simv<Pac1[j].N_simv))) then begin nn:=nin; nnkt:=kt; end;
           if ((nin=(i-1))or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin+1].N_simv>Pac1[j].N_simv))) then begin nnk:=nin; break; end;
           nin:=nin+1;
         end;
end;
if ((chb_k.Checked=true) and (Pac1[j].Pressure<>0) and ((j=0) or (j=i-1) or
(Pac1[j-1].Pressure=0) or (Pac1[j+1].Pressure=0)))  then
begin
if (nnkt<>kt) then
begin
  net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and ((Kontr_T[net].type_t=2) or (Kontr_T[net].type_t=3)))
then
    begin
       for ni:=net to kt-2 do
       begin
         Kontr_T[ni]:=Kontr_T[ni+1];
       end;
       kt:=kt-1;
    end
   else if ((net<>kt) and (Kontr_T[net].type_t=1)) then goto nt;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=1;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt: end;
////////////////  }  Визначення начальної або кінцевої КТ
////////////////  Визначення кутової КТ {  ////////////////////
  if ( (chb_r.Checked=true) and (ekt=false) and
  (Pac1[j].Pressure<>0) and (Pac1[j-1].Pressure<>0) and
(Pac1[j+1].Pressure<>0) and (Pac1[j].N_simv=Pac1[j-1].N_simv)  and
(Pac1[j].N_simv=Pac1[j+1].N_simv)
  and
(j<>(i-1)) and (j<>0) and ((Sign(Pac1[j].Position_X-
Pac1[j-1].Position_X)<>Sign(Pac1[j+1].Position_X-Pac1[j].Position_X)) or
 (Sign(Pac1[j].Position_Y-Pac1[j-1].Position_Y)<>
Sign(Pac1[j+1].Position_Y-Pac1[j].Position_Y))))
 then
  begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and (Kontr_T[net].type_t=3)) then
   begin
       for ni:=net to kt-2 do
       begin
         Kontr_T[ni]:=Kontr_T[ni+1];
       end;
       kt:=kt-1;
   end
   else if ((net<>kt) and ((Kontr_T[net].type_t=1) or
(Kontr_T[net].type_t=2))) then goto nt1;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=2;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt1: end;
////////////////  }  Визначення кутової КТ ////////////////
////////////////  Визначення КТ перетинання {  ///////////////
if ((chb_p.Checked=true) and (Pac1[j].Pressure<>0)) then
begin
if (ekt=false) then
begin
  for ni:=nn to (j-1) do
  begin
    if ( (Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y=Pac1[ni].Position_Y) and  (Pac1[ni].Pressure<>0) )  then
    begin break;   end;
  end;
  if(ni<>j) then
  begin
 if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
   if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
  end
  else
  begin
  for ni:=nn to (nnk-1) do
  begin
       if ((Pac1[ni].Pressure<>0) and (Pac1[ni+1].Pressure<>0)  and
(Pac1[j].ID<>Pac1[ni].ID) and (Pac1[j].ID<>Pac1[ni+1].ID))  then
       begin
         if(Pac1[ni].Position_X=Pac1[ni+1].Position_X) then
         begin
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
         end
         else if(Pac1[ni].Position_Y=Pac1[ni+1].Position_Y) then
         begin
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           if((Pac1[j].Position_Y=Pac1[ni].Position_Y) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1)) then
           begin break; end;
         end
```

```
          else
          begin
           x11:=Pac1[ni].Position_X;  x12:=Pac1[ni+1].Position_X;
           y11:=Pac1[ni].Position_Y;  y12:=Pac1[ni+1].Position_Y;
           if (x11=0) then
           begin
            b1:=y11;  a1:=(y12-b1)/x12;
           end
           else
           begin
             b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
           end;
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_Y=(a1*Pac1[j].Position_X+b1)) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
            begin break; end;
           end;
        end;
  end;
  if(ni<>nnk) then
  begin
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
    if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
  end;
  end;
  end;
  for ni:=nn+1 to (j-2) do
  begin
  if ((Pac1[ni].Pressure<>0) and (Pac1[ni-1].Pressure<>0) and
(Pac1[j-1].Pressure<>0) )  then
  begin
    x11:=Pac1[ni].Position_X;  x12:=Pac1[ni-1].Position_X;
    y11:=Pac1[ni].Position_Y;  y12:=Pac1[ni-1].Position_Y;
    x21:=Pac1[j].Position_X;  y21:=Pac1[j].Position_Y;
    if (j<>(nn)) then begin x22:=Pac1[j-1].Position_X; y22:=
Pac1[j-1].Position_Y end  else begin x22:=Pac1[j].Position_X;
y22:=Pac1[j].Position_Y; end;
    if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
    if (x11=0) then
    begin
      b1:=y11;  a1:=(y12-b1)/x12;
    end
    else
    begin
      b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
    end;
    if (x21=0) then
    begin
      b2:=y21;  a2:=(y22-b2)/x22;
    end
    else
    begin
      b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
    end;
    if (a1<>a2) then
      begin
        y0:=(b2*a1-b1*a2)/(a1-a2);  x0:=(y0-b1)/a1;
        minx1:=Min(x11,x12);  minx2:=Min(x21,x22); maxx1:=Max(x11,x12);
        maxx2:=Max(x21,x22); miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
        maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
        if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then
          begin goto enen1;   end;
      end;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      x0:=x11; y0:=a2*x0+b2;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then goto enen1;
    end
    else if ((x11<>x12) and (x21=x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12); a1:=(y11-b1)/x11;
      end;
      x0:=x21;  y0:=a1*x0+b1;
      miny1:=Min(y11,y12); miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then    goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11=y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      y0:=y11;  x0:=(y0-b2)/a2;
      minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then   goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
      end;
      y0:=y21; x0:=(y0-b1)/a1;
      minx1:=Min(x11,x12); minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12);  maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then      goto enen1;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      y0:=y21; x0:=x11;
      miny1:=Min(y11,y12); minx1:=Min(x21,x22);
      maxy1:=Max(y11,y12); maxx1:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1;
    end
    else if ((x11<>x12) and (x21=x22) and (y11=y12) and (y21<>y22)) then
    begin
      y0:=y11; x0:=x21;
      miny1:=Min(y21,y22); minx1:=Min(x11,x12);
```

```
    maxy1:=Max(y21,y22); maxx1:=Max(x11,x12);
    if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then     goto enen1;
   end;
   goto enen;
   enen1:
 if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=x0) and (Kontr_T[net].Position_Y=y0)) then
break;
   if (net<>kt) then begin goto enen; end;
end;
   jni3:=j;
jni4:   if ((sqrt(sqr(round(x0)-Pac1[jni3].Position_X)+sqr(round(y0)-
Pac1[jni3].Position_Y)))>=(sqrt(sqr(round(x0)-
Pac1[jni3-1].Position_X)+sqr(round(y0)-Pac1[jni3-1].Position_Y)))) then
   begin jni:=jni3; end else begin jni:=jni3-1; end;
   jni1:=1;
jni2:   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if (Kontr_T[net].ID=Pac1[jni].ID) then break;
   if ((net=kt) or (nnkt=kt)) then
   begin
     Kontr_T[kt].ID:=Pac1[jni].ID; Kontr_T[kt].Position_X:=round(x0);
Kontr_T[kt].Position_Y:=round(y0); if (jni3=j) then begin
Kontr_T[kt].Pressure:=round((Pac1[j].Pressure+Pac1[j-1].Pressure)/2); end
else if (jni3=ni) then begin
Kontr_T[kt].Pressure:=round((Pac1[ni].Pressure+Pac1[ni-1].Pressure)/2);
end; Kontr_T[kt].type_t:=3;  Kontr_T[kt].N_simv:=Pac1[jni].N_simv;
    kt:=kt+1; p:=p+1; goto enen;
   end;
   if (jni1=1) then begin jni1:=jni+1; if (jni=jni3) then begin jni:=jni-1; end
else if (jni=jni3-1) then begin jni:=jni+1; end; goto jni2; end
   else if((jni1=2) and (jni3=j)) then begin jni3:=ni; goto jni4; end else goto
enen;
enen:
  end;
  end;
end;
pr:=Pac1[j].Pressure;
end;
////////////////  }  Визначення КТ перетинання  //////////////
//////  Видалення кутових КТ (по дліні), які знаходяться поруч{
if(chb_bl.Checked=true) then
begin
  ut:=0; ut1:=0; ut2:=0; ij:=0; nvt:=1;
  uttt5: while (ut<=(kt-1)) do
         begin
           if (Kontr_T[ut].type_t=2) then break;
           ut:=ut+1;
         end;
  if(ut=kt) then goto ee;
if(nvt=1) then begin ut1:=ut; nvt:=2; ut:=ut+1; goto uttt5; end else ut2:=ut;
  if (Kontr_T[ut1].N_simv<>Kontr_T[ut2].N_simv) then begin ut1:=ut2;
nvt:=2; ut:=ut1+1; goto uttt5; end;
       ut12:=ut1;
       mu12: while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=Kontr_T[ut12].ID) then break;
       ij:=ij+1;
       end;
       if(ut12=ut1) then begin utt:=ij; utt1:=ij; ut12:=ut2; ij:=ij+1; goto mu12;
end else uttk:=ij;
  PR1:=0;
  while (utt1<uttk) do
  begin
     if(Pac1[utt1].Pressure=0) then break;
     PR1:=PR1+sqrt(sqr(Pac1[utt1].Position_X-Pac1[utt1+1].Position_X)+
sqr(Pac1[utt1].Position_Y-Pac1[utt1+1].Position_Y));
  utt1:=utt1+1;
  end;
    if(utt1<>uttk) then begin ut1:=ut2; nvt:=2; ut:=ut1+1; goto uttt5; end;
    if ((PR1<=StrToFloat(ed_bl.Text)) and (PR1>0)) then
    begin
     if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)<>
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-Pac1[utt-1].Position_Y)=
Sign(Pac1[utt+1].Position_Y-Pac1[utt].Position_Y)))) then
     begin
     if (Pac1[utt].Position_Y<>Pac1[utt-1].Position_Y) then
per11:=180*(arctan(abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)/abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)))/3.1415926535897932385 else per11:=90;
     if (Pac1[utt+1].Position_Y<>Pac1[utt].Position_Y) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)/abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))/3.1415926535897932385 else per12:=90;
     end;
     if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)=
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)<>Sign(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))) then
     begin
     if (Pac1[utt].Position_X<>Pac1[utt-1].Position_X) then
per11:=180*(arctan(abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)/abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)))/3.1415926535897932385 else per11:=90;
     if (Pac1[utt+1].Position_X<>Pac1[utt].Position_X) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)/abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)))/3.1415926535897932385 else per12:=90;
     end;
     per:=180-per11-per12;
     if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-1].Position_X)<>
Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-Pac1[uttk-1].Position_Y)=
Sign(Pac1[uttk+1].Position_Y-Pac1[uttk].Position_Y)))) then
     begin
     if (Pac1[uttk].Position_Y<>Pac1[uttk-1].Position_Y) then
per21:=180*(arctan(abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)/abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)))/3.1415926535897932385 else per21:=90;
     if (Pac1[uttk+1].Position_Y<>Pac1[uttk].Position_Y) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)/abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))/3.1415926535897932385 else per22:=90;
     end;
     if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-1].Position_X)=
Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)<>Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
     begin
     if (Pac1[uttk].Position_X<>Pac1[uttk-1].Position_X) then
per21:=180*(arctan(abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)/abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)))/3.1415926535897932385 else per21:=90;
     if (Pac1[uttk+1].Position_X<>Pac1[uttk].Position_X) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)/abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)))/3.1415926535897932385 else per22:=90;
     end;
     per1:=180-per21-per22;
     if (per<=per1) then
     begin
      for u:=ut2 to kt-2 do
      begin
        Kontr_T[u]:=Kontr_T[u+1];
      end;
     end
      else
      begin
       for u:=ut1 to kt-2 do
       begin
         Kontr_T[u]:=Kontr_T[u+1];
       end;
       ut1:=ut2-1;
      end;
       kt:=kt-1;
   end
   else begin ut1:=ut2; ut:=ut1+1; end;
   nvt:=2;     ij:=ij+1;
   goto uttt5;
ee: end;
/////  }  Видалення кутових КТ (по дліні), які знаходяться поруч
```

```
///// Визначення кількості КТ в кожному символі {
for ww:=0 to simv do
begin
  KKontr_T[ww]:=0;
end;
for w:=0 to kt-1 do
begin
KKontr_T[Kontr_T[w].N_simv]:=KKontr_T[Kontr_T[w].N_simv]+1;
end;
///// } Визначення кількості КТ в кожному символі
////////////////// } Визначення КТ ////////////////////////////
//////// Вивід результатів аналіза в таблиці { /////////////////
if (str_gr.Cells[kt+1,ii+1]='') then str_gr.Cells[kt+1,ii+1]:=IntToStr(0);
str_gr.Cells[kt+1,ii+1]:=IntToStr(StrToInt(str_gr.Cells[kt+1,ii+1])+1);
for ww:=0 to simv do
begin
if (str_gr_1.Cells[KKontr_T[ww]+1,ii+1]='') then
str_gr_1.Cells[KKontr_T[ww]+1,ii+1]:=IntToStr(0);
str_gr_1.Cells[KKontr_T[ww]+1,ii+1]:=
IntToStr(StrToInt(str_gr_1.Cells[KKontr_T[ww]+1,ii+1])+1);
end;
if (str_gr_t.Cells[i+1,ii+1]='') then str_gr_t.Cells[i+1,ii+1]:=IntToStr(0);
str_gr_t.Cells[i+1,ii+1]:=IntToStr(StrToInt(str_gr_t.Cells[i+1,ii+1])+1);
if (str_gr.Cells[2,ii+1]='') then str_gr.Cells[2,ii+1]:=IntToStr(0);
str_gr.Cells[2,ii+1]:=IntToStr(StrToInt(str_gr.Cells[2,ii+1])+1);
if (str_gr_1.Cells[2,ii+1]='') then str_gr_1.Cells[2,ii+1]:=IntToStr(0);
str_gr_1.Cells[2,ii+1]:=IntToStr(StrToInt(str_gr_1.Cells[2,ii+1])+1);
if (str_gr_kt.Cells[2,ii+1]='') then str_gr_kt.Cells[2,ii+1]:=IntToStr(0);
str_gr_kt.Cells[2,ii+1]:=IntToStr(StrToInt(str_gr_kt.Cells[2,ii+1])+1);
if (str_gr_t.Cells[2,ii+1]='') then str_gr_t.Cells[2,ii+1]:=IntToStr(0);
str_gr_t.Cells[2,ii+1]:=IntToStr(StrToInt(str_gr_t.Cells[2,ii+1])+1);
if (str_gr_kt.Cells[simv_s+1+2,ii+1]='') then
str_gr_kt.Cells[simv_s+1+2,ii+1]:=IntToStr(0);
str_gr_kt.Cells[simv_s+1+2,ii+1]:=
IntToStr(StrToInt(str_gr_kt.Cells[simv_s+1+2,ii+1])+1);
////////// } Вивід результатів аналіза в таблиці //////////////////
end;
endfn: iff:=FindNext(SR);
 end;
FindClose(SR);
end;
end;
procedure Tfm_tochki.DriveComboBox2Change(Sender: TObject);
begin
DirectoryListBox2.Drive:=DriveComboBox2.Drive;
end;
/////////////// Відображення на графіку даних з таблиці str_gr {
procedure Tfm_tochki.str_grClick(Sender: TObject);
var i:integer;
begin
Series1.Clear; Series2.Clear; Series3.Clear; Series4.Clear; Series5.Clear;
Series6.Clear; Series7.Clear; Series8.Clear; Series9.Clear; Series10.Clear;
Series11.Clear; Series12.Clear; Series13.Clear; Series14.Clear;
Series15.Clear;
for i:=Min(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) to
Max(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) do
begin
if (str_gr.Cells[0,1]<>'') then Series1.Active:=true;
With Series1 do
begin
Title:=str_gr.Cells[0,1];
if (str_gr.Cells[i+1,1]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,1]),'',clYellow) else
AddXY(i,0,'',clYellow);
end;
if (str_gr.Cells[0,2]<>'') then Series2.Active:=true;
With Series2 do
begin
Title:=str_gr.Cells[0,2];
if (str_gr.Cells[i+1,2]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,2]),'',clRed) else AddXY(i,0,'',clRed);
end;
if (str_gr.Cells[0,3]<>'') then Series3.Active:=true;
With Series3 do
begin
Title:=str_gr.Cells[0,3];
if (str_gr.Cells[i+1,3]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,3]),'',clGreen) else AddXY(i,0,'',clGreen);
end;
if (str_gr.Cells[0,4]<>'') then Series4.Active:=true;
With Series4 do
begin
Title:=str_gr.Cells[0,4];
if (str_gr.Cells[i+1,4]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,4]),'',clBlue) else AddXY(i,0,'',clBlue);
end;
if (str_gr.Cells[0,5]<>'') then Series5.Active:=true;
With Series5 do
begin
Title:=str_gr.Cells[0,5];
if (str_gr.Cells[i+1,5]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,5]),'',clLime) else AddXY(i,0,'',clLime);
end;
if (str_gr.Cells[0,6]<>'') then Series6.Active:=true;
With Series6 do
begin
Title:=str_gr.Cells[0,6];
if (str_gr.Cells[i+1,6]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,6]),'',clFuchsia) else
AddXY(i,0,'',clFuchsia);
end;
if (str_gr.Cells[0,7]<>'') then Series7.Active:=true;
With Series7 do
begin
Title:=str_gr.Cells[0,7];
if (str_gr.Cells[i+1,7]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,7]),'',clAqua) else AddXY(i,0,'',clAqua);
end;
if (str_gr.Cells[0,8]<>'') then Series8.Active:=true;
With Series8 do
begin
Title:=str_gr.Cells[0,8];
if (str_gr.Cells[i+1,8]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,8]),'',clSkyBlue) else
AddXY(i,0,'',clSkyBlue);
end;
if (str_gr.Cells[0,9]<>'') then Series9.Active:=true;
With Series9 do
begin
Title:=str_gr.Cells[0,9];
if (str_gr.Cells[i+1,9]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,9]),'',clOlive) else AddXY(i,0,'',clOlive);
end;
if (str_gr.Cells[0,10]<>'') then Series10.Active:=true;
With Series10 do
begin
Title:=str_gr.Cells[0,10];
if (str_gr.Cells[i+1,10]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,10]),'',clMaroon) else
AddXY(i,0,'',clMaroon);
end;
if (str_gr.Cells[0,11]<>'') then Series11.Active:=true;
With Series11 do
begin
Title:=str_gr.Cells[0,11];
if (str_gr.Cells[i+1,11]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,11]),'',clGray) else AddXY(i,0,'',clGray);
end;
if (str_gr.Cells[0,12]<>'') then Series12.Active:=true;
With Series12 do
begin
Title:=str_gr.Cells[0,12];
if (str_gr.Cells[i+1,12]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,12]),'',clMoneyGreen) else
AddXY(i,0,'',clMoneyGreen);
end;
if (str_gr.Cells[0,13]<>'') then Series13.Active:=true;
With Series13 do
begin
Title:=str_gr.Cells[0,13];
if (str_gr.Cells[i+1,13]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,13]),'',clBackground) else
AddXY(i,0,'',clBackground);
end;
if (str_gr.Cells[0,14]<>'') then Series14.Active:=true;
With Series14 do
begin
Title:=str_gr.Cells[0,14];
```

```
if (str_gr.Cells[i+1,14]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,14]),'',clGradientActiveCaption) else
AddXY(i,0,'',clGradientActiveCaption);
end;
if (str_gr.Cells[0,15]<>'')  then Series15.Active:=true;
With Series15 do
begin
Title:=str_gr.Cells[0,15];
if (str_gr.Cells[i+1,15]<>'') then
AddXY(i,StrToInt(str_gr.Cells[i+1,15]),'',clMenuBar) else
AddXY(i,0,'',clMenuBar);
end;
end;
end;
///////////////    }  Відображення на графіку даних з таблиці str_gr
///////////////      Відображення на графіку даних з таблиці str_gr_1  {
procedure Tfm_tochki.str_gr_1Click(Sender: TObject);
var i:integer;
begin
Series1.Clear; Series2.Clear; Series3.Clear; Series4.Clear; Series5.Clear;
Series6.Clear; Series7.Clear; Series8.Clear; Series9.Clear; Series10.Clear;
Series11.Clear; Series12.Clear; Series13.Clear; Series14.Clear;
Series15.Clear;
for i:=Min(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) to
Max(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) do
begin
With Series1 do
begin
Title:=str_gr_1.Cells[0,1];
if (str_gr_1.Cells[i+1,1]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,1]),'',clYellow) else
AddXY(i,0,'',clYellow);
end;
With Series2 do
begin
Title:=str_gr_1.Cells[0,2];
if (str_gr_1.Cells[i+1,2]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,2]),'',clRed) else AddXY(i,0,'',clRed);
end;
With Series3 do
begin
Title:=str_gr_1.Cells[0,3];
if (str_gr_1.Cells[i+1,3]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,3]),'',clGreen) else
AddXY(i,0,'',clGreen);
end;
With Series4 do
begin
Title:=str_gr_1.Cells[0,4];
if (str_gr_1.Cells[i+1,4]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,4]),'',clBlue) else AddXY(i,0,'',clBlue);
end;
With Series5 do
begin
Title:=str_gr_1.Cells[0,5];
if (str_gr_1.Cells[i+1,5]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,5]),'',clLime) else AddXY(i,0,'',clLime);
end;
With Series6 do
begin
Title:=str_gr_1.Cells[0,6];
if (str_gr_1.Cells[i+1,6]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,6]),'',clFuchsia) else
AddXY(i,0,'',clFuchsia);
end;
With Series7 do
begin
Title:=str_gr_1.Cells[0,7];
if (str_gr_1.Cells[i+1,7]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,7]),'',clAqua) else
AddXY(i,0,'',clAqua);
end;
With Series8 do
begin
Title:=str_gr_1.Cells[0,8];
if (str_gr_1.Cells[i+1,8]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,8]),'',clSkyBlue) else
AddXY(i,0,'',clSkyBlue);
end;
With Series9 do
begin
Title:=str_gr_1.Cells[0,9];
if (str_gr_1.Cells[i+1,9]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,9]),'',clOlive) else
AddXY(i,0,'',clOlive);
end;
With Series10 do
begin
Title:=str_gr_1.Cells[0,10];
if (str_gr_1.Cells[i+1,10]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,10]),'',clMaroon) else
AddXY(i,0,'',clMaroon);
end;
With Series11 do
begin
Title:=str_gr_1.Cells[0,11];
if (str_gr_1.Cells[i+1,11]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,11]),'',clGray) else
AddXY(i,0,'',clGray);
end;
With Series12 do
begin
Title:=str_gr_1.Cells[0,12];
if (str_gr_1.Cells[i+1,12]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,12]),'',clMoneyGreen) else
AddXY(i,0,'',clMoneyGreen);
end;
With Series13 do
begin
Title:=str_gr_1.Cells[0,13];
if (str_gr_1.Cells[i+1,13]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,13]),'',clBackground) else
AddXY(i,0,'',clBackground);
end;
With Series14 do
begin
Title:=str_gr_1.Cells[0,14];
if (str_gr_1.Cells[i+1,14]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,14]),'',clGradientActiveCaption) else
AddXY(i,0,'',clGradientActiveCaption);
end;
With Series15 do
begin
Title:=str_gr_1.Cells[0,15];
if (str_gr_1.Cells[i+1,15]<>'') then
AddXY(i,StrToInt(str_gr_1.Cells[i+1,15]),'',clMenuBar) else
AddXY(i,0,'',clMenuBar);
end;
end;
end;
///////////////   }  Відображення на графіку даних з таблиці  str_gr_1
procedure Tfm_tochki.ed_minKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9'])  then Key:=#0;
end;
procedure Tfm_tochki.ed_maxKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9'])  then Key:=#0;
end;
procedure Tfm_tochki.FormCreate(Sender: TObject);
begin
ed_min.MaxLength:=3; ed_max.MaxLength:=4;
end;
///////////////   Відображення на графіку даних з таблиці str_gr_kt  {
procedure Tfm_tochki.str_gr_ktClick(Sender: TObject);
var i:integer;
begin
Series1.Clear; Series2.Clear; Series3.Clear; Series4.Clear; Series5.Clear;
Series6.Clear; Series7.Clear; Series8.Clear; Series9.Clear; Series10.Clear;
Series11.Clear; Series12.Clear; Series13.Clear; Series14.Clear;
Series15.Clear;
for i:=Min(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) to
Max(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) do
begin
With Series1 do
begin
Title:=str_gr_kt.Cells[0,1];
if (str_gr_kt.Cells[i+2,1]<>'') then
```

```
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,1]),",clYellow) else
AddXY(i,0,",clYellow);
end;
With Series2 do
begin
Title:=str_gr_kt.Cells[0,2];
if (str_gr_kt.Cells[i+2,2]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,2]),",clRed) else AddXY(i,0,",clRed);
end;
With Series3 do
begin
Title:=str_gr_kt.Cells[0,3];
if (str_gr_kt.Cells[i+2,3]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,3]),",clGreen) else
AddXY(i,0,",clGreen);
end;
With Series4 do
begin
Title:=str_gr_kt.Cells[0,4];
if (str_gr_kt.Cells[i+2,4]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,4]),",clBlue) else AddXY(i,0,",clBlue);
end;
With Series5 do
begin
Title:=str_gr_kt.Cells[0,5];
if (str_gr_kt.Cells[i+2,5]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,5]),",clLime) else
AddXY(i,0,",clLime);
end;
With Series6 do
begin
Title:=str_gr_kt.Cells[0,6];
if (str_gr_kt.Cells[i+2,6]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,6]),",clFuchsia) else
AddXY(i,0,",clFuchsia);
end;
With Series7 do
begin
Title:=str_gr_kt.Cells[0,7];
if (str_gr_kt.Cells[i+2,7]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,7]),",clAqua) else
AddXY(i,0,",clAqua);
end;
With Series8 do
begin
Title:=str_gr_kt.Cells[0,8];
if (str_gr_kt.Cells[i+2,8]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,8]),",clSkyBlue) else
AddXY(i,0,",clSkyBlue);
end;
With Series9 do
begin
Title:=str_gr_kt.Cells[0,9];
if (str_gr_kt.Cells[i+2,9]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,9]),",clOlive) else
AddXY(i,0,",clOlive);
end;
With Series10 do
begin
Title:=str_gr_kt.Cells[0,10];
if (str_gr_kt.Cells[i+2,10]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,10]),",clMaroon) else
AddXY(i,0,",clMaroon);
end;
With Series11 do
begin
Title:=str_gr_kt.Cells[0,11];
if (str_gr_kt.Cells[i+2,11]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,11]),",clGray) else
AddXY(i,0,",clGray);
end;
With Series12 do
begin
Title:=str_gr_kt.Cells[0,12];
if (str_gr_kt.Cells[i+2,12]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,12]),",clMoneyGreen) else
AddXY(i,0,",clMoneyGreen);
end;
With Series13 do
begin
Title:=str_gr_kt.Cells[0,13];
if (str_gr_kt.Cells[i+2,13]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,13]),",clBackground) else
AddXY(i,0,",clBackground);
end;
With Series14 do
begin
Title:=str_gr_kt.Cells[0,14];
if (str_gr_kt.Cells[i+2,14]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,14]),",clGradientActiveCaption) else
AddXY(i,0,",clGradientActiveCaption);
end;
With Series15 do
begin
Title:=str_gr_kt.Cells[0,15];
if (str_gr_kt.Cells[i+2,15]<>'') then
AddXY(i,StrToInt(str_gr_kt.Cells[i+2,15]),",clMenuBar) else
AddXY(i,0,",clMenuBar);
end;
end;
end;
//////////////  }    Відображення на графіку даних з таблиці str_gr_kt
//////////////  Відображення на графіку даних з таблиці  str_gr_t  {
procedure Tfm_tochki.str_gr_tClick(Sender: TObject);
var i:integer;
begin
Series1.Clear; Series2.Clear; Series3.Clear; Series4.Clear; Series5.Clear;
Series6.Clear; Series7.Clear; Series8.Clear; Series9.Clear; Series10.Clear;
Series11.Clear; Series12.Clear; Series13.Clear; Series14.Clear;
Series15.Clear;
for i:=Min(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) to
Max(StrToInt(ed_min.Text),StrToInt(ed_max.Text)) do
begin
if (str_gr_t.Cells[0,1]<>'')  then Series1.Active:=true;
With Series1 do
begin
Title:=str_gr_t.Cells[0,1];
if (str_gr_t.Cells[i+1,1]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,1]),",clYellow) else
AddXY(i,0,",clYellow);
end;
if (str_gr_t.Cells[0,2]<>'')  then Series2.Active:=true;
With Series2 do
begin
Title:=str_gr_t.Cells[0,2];
if (str_gr_t.Cells[i+1,2]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,2]),",clRed) else AddXY(i,0,",clRed);
end;
if (str_gr_t.Cells[0,3]<>'')  then Series3.Active:=true;
With Series3 do
begin
Title:=str_gr_t.Cells[0,3];
if (str_gr_t.Cells[i+1,3]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,3]),",clGreen) else
AddXY(i,0,",clGreen);
end;
if (str_gr_t.Cells[0,4]<>'')  then Series4.Active:=true;
With Series4 do
begin
Title:=str_gr_t.Cells[0,4];
if (str_gr_t.Cells[i+1,4]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,4]),",clBlue) else AddXY(i,0,",clBlue);
end;
if (str_gr_t.Cells[0,5]<>'')  then Series5.Active:=true;
With Series5 do
begin
Title:=str_gr_t.Cells[0,5];
if (str_gr_t.Cells[i+1,5]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,5]),",clLime) else AddXY(i,0,",clLime);
end;
if (str_gr_t.Cells[0,6]<>'')  then Series6.Active:=true;
With Series6 do
begin
Title:=str_gr_t.Cells[0,6];
if (str_gr_t.Cells[i+1,6]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,6]),",clFuchsia) else
AddXY(i,0,",clFuchsia);
end;
```

```
if (str_gr_t.Cells[0,7]<>'')  then Series7.Active:=true;
With Series7 do
begin
Title:=str_gr_t.Cells[0,7];
if (str_gr_t.Cells[i+1,7]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,7]),'',clAqua) else AddXY(i,0,'',clAqua);
end;
if (str_gr_t.Cells[0,8]<>'')  then Series8.Active:=true;
With Series8 do
begin
Title:=str_gr_t.Cells[0,8];
if (str_gr_t.Cells[i+1,8]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,8]),'',clSkyBlue) else
AddXY(i,0,'',clSkyBlue);
end;
if (str_gr_t.Cells[0,9]<>'')  then Series9.Active:=true;
With Series9 do
begin
Title:=str_gr_t.Cells[0,9];
if (str_gr_t.Cells[i+1,9]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,9]),'',clOlive) else AddXY(i,0,'',clOlive);
end;
if (str_gr_t.Cells[0,10]<>'')  then Series10.Active:=true;
With Series10 do
begin
Title:=str_gr_t.Cells[0,10];
if (str_gr_t.Cells[i+1,10]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,10]),'',clMaroon) else
AddXY(i,0,'',clMaroon);
end;
if (str_gr_t.Cells[0,11]<>'')  then Series11.Active:=true;
With Series11 do
begin
Title:=str_gr_t.Cells[0,11];
if (str_gr_t.Cells[i+1,11]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,11]),'',clGray) else
AddXY(i,0,'',clGray);
end;
if (str_gr_t.Cells[0,12]<>'')  then Series12.Active:=true;
With Series12 do
begin
Title:=str_gr_t.Cells[0,12];
if (str_gr_t.Cells[i+1,12]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,12]),'',clMoneyGreen) else
AddXY(i,0,'',clMoneyGreen);
end;
if (str_gr_t.Cells[0,13]<>'')  then Series13.Active:=true;
With Series13 do
begin
Title:=str_gr_t.Cells[0,13];
if (str_gr_t.Cells[i+1,13]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,13]),'',clBackground) else
AddXY(i,0,'',clBackground);
end;
if (str_gr_t.Cells[0,14]<>'')  then Series14.Active:=true;
With Series14 do
begin
Title:=str_gr_t.Cells[0,14];
if (str_gr_t.Cells[i+1,14]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,14]),'',clGradientActiveCaption) else
AddXY(i,0,'',clGradientActiveCaption);
end;
if (str_gr_t.Cells[0,15]<>'')  then Series15.Active:=true;
With Series15 do
begin
Title:=str_gr_t.Cells[0,15];
if (str_gr_t.Cells[i+1,15]<>'') then
AddXY(i,StrToInt(str_gr_t.Cells[i+1,15]),'',clMenuBar) else
AddXY(i,0,'',clMenuBar);
end;
end;
end;
//////////////  }   Відображення на графіку даних з таблиці  str_gr_t
end.
```

Файл Unit6.pas:

```
unit Unit6;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DB, DBTables, FileCtrl;
type
  Tfm_obrab = class(TForm)
    bt_del: TButton;
    DirectoryListBox2: TDirectoryListBox;
    DriveComboBox2: TDriveComboBox;
    qr_del: TQuery;
    ds_qr_del: TDataSource;
    procedure bt_delClick(Sender: TObject);
    procedure DriveComboBox2Change(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_obrab: Tfm_obrab;
implementation
uses Unit1, Unit2, Unit3, Unit4, Unit5;
{$R *.dfm}
procedure Tfm_obrab.bt_delClick(Sender: TObject);
var ii,jj,iff,kol1,kolf:integer;
dr: array [0..32000] of AnsiString;
d,S1,pp:AnsiString;
SR: TSearchRec;
Label k1, e1;
begin
kol:=0;
ii:=0;
///////    Пошук наявних файлів в БДНЗ
d:=DirectoryListBox2.Directory;
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
  if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
 ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
FindClose(SR);
kol:=ii; kol1:=0;
k1: for jj:=kol1 to kol do
begin
d:=dr[jj];
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
 ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
FindClose(SR);
end;
kol1:=kol; kol:=ii-1;
if (kol1<>kol) then goto k1;
for ii:=0 to kol do
begin
kolf:=0;
if (Length(dr[ii])=3) then begin iff:=FindFirst(dr[ii] +'*.db',faAnyFile, SR);
end
else begin iff:=FindFirst(dr[ii] +'\*.db',faAnyFile, SR); end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr<>faDirectory)) then
```

```
 begin
if (Length(d)=3) then
begin pp:=dr[ii]+SR.Name; end
else begin pp:=dr[ii] +'\'+SR.Name; end;
///Видалення з БДНЗ записів, в котрих не вказано прізвище користувача
S1:='delete from "'+ pp + '" where Fam="";
 qr_del.SQL.Clear;  qr_del.SQL.Add(S1);  qr_del.ExecSQL();
  end;
 iff:=FindNext(SR);
 end;
FindClose(SR);
end;
end;
procedure Tfm_obrab.DriveComboBox2Change(Sender: TObject);
begin
DirectoryListBox2.Drive:=DriveComboBox2.Drive;
end;
procedure Tfm_obrab.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Action:=caFree;
end;
end.
```

Файл Unit7.pas:

```
unit Unit7;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, FileCtrl, StdCtrls, DB, DBTables,math, ExtCtrls;
type
  Tfm_raspozn = class(TForm)
    DriveComboBox2: TDriveComboBox;
    DirectoryListBox2: TDirectoryListBox;
    FileListBox2: TFileListBox;  DriveComboBox1: TDriveComboBox;
    DirectoryListBox1: TDirectoryListBox;
    bt_ok: TButton;     tb_raspozn: TTable;
    ds_tb_raspozn: TDataSource;     chb_k: TCheckBox; chb_bl: TCheckBox;
    chb_r: TCheckBox; chb_p: TCheckBox; chb_s: TCheckBox;
    chb_t: TCheckBox; ed_t: TEdit; chb_x: TCheckBox; ed_kol_t: TEdit;
    ed_x: TEdit; chb_povt: TCheckBox; ed_sdv_y: TEdit;
    chb_sdv_y: TCheckBox; ed_sdv_x: TEdit; chb_sdv_x: TCheckBox;
    ed_pov: TEdit; chb_pov: TCheckBox; ed_m: TEdit; chb_m: TCheckBox;
    chb_ramka_1: TCheckBox; chb_ramka: TCheckBox; ed_bl: TEdit;
    ed_pov_1: TEdit; chb_pov_1: TCheckBox;  chb_sdv_y_1: TCheckBox;
    ds_tb_main: TDataSource; tb_main: TTable;  tb_mainID: TIntegerField;
    tb_mainFam: TStringField; tb_mainImya: TStringField;
    tb_mainOtch: TStringField; tb_mainContext: TIntegerField;
    tb_mainStatus: TIntegerField; tb_mainCursor: TIntegerField;
    tb_mainButtons: TIntegerField; tb_mainPosition_X: TIntegerField;
    tb_mainPosition_Y: TIntegerField; tb_mainPressure: TIntegerField;
    tb_mainTangPress: TIntegerField;
    tb_mainOrientation_orAzimuth: TIntegerField;
    tb_mainOrientation_orAltitude: TIntegerField;
    tb_mainOrientation_orTwist: TIntegerField;
    tb_mainRotation_roPitch: TIntegerField;
    tb_mainRotation_roRoll: TIntegerField;
    tb_mainRotation_roYaw: TIntegerField; tb_mainData: TDateField;
    tb_mainTime: TTimeField; tb_mainTime_ms: TIntegerField;
    tb_raspoznPosition_X_0: TIntegerField;
    tb_raspoznPosition_Y_0: TIntegerField;
    tb_raspoznType_t_0: TIntegerField;
    tb_raspoznPosition_X_1: TIntegerField;
    tb_raspoznPosition_Y_1: TIntegerField;
    tb_raspoznType_t_1: TIntegerField;
    tb_raspoznPosition_X_2: TIntegerField;
    tb_raspoznPosition_Y_2: TIntegerField;
    tb_raspoznType_t_2: TIntegerField;
    tb_raspoznPosition_X_3: TIntegerField;
    tb_raspoznPosition_Y_3: TIntegerField;
    tb_raspoznType_t_3: TIntegerField;
    tb_raspoznPosition_X_4: TIntegerField;
    tb_raspoznPosition_Y_4: TIntegerField;
    tb_raspoznType_t_4: TIntegerField;
    tb_raspoznPosition_X_5: TIntegerField;
    tb_raspoznPosition_Y_5: TIntegerField;
    tb_raspoznType_t_5: TIntegerField;
    tb_raspoznPosition_X_6: TIntegerField;
    tb_raspoznPosition_Y_6: TIntegerField;
    tb_raspoznType_t_6: TIntegerField;
    tb_raspoznPosition_X_7: TIntegerField;
    tb_raspoznPosition_Y_7: TIntegerField;
    tb_raspoznType_t_7: TIntegerField;
    tb_raspoznPosition_X_8: TIntegerField;
    tb_raspoznPosition_Y_8: TIntegerField;
    tb_raspoznType_t_8: TIntegerField;
    tb_raspoznPosition_X_9: TIntegerField;
    tb_raspoznPosition_Y_9: TIntegerField;
    tb_raspoznType_t_9: TIntegerField;
    tb_raspoznPosition_X_10: TIntegerField;
    tb_raspoznPosition_Y_10: TIntegerField;
    tb_raspoznType_t_10: TIntegerField;
    tb_raspoznPosition_X_11: TIntegerField;
    tb_raspoznPosition_Y_11: TIntegerField;
    tb_raspoznType_t_11: TIntegerField;
    tb_raspoznPosition_X_12: TIntegerField;
    tb_raspoznPosition_Y_12: TIntegerField;
    tb_raspoznType_t_12: TIntegerField;
    tb_raspoznPosition_X_13: TIntegerField;
    tb_raspoznPosition_Y_13: TIntegerField;
    tb_raspoznType_t_13: TIntegerField;
    tb_raspoznPosition_X_14: TIntegerField;
    tb_raspoznPosition_Y_14: TIntegerField;
    tb_raspoznType_t_14: TIntegerField;
    tb_raspoznPosition_X_15: TIntegerField;
    tb_raspoznPosition_Y_15: TIntegerField;
    tb_raspoznType_t_15: TIntegerField;
    tb_raspoznPosition_X_16: TIntegerField;
    tb_raspoznPosition_Y_16: TIntegerField;
    tb_raspoznType_t_16: TIntegerField;
    tb_raspoznPosition_X_17: TIntegerField;
    tb_raspoznPosition_Y_17: TIntegerField;
    tb_raspoznType_t_17: TIntegerField;
    tb_raspoznPosition_X_18: TIntegerField;
    tb_raspoznPosition_Y_18: TIntegerField;
    tb_raspoznType_t_18: TIntegerField;
    tb_raspoznPosition_X_19: TIntegerField;
    tb_raspoznPosition_Y_19: TIntegerField;
    tb_raspoznType_t_19: TIntegerField;
    tb_raspoznPosition_X_20: TIntegerField;
    tb_raspoznPosition_Y_20: TIntegerField;
    tb_raspoznType_t_20: TIntegerField;
    tb_raspoznPosition_X_21: TIntegerField;
    tb_raspoznPosition_Y_21: TIntegerField;
    tb_raspoznType_t_21: TIntegerField;
    tb_raspoznPosition_X_22: TIntegerField;
    tb_raspoznPosition_Y_22: TIntegerField;
    tb_raspoznType_t_22: TIntegerField;
    tb_raspoznPosition_X_23: TIntegerField;
    tb_raspoznPosition_Y_23: TIntegerField;
    tb_raspoznType_t_23: TIntegerField;
    tb_raspoznPosition_X_24: TIntegerField;
    tb_raspoznPosition_Y_24: TIntegerField;
    tb_raspoznType_t_24: TIntegerField;
    tb_raspoznPosition_X_25: TIntegerField;
    tb_raspoznPosition_Y_25: TIntegerField;
    tb_raspoznType_t_25: TIntegerField;
    tb_raspoznPosition_X_26: TIntegerField;
    tb_raspoznPosition_Y_26: TIntegerField;
    tb_raspoznType_t_26: TIntegerField;
    tb_raspoznPosition_X_27: TIntegerField;
    tb_raspoznPosition_Y_27: TIntegerField;
    tb_raspoznType_t_27: TIntegerField;
    tb_raspoznPosition_X_28: TIntegerField;
    tb_raspoznPosition_Y_28: TIntegerField;
    tb_raspoznType_t_28: TIntegerField;
    tb_raspoznPosition_X_29: TIntegerField;
    tb_raspoznPosition_Y_29: TIntegerField;
    tb_raspoznType_t_29: TIntegerField;
    tb_raspoznID: TStringField; bt_ok_r: TButton; Edit1: TEdit;
    Edit2: TEdit; Edit3: TEdit; Edit5: TEdit; Edit4: TEdit;
    Label1: TLabel; Label2: TLabel; Label3: TLabel;  Label4: TLabel;
    Label5: TLabel; Label6: TLabel; Edit6: TEdit; Edit7: TEdit; Edit8: TEdit;
    Edit9: TEdit; Edit10: TEdit; Edit11: TEdit; Edit12: TEdit;
    RadioGroup1: TRadioGroup; rb_one: TRadioButton;
    rb_all: TRadioButton; Label7: TLabel; Label8: TLabel;
    chb_simv_m: TCheckBox; Edit13: TEdit; Edit14: TEdit; Edit15: TEdit;
    Edit16: TEdit; Edit17: TEdit; Edit18: TEdit; Edit19: TEdit; Edit20: TEdit;
```

```
    Edit21: TEdit; Label9: TLabel; Edit22: TEdit; Label10: TLabel;
    Label11: TLabel; Label12: TLabel;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DriveComboBox1Change(Sender: TObject);
    procedure DriveComboBox2Change(Sender: TObject);
    procedure DirectoryListBox2Change(Sender: TObject);
    procedure FileListBox2Change(Sender: TObject);
    procedure bt_okClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure ed_kol_tChange(Sender: TObject);
    procedure ed_kol_tKeyPress(Sender: TObject; var Key: Char);
    procedure ed_xChange(Sender: TObject);
    procedure ed_xKeyPress(Sender: TObject; var Key: Char);
    procedure ed_blKeyPress(Sender: TObject; var Key: Char);
    procedure ed_mKeyPress(Sender: TObject; var Key: Char);
    procedure ed_povKeyPress(Sender: TObject; var Key: Char);
    procedure ed_sdv_xKeyPress(Sender: TObject; var Key: Char);
    procedure ed_sdv_yKeyPress(Sender: TObject; var Key: Char);
    procedure ed_pov_1KeyPress(Sender: TObject; var Key: Char);
    procedure bt_ok_rClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  fm_raspozn: Tfm_raspozn;
  S:string;
  N_O:array [0..1000] of integer;
  N_O_N:array [0..1000] of real;
  N_O_K:array [0..1000] of integer;
  U_O:array [0..1000,0..3200,0..50] of integer;
  U_O_N:array [0..1000,0..3200,0..50] of real;
type dannie=record
ID:Integer; Fam:String; Imya:String; Otch:String; Context:Integer;
Status:Integer; Cursor:Integer; Buttons:Integer; Position_X:Integer;
Position_Y:Integer; Pressure:Integer; TangPress:Integer;
Orientation_orAzimuth:Integer; Orientation_orAltitude:Integer;
Orientation_orTwist:Integer; Rotation_roPitch:Integer;
Rotation_roRoll:Integer;  Rotation_roYaw:Integer; Data:TDateTime;
Time:TDateTime; Time_ms:Integer; N_simv:Integer;
end;
type kontr=record
ID:Integer; Fam:String; Imya:String; Otch:String; Context:Integer;
Status:Integer; Cursor:Integer; Buttons:Integer; Position_X:Integer;
Position_Y:Integer; Pressure:Integer; TangPress:Integer;
Orientation_orAzimuth:Integer; Orientation_orAltitude:Integer;
Orientation_orTwist:Integer; Rotation_roPitch:Integer;
Rotation_roRoll:Integer; Rotation_roYaw:Integer; Data:TDateTime;
Time:TDateTime;  Time_ms:Integer; type_t:integer; N_simv:Integer;
end;
var Pac1: array [0..32000] of dannie;
var Kontr_T: array [0..32000] of kontr;
var KKontr_T: array [0..32000] of integer;
nom_zap:integer;
implementation
uses Unit1, Unit2, Unit3, Unit4, Unit5, Unit6;
{$R *.dfm}
procedure Tfm_raspozn.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Action:=caFree;
end;
procedure Tfm_raspozn.DriveComboBox1Change(Sender: TObject);
begin
DirectoryListBox1.Drive:=DriveComboBox1.Drive;
end;
procedure Tfm_raspozn.DriveComboBox2Change(Sender: TObject);
begin
DirectoryListBox2.Drive:=DriveComboBox2.Drive;
end;
procedure Tfm_raspozn.DirectoryListBox2Change(Sender: TObject);
begin
if (rb_all.Checked=true) then begin bt_ok_r.Enabled:=true;
FileListBox2.Enabled:=false end
else
begin
FileListBox2.Enabled:=true;
FileListBox2.Directory:=DirectoryListBox2.Directory;
end;
end;
procedure Tfm_raspozn.FileListBox2Change(Sender: TObject);
begin
if (FileListBox2.FileName<>'') then bt_ok_r.Enabled:=true else
bt_ok_r.Enabled:=false;
end;
procedure Tfm_raspozn.bt_okClick(Sender: TObject);
var path:string;
iff, f,
pr,x,y,dx1,dx2,dy1,dy2,dx,dy,kt,i,ni,p,nii,j,j1,kz,w,nol,u,tt,ut,ut1:integer;
min_x_r, max_x_r, min_y_r, max_y_r, simv, min_rr, nmin_rr, jj, XS, YS, nn,
MM, XL, YL, XR, YR, ij, ww, sww,ut2,utt:integer;
jjj, kol1, kolf, ii:integer;
d:AnsiString;
dr: array [0..32000] of AnsiString;
min_x_r1: array [0..100] of integer;
max_x_r1: array [0..100] of integer;
min_y_r1: array [0..100] of integer;
max_y_r1: array [0..100] of integer;
trr: array [0..100] of integer;
kx: array [0..9] of integer;
ky: array [0..9] of integer;
kp: array [0..9] of integer;
ss,ss1,ss2,ss3,ss4:Extended;
sss, ekt:boolean;
x11,x12,x21,x22,x0,y11,y12,y21,y22,y0,a1,a2,b1,b2,minx1,minx2,maxx1,
maxx2,miny1,miny2,maxy1,maxy2,tt1,per,per1,per11,per12,per21,per22,
sssss, PR2, PR1:real;
ED_X1, ED_Y1, ED_X11, ED_Y11, net, ij1, jni, jni1, jni3, nnk, nin,nnkt,nvt,
ut12, uttk, utt1, jn:integer;
ED_M1, ED_P1, ED_M11:real;
SR, SR1: TSearchRec;
Label xv, xv1, rrr, rrr_1, rrr_2, rrr_3, k1, xv_, xv1_, rrr_3_, endf, uttt, utttt,
enen, enen1, slt1, jni2_, jni4_, nt_, nt1_, enen_, enen1_, ee1,uttt51,mu121;
begin
///////   Поиск учебных данных
iff:=FindFirst(S+'\Data_bukvy\i\i_*.db',faAnyFile, SR);
while iff=0 do
 begin
 DeleteFile(S+'\Data_bukvy\i\'+SR.Name);  iff := FindNext(SR);
 end;
FindClose(SR); iff:=FindFirst(S+'\Data_bukvy\z\z_*.db',faAnyFile, SR);
while iff=0 do
 begin
 DeleteFile(S+'\Data_bukvy\z\'+SR.Name);  iff := FindNext(SR);
 end;
FindClose(SR); iff:=FindFirst(S+'\Data_bukvy\a\a_*.db',faAnyFile, SR);
while iff=0 do
 begin
 DeleteFile(S+'\Data_bukvy\a\'+SR.Name);  iff := FindNext(SR);
 end;
FindClose(SR); iff:=FindFirst(S+'\Data_bukvy\c\c_*.db',faAnyFile, SR);
while iff=0 do
 begin
 DeleteFile(S+'\Data_bukvy\c\'+SR.Name);  iff := FindNext(SR);
 end;
FindClose(SR); iff:=FindFirst(S+'\Data_bukvy\ya\ya_*.db',faAnyFile, SR);
while iff=0 do
 begin
 DeleteFile(S+'\Data_bukvy\ya\'+SR.Name);  iff := FindNext(SR);
 end;
FindClose(SR); ChDir(S+'\'); ChDir(S+'\'); kol:=0; ii:=0;
d:=DirectoryListBox1.Directory;
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
FindClose(SR); kol:=ii; kol1:=0;
```

```
k1: for jjj:=kol1 to kol do
begin
d:=dr[jjj];
if (Length(d)=3) then
begin iff:=FindFirst(d +'*.*',faDirectory, SR); dr[ii]:=d; end
else begin iff:=FindFirst(d +'\*.*',faDirectory, SR); dr[ii]:=d +'\'; end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr=faDirectory)) then
 begin
if (Length(d)=3) then
begin dr[ii]:=d+SR.Name; end
else begin dr[ii]:=d +'\'+SR.Name; end;
 ii:=ii+1;
 end;
 iff:=FindNext(SR);
 end;
FindClose(SR);
end;
kol1:=kol; kol:=ii-1;
if (kol1<>kol) then goto k1;
for ii:=0 to kol do
begin
kolf:=0;
if (Length(dr[ii])=3) then begin iff:=FindFirst(dr[ii] +'*.db',faAnyFile, SR);
end
else begin iff:=FindFirst(dr[ii] +'\*.db',faAnyFile, SR); end;
while (iff=0) do
 begin
if ((SR.Name<>'.') and (SR.Name<>'..') and (SR.Attr<>faDirectory)) then
 begin
 kolf:=kolf+1; pr:=0; x:=0; y:=0; dx1:=0; dx2:=0; dy1:=0; dy2:=0; kt:=0;
i:=0; nom_zap:=0; p:=0; kz:=0;
if (Length(d)=3) then
begin tb_main.TableName:=dr[ii]+SR.Name; end
else begin tb_main.TableName:=dr[ii] +'\'+SR.Name; end;
Label8.Caption:=tb_main.TableName; Label8.Refresh(); tb_main.Open();
while tb_main.Eof<>true do
begin
////////////////// Створення масива точок, якщо виконувати згладжування {
if (chb_s.Checked=true)
then
begin
if (tb_mainPressure.AsInteger=0) then
  begin
  if (kz>9) then
    begin
    for j1:=kz-5 to kz-2 do
    begin
    Pac1[i].ID:=i;  Pac1[i].Position_X:=kx[j1];  Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1];  i:=i+1; nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end
  else if(kz>0) then
    begin
    for j1:=0 to kz-1 do
    begin
    Pac1[i].ID:=i;  Pac1[i].Position_X:=kx[j1]; Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1]; i:=i+1;  nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end;
  Pac1[i].ID:=tb_mainID.AsInteger;
  Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
  Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
  Pac1[i].Pressure:=tb_mainPressure.AsInteger;
  i:=i+1;   nom_zap:=nom_zap+1;
  end
else
if(kz<=4) then
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
i:=i+1; nom_zap:=nom_zap+1; kz:=kz+1;
end
else if ((kz>4) and (kz<9)) then
begin
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger; kz:=kz+1;
end
else if (kz>=9) then
begin
kx[9]:=tb_mainPosition_X.AsInteger;
ky[9]:=tb_mainPosition_Y.AsInteger;
kp[9]:=tb_mainPressure.AsInteger;
Pac1[i].Position_X:=0; Pac1[i].Position_Y:=0;
Pac1[i].Pressure:=0;
for j1:=0 to 9 do
  begin
  Pac1[i].Position_X:=Pac1[i].Position_X+kx[j1];
  Pac1[i].Position_Y:=Pac1[i].Position_Y+ky[j1];
  Pac1[i].Pressure:=Pac1[i].Pressure+kp[j1];
  if (j1<>0) then
    begin
    kx[j1-1]:=kx[j1];  ky[j1-1]:=ky[j1];  kp[j1-1]:=kp[j1];
    end;
  end;
Pac1[i].ID:=i;
Pac1[i].Position_X:=round(Pac1[i].Position_X/10);
Pac1[i].Position_Y:=round(Pac1[i].Position_Y/10);
Pac1[i].Pressure:=round(Pac1[i].Pressure/10);
i:=i+1; nom_zap:=nom_zap+1;
if (kz=9) then kz:=kz+1;
end;
end
////////////////// }  Створення масива точок, якщо виконувати згладжування
////////////////// Створення масива точок, якщо не виконувати згладжування
}
else
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
i:=i+1;
nom_zap:=nom_zap+1;
end;
////////////////// }  Створення масива точок, якщо не виконувати
згладжування
tb_main.Next();
end;
tb_main.Close();
////////////////// Видалення повторів {    ////////////////////////////
if(chb_povt.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].Position_X=Pac1[j+1].Position_X) and
(Pac1[j].Position_Y=Pac1[j+1].Position_Y) and (j<(i-1))) then
    begin
      u:=j+2;
      while ((Pac1[u].Pressure<>0) and
(Pac1[j].Position_X=Pac1[u].Position_X) and
(Pac1[j].Position_Y=Pac1[u].Position_Y) and (u<=(i-1))) do
      begin
        u:=u+1;
      end;
      nol:=u-1;
      begin
        i:=i-nol+j;
        nom_zap:=nom_zap-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
end;
////////////////// }  Видалення повторів   ////////////////////////////
```

```
//////////////// Видалення випадкових точок { ////////////////////////////
if(chb_t.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (((Pac1[j].Pressure=0) and (j<(i-1))) or (j=0)) then
    begin
      if(Pac1[j].Pressure=0) then jn:=j else jn:=j-1;
      slt1:nol:=jn;
      for u:=jn+1 to (jn+1+StrToInt(ed_t.Text)) do
       if (u=i) then begin nol:=u-1; break; end else if (Pac1[u].Pressure=0) then
nol:=u;
      if(nol<>jn) then
      begin
        i:=i-nol+jn;
        nom_zap:=nom_zap-nol+jn;
        for u:=jn+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-jn];
        end;
        if (jn<(i-1)) then goto slt1;
      end;
    end;
  end;
end;
//////////////// } Видалення випадкових точок ////////////////////////////
//////////////// Видалення хвостів за кількістю точок { ///////////////
tt:=StrToInt(ed_kol_t.Text);
if ((chb_x.Checked=true) and (StrToInt(ed_kol_t.Text)<>0)) then
begin
xv:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;
      for u:=j+1 to (j+1+tt) do
       if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
        else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then nol:=u;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
//////////////// } Видалення хвостів за кількістю точок ////////////////
//////////////// Видалення хвостів за длиною { ////////////////////////////
tt1:=StrToFloat(ed_x.Text);
if ((chb_x.Checked=true) and (StrToFloat(ed_x.Text)<>0)) then
begin
xv1:  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure=0) and (j<(i-3))) then
    begin
      nol:=j;   u:=j+1;    PR2:=0;
      while (PR2<=tt1) do
      begin
       if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
        else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then begin nol:=u;  end;
       u:=u+1;
       PR2:=PR2+sqrt(sqr(Pac1[u].Position_X-Pac1[u-1].Position_X)+
sqr(Pac1[u].Position_Y-Pac1[u-1].Position_Y));
      end;
      if(nol<>j) then
      begin
        i:=i-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
  nom_zap:=i;
end;
//////////////// } Видалення хвостів за длиною ////////////////////////////
//////////////// Визначення меж букв { ////////////////////////////
begin
  simv:=0;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r1[simv]:=Pac1[j].Position_X;
      max_x_r1[simv]:=Pac1[j].Position_X;
      min_y_r1[simv]:=Pac1[j].Position_Y;
      max_y_r1[simv]:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
       if (Pac1[j].Pressure<>0) then
       begin
         if (Pac1[j].Position_X<min_x_r1[simv]) then
min_x_r1[simv]:=Pac1[j].Position_X;
         if (Pac1[j].Position_X>max_x_r1[simv]) then
max_x_r1[simv]:=Pac1[j].Position_X;
         if (Pac1[j].Position_Y<min_y_r1[simv]) then
min_y_r1[simv]:=Pac1[j].Position_Y;
         if (Pac1[j].Position_Y>max_y_r1[simv]) then
max_y_r1[simv]:=Pac1[j].Position_Y;
       end
       else
       begin
         trr[simv]:=Pac1[j].ID;
         if((j<>0) and (j<>(i-1))) then
         begin
           simv:=simv+1; min_x_r1[simv]:=Pac1[j+1].Position_X;
           max_x_r1[simv]:=Pac1[j+1].Position_X;
           min_y_r1[simv]:=Pac1[j+1].Position_Y;
           max_y_r1[simv]:=Pac1[j+1].Position_Y;
         end;
       end;
  end;
if (simv<5) then goto endf;
rrr_3: if (simv>5) then
  begin
    min_rr:=min_x_r1[1]-max_x_r1[0];  nmin_rr:=1;
    for j:=1 to simv do
    begin
      if ((min_x_r1[j]-max_x_r1[j-1])<min_rr) then begin min_rr:=min_x_r1[j]-
max_x_r1[j-1]; nmin_rr:=j; end;
    end;
    max_x_r1[nmin_rr-1]:=Max(max_x_r1[nmin_rr-1],max_x_r1[nmin_rr]);
    min_x_r1[nmin_rr-1]:=Min(min_x_r1[nmin_rr-1],min_x_r1[nmin_rr]);
    max_y_r1[nmin_rr-1]:=Max(max_y_r1[nmin_rr-1],max_y_r1[nmin_rr]);
    min_y_r1[nmin_rr-1]:=Min(min_y_r1[nmin_rr-1],min_y_r1[nmin_rr]);
    for j:=nmin_rr to simv do
    begin
      max_x_r1[j]:=max_x_r1[j+1];  min_x_r1[j]:=min_x_r1[j+1];
      max_y_r1[j]:=max_y_r1[j+1];  min_y_r1[j]:=min_y_r1[j+1];
      trr[j-1]:=trr[j];
    end;
    simv:=simv-1;
    goto rrr_3;
  end;
end;
  nn:=Pac1[0].ID; ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
```

```
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    Pac1[ij].N_simv:=jj;
  end;
  nn:=trr[jj];
  end;
//////////////// } Визначення меж букв ////////////////////////////
//////////////// Зсув { ////////////////////////////
if((chb_sdv_x.Checked=true) or (chb_sdv_y.Checked=true)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X; max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y; max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then      min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then      max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then      min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then      max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  ED_X1:=StrToInt(ed_sdv_x.Text); ED_Y1:=StrToInt(ed_sdv_y.Text);
  if ((chb_sdv_x.Checked=true) and (ED_X1=777)) then
ED_X1:=round((8000-(max_x_r-min_x_r))/2)-min_x_r;
  if ((chb_sdv_y.Checked=true) and (ED_Y1=777)) then
ED_Y1:=round((6000-(max_y_r-min_y_r))/2)-min_y_r;
  for j:=0 to i-1 do
  begin
    if(chb_sdv_x.Checked=true) then
Pac1[j].Position_X:=Pac1[j].Position_X+ED_X1;
    if(chb_sdv_y.Checked=true) then
Pac1[j].Position_Y:=Pac1[j].Position_Y+ED_Y1;
  end;
end;
//////////////// } Зсув ////////////////////////////
//////////////// Масштабування { ////////////////////////////
if((chb_m.Checked=true) and (StrToFloat(ed_m.Text)>0)) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then      min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then      max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then      min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then      max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  ED_M1:=StrToFloat(ed_m.Text);
  if (ED_M1=777) then
  ED_M1:=Min (Min ((0-((max_x_r-min_x_r)/2+min_x_r))/(min_x_r-
((max_x_r-min_x_r)/2+min_x_r)),(8000-((max_x_r-min_x_r)/2+
min_x_r)/(max_x_r-((max_x_r-min_x_r)/2+min_x_r)))),
  Min ((0-((max_y_r-min_y_r)/2+min_y_r))/(min_y_r-((max_y_r-
min_y_r)/2+ min_y_r)),(6000-((max_y_r-min_y_r)/2+min_y_r)/(max_y_r-
((max_y_r-min_y_r)/2+min_y_r))))   );
  for j:=0 to i-1 do
  begin
    Pac1[j].Position_X:=round((Pac1[j].Position_X-((max_x_r-
min_x_r)/2+min_x_r))*ED_M1+((max_x_r-min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((Pac1[j].Position_Y-((max_y_r-
min_y_r)/2+min_y_r))*ED_M1+((max_y_r-min_y_r)/2+min_y_r));
  end;
end;
//////////////// } Масштабування ////////////////////////////
//////////////// Поворот { ////////////////////////////
if(chb_pov.Checked=true) then
begin
    for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then      min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then      max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then      min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then      max_y_r:=Pac1[j].Position_Y;
    end;
  end;
  for j:=0 to i-1 do
  begin
    XS:=Pac1[j].Position_X;  YS:=Pac1[j].Position_Y;
    Pac1[j].Position_X:=round((XS-((max_x_r-min_x_r)/2+min_x_r))*
cos(StrToFloat(ed_pov.Text)*3.1415926535897932385/180.0)+
(YS-((max_y_r-min_y_r)/2+min_y_r))* sin(StrToFloat(ed_pov.Text)*
PI/180.0))+round(((max_x_r-min_x_r)/2+ min_x_r));
    Pac1[j].Position_Y:=round((YS-((max_y_r-min_y_r)/2+min_y_r))*
cos(StrToFloat(ed_pov.Text)*3.1415926535897932385/180.0)-(XS-
((max_x_r-min_x_r)/2+min_x_r))*sin(StrToFloat(ed_pov.Text)*
PI/180.0))+round(((max_y_r-min_y_r)/2+min_y_r));
  end;
end;
//////////////// } Поворот ////////////////////////////
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
//////////////// Посимвольний поворот { ////////////////////////////
if(chb_pov_1.Checked=true) then
begin
ED_P1:=StrToFloat(ed_pov_1.Text);
if (ED_P1=777) then ED_P1:=
180.0*(arctan((YR-YL)/(XR-XL)))/3.1415926535897932385;
  nn:=Pac1[0].ID;
  ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      ij1:=ij;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
      while (ij<=(i-1)) do
      begin
      if (Pac1[ij].ID>=j) then break;
      ij:=ij+1;
      end;
      if (Pac1[ij].Pressure<>0) then
      begin
        if (Pac1[ij].Position_X<min_x_r1[jj]) then
```

```
min_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
        if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
        if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
j:=nn;
while (j<=trr[jj]-1) do
        begin
          ij:=ij1;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          XS:=Pac1[ij].Position_X;  YS:=Pac1[ij].Position_Y;
          Pac1[ij].Position_X:=round((XS-((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]))*cos(ED_P1* 3.1415926535897932385/180.0)+(YS-
((max_y_r1[jj]-min_y_r1[jj])/2+ min_y_r1[jj]))* sin(ED_P1*PI/180.0))
+round(((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]));
          Pac1[ij].Position_Y:=round((YS-((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]))*cos(ED_P1*3.1415926535897932385/180.0)-(XS-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))*sin(ED_P1*PI/180.0))+
round(((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]));
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольний поворот  ////////////////////////////
////////////////  Посимвольний зсув по Y  {  ////////////////////////////
if(chb_sdv_y_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
        ij:=0;
        while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=j) then break;
        ij:=ij+1;
        end;
        if (Pac1[ij].Pressure<>0) then
        begin
          if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
          if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
          if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
          if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
        end;
  end;
  nn:=trr[jj];
  end;
  MM:=max_y_r1[0];
  for jj:=0 to simv do
  begin
    MM:=Max(MM,max_y_r1[jj]);
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
j:=nn;
while (j<=trr[jj]-1) do
        begin
          ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          Pac1[ij].Position_Y:=Pac1[ij].Position_Y+(MM-max_y_r1[jj]);
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольний зсув по Y  ////////////////////////////
////////////////  Посимвольне масштабування{  ////////////////////////////
if(chb_simv_m.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
        ij:=0;
        while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=j) then break;
        ij:=ij+1;
        end;
        if (Pac1[ij].Pressure<>0) then
        begin
          if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
          if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
          if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
          if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
        end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_X11:=round((8000-(max_x_r1[jj]-min_x_r1[jj]))/2)-min_x_r1[jj];
  ED_Y11:=round((6000-(max_y_r1[jj]-min_y_r1[jj]))/2)-min_y_r1[jj];
j:=nn;
while (j<=trr[jj]-1) do
        begin
          ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          Pac1[ij].Position_X:=Pac1[ij].Position_X+ED_X11;
```

```
          Pac1[ij].Position_Y:=Pac1[ij].Position_Y+ED_Y11;
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
    nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_M11:=
  Min (
  Min ((0-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))/(min_x_r1[jj]-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj])),(8000-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj])/(max_x_r1[jj]-((max_x_r1[jj]-
min_x_r1[jj])/2+ min_x_r1[jj]))),Min ((0-((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]))/(min_y_r1[jj]-((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj])),
(6000-((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj])/(max_y_r1[jj]-
((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]))))   );
j:=nn;
while (j<=trr[jj]-1) do
       begin
        ij:=0;
         while (ij<=(i-1)) do
         begin
           if (Pac1[ij].ID>=j) then break;
           ij:=ij+1;
         end;
           Pac1[ij].Position_X:=round((Pac1[ij].Position_X-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*ED_M11+((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]));
           Pac1[ij].Position_Y:=round((Pac1[ij].Position_Y-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*ED_M11+((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]));
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
       end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольне масштабування  ////////////////////////////
////////////////  Загальна рамка  {  ////////////////////////////
if(chb_ramka.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
  end;
  for j:=0 to i-1 do
  begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
  end;
end;
////////////////  }  Загальна рамка  ////////////////////////////
////////////////  Посимвольна рамка  {  ////////////////////////////
if(chb_ramka_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
end;
////////////////  }  Посимвольна рамка  ////////////////////////////
////////////////  Визначення точок і КТ {  ////////////////
nnk:=-1; jj:=-1;
for j:=0 to i-1 do
begin
dx:=x-Pac1[j].Position_X;   dy:=y-Pac1[j].Position_Y;
/////  Визначення начальної або кінцевої КТ {
ekt:=false;
if ((nnk=-1) or (Pac1[j].N_simv<>jj)) then
begin
```

```
  jj:=jj+1;  nin:=nnk+1;
  while (nin<=(i-1)) do
        begin
         if ((nin=0) or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin-1].N_simv<Pac1[j].N_simv))) then begin nn:=nin; nnkt:=kt; end;
         if ((nin=(i-1))or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin+1].N_simv>Pac1[j].N_simv))) then begin nnk:=nin; break; end;
          nin:=nin+1;
        end;
end;
if ((chb_k.Checked=true) and (Pac1[j].Pressure<>0) and ((j=0) or (j=i-1) or
(Pac1[j-1].Pressure=0) or (Pac1[j+1].Pressure=0)))  then
begin
if (nnkt<>kt) then
begin
 net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and ((Kontr_T[net].type_t=2) or (Kontr_T[net].type_t=3)))
then
   begin
       for ni:=net to kt-2 do
      begin
        Kontr_T[ni]:=Kontr_T[ni+1];
      end;
      kt:=kt-1;
   end
   else if ((net<>kt) and (Kontr_T[net].type_t=1)) then goto nt_;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=1;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt_: end;
//////////  }  Визначення начальної або кінцевої КТ
////////////////  Визначення кутової КТ {
 if ( (chb_r.Checked=true) and (ekt=false) and   (Pac1[j].Pressure<>0) and
(Pac1[j-1].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].N_simv=Pac1[j-1].N_simv)  and
(Pac1[j].N_simv=Pac1[j+1].N_simv) and (j<>(i-1)) and (j<>0) and
((Sign(Pac1[j].Position_X-
Pac1[j-1].Position_X)<>Sign(Pac1[j+1].Position_X- Pac1[j].Position_X)) or
 (Sign(Pac1[j].Position_Y-Pac1[j-
1].Position_Y)<>Sign(Pac1[j+1].Position_Y-Pac1[j].Position_Y))))
then
 begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
   if ((net<>kt) and (Kontr_T[net].type_t=3)) then
   begin
       for ni:=net to kt-2 do
      begin
        Kontr_T[ni]:=Kontr_T[ni+1];
      end;
      kt:=kt-1;
   end
   else if ((net<>kt) and ((Kontr_T[net].type_t=1) or
(Kontr_T[net].type_t=2))) then goto nt1_;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=2;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt1_: end;
////////////////  }  Визначення кутової КТ
////////////////  Визначення КТ перетинання {  //////////////
if ((chb_p.Checked=true) and (Pac1[j].Pressure<>0)) then
begin
if (ekt=false) then
begin
 for ni:=nn to (j-1) do
 begin
   if ( (Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y=Pac1[ni].Position_Y) and  (Pac1[ni].Pressure<>0) )  then
   begin break;   end;
 end;
 if(ni<>j) then
 begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
   if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
 end
 else
 begin
 for ni:=nn to (nnk-1) do
 begin
      if ((Pac1[ni].Pressure<>0) and (Pac1[ni+1].Pressure<>0)  and
(Pac1[j].ID<>Pac1[ni].ID) and (Pac1[j].ID<>Pac1[ni+1].ID))  then
       begin
         if(Pac1[ni].Position_X=Pac1[ni+1].Position_X) then
         begin
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
         end
         else if(Pac1[ni].Position_Y=Pac1[ni+1].Position_Y) then
         begin
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           if((Pac1[j].Position_Y=Pac1[ni].Position_Y) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1)) then
           begin break; end;
         end
         else
         begin
          x11:=Pac1[ni].Position_X; x12:=Pac1[ni+1].Position_X;
          y11:=Pac1[ni].Position_Y; y12:=Pac1[ni+1].Position_Y;
          if (x11=0) then
          begin
           b1:=y11;  a1:=(y12-b1)/x12;
          end
          else
          begin
            b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
          end;
         minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
         maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
         miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
         maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
         if((Pac1[j].Position_Y=(a1*Pac1[j].Position_X+b1)) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
          begin break; end;
         end;
       end;
 end;
 if(ni<>nnk) then
 begin
if (nnkt<>kt) then
begin
   net:=round(sin(0));
   for net:=nnkt to kt-1 do
    if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
   if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
```

```
  end;
  end;
  end;
  for ni:=nn+1 to (j-2) do
  begin
  if ((Pac1[ni].Pressure<>0) and (Pac1[ni-1].Pressure<>0) and
(Pac1[j-1].Pressure<>0) )  then
  begin
    x11:=Pac1[ni].Position_X;  x12:=Pac1[ni-1].Position_X;
    y11:=Pac1[ni].Position_Y;  y12:=Pac1[ni-1].Position_Y;
    x21:=Pac1[j].Position_X;  y21:=Pac1[j].Position_Y;
    if (j<>(nn)) then begin x22:=Pac1[j-1].Position_X; y22:=
Pac1[j-1].Position_Y end  else begin x22:=Pac1[j].Position_X;
y22:=Pac1[j].Position_Y; end;
    if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
    if (x11=0) then
    begin
      b1:=y11; a1:=(y12-b1)/x12;
    end
    else
    begin
      b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
    end;
    if (x21=0) then
    begin
      b2:=y21;  a2:=(y22-b2)/x22;
    end
    else
    begin
      b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
    end;
    if (a1<>a2) then
      begin
        y0:=(b2*a1-b1*a2)/(a1-a2);  x0:=(y0-b1)/a1;
        minx1:=Min(x11,x12);  minx2:=Min(x21,x22);  maxx1:=Max(x11,x12);
        maxx2:=Max(x21,x22); miny1:=Min(y11,y12); miny2:=Min(y21,y22);
        maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
        if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then
          begin goto enen1_;   end;
      end;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22); a2:=(y21-b2)/x21;
      end;
      x0:=x11; y0:=a2*x0+b2;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then goto enen1_;
    end
    else if ((x11<>x12) and (x21=x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12); a1:=(y11-b1)/x11;
      end;
      x0:=x21; y0:=a1*x0+b1;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then    goto enen1_;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11=y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22); a2:=(y21-b2)/x21;
      end;
      y0:=y11; x0:=(y0-b2)/a2;
      minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then   goto enen1_;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12); a1:=(y11-b1)/x11;
      end;
      y0:=y21; x0:=(y0-b1)/a1;
      minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then      goto enen1_;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      y0:=y21; x0:=x11;
      miny1:=Min(y11,y12);  minx1:=Min(x21,x22);
      maxy1:=Max(y11,y12); maxx1:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1_;
    end
    else if ((x11<>x12) and (x21=x22) and (y11=y12) and (y21<>y22)) then
    begin
      y0:=y11; x0:=x21;
      miny1:=Min(y21,y22); minx1:=Min(x11,x12);
      maxy1:=Max(y21,y22); maxx1:=Max(x11,x12);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1_;
    end;
    goto enen_;
    enen1_:
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=x0) and (Kontr_T[net].Position_Y=y0)) then
break;
    if (net<>kt) then begin goto enen_; end;
end;
    jni3:=j;
jni4_:    if ((sqrt(sqr(round(x0)-Pac1[jni3].Position_X)+sqr(round(y0)-
Pac1[jni3].Position_Y)))>=(sqrt(sqr(round(x0)-Pac1[jni3-1].Position_X)+
sqr(round(y0)-Pac1[jni3-1].Position_Y)))) then
    begin jni:=jni3; end else begin jni:=jni3-1; end;
    jni1:=1;
jni2_:    net:=round(sin(0));
    for net:=nnkt to kt-1 do
      if (Kontr_T[net].ID=Pac1[jni].ID) then break;
    if ((net=kt) or (nnkt=kt)) then
    begin
      Kontr_T[kt].ID:=Pac1[jni].ID; Kontr_T[kt].Position_X:=round(x0);
Kontr_T[kt].Position_Y:=round(y0); if (jni3=j) then begin
Kontr_T[kt].Pressure:=round((Pac1[j].Pressure+Pac1[j-1].Pressure)/2); end
else if (jni3=ni) then begin
Kontr_T[kt].Pressure:=round((Pac1[ni].Pressure+Pac1[ni-1].Pressure)/2);
end; Kontr_T[kt].type_t:=3;  Kontr_T[kt].N_simv:=Pac1[jni].N_simv;
      kt:=kt+1; p:=p+1; goto enen_;
    end;
    if (jni1=1) then begin jni1:=jni+1; if (jni=jni3) then begin jni:=jni-1; end
else if (jni=jni3-1) then begin jni:=jni+1; end; goto jni2_; end
    else if((jni1=2) and (jni3=j)) then begin jni3:=ni; goto jni4_; end else goto
enen_;
```

```
enen_:
  end;
  end;
end;
pr:=Pac1[j].Pressure;
end;
/////////////////  }  Визначення КТ перетинання ///////////////
/////  Видалення кутових КТ (за дліною), які знаходяться поруч  {
if(chb_bl.Checked=true) then
begin
  ut:=0; ut1:=0; ut2:=0; ij:=0; nvt:=1;
  uttt51: while (ut<=(kt-1)) do
          begin
            if (Kontr_T[ut].type_t=2) then break;
            ut:=ut+1;
          end;
  if(ut=kt) then goto ee1;
if(nvt=1) then begin ut1:=ut; nvt:=2; ut:=ut+1; goto uttt51; end else ut2:=ut;
  if (Kontr_T[ut1].N_simv<>Kontr_T[ut2].N_simv) then begin ut1:=ut2;
nvt:=2; ut:=ut1+1; goto uttt51; end;
        ut12:=ut1;
        mu121: while (ij<=(i-1)) do
        begin
        if (Pac1[ij].ID>=Kontr_T[ut12].ID) then break;
        ij:=ij+1;
        end;
        if(ut12=ut1) then begin utt:=ij; utt1:=ij; ut12:=ut2; ij:=ij+1; goto mu121;
end else uttk:=ij;
  PR1:=0;
  while (utt1<uttk) do
  begin
     if(Pac1[utt1].Pressure=0) then break;
     PR1:=PR1+sqrt(sqr(Pac1[utt1].Position_X-Pac1[utt1+1].Position_X)+
sqr(Pac1[utt1].Position_Y-Pac1[utt1+1].Position_Y));
  utt1:=utt1+1;
  end;
    if(utt1<>uttk) then begin ut1:=ut2; nvt:=2; ut:=ut1+1; goto uttt51; end;
    if ((PR1<=StrToFloat(ed_bl.Text)) and (PR1>0)) then
    begin
      if ((Sign(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)<>Sign(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)) and ((Sign(Pac1[utt].Position_Y-Pac1[utt-
1].Position_Y)= Sign(Pac1[utt+1].Position_Y-Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_Y<>Pac1[utt-1].Position_Y) then
per11:=180*(arctan(abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)/abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_Y<>Pac1[utt].Position_Y) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)/abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))/3.1415926535897932385 else per12:=90;
      end;
      if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)=
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)<>Sign(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_X<>Pac1[utt-1].Position_X) then
per11:=180*(arctan(abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)/abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_X<>Pac1[utt].Position_X) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)/abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)))/3.1415926535897932385 else per12:=90;
      end;
      per:=180-per11-per12;
      if ((Sign(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)<>Sign(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)) and ((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)=Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_Y<>Pac1[uttk-1].Position_Y) then
per21:=180*(arctan(abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)/abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_Y<>Pac1[uttk].Position_Y) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)/abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))/3.1415926535897932385 else per22:=90;
      end;
      if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-
1].Position_X)=Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)<>Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_X<>Pac1[uttk-1].Position_X) then
per21:=180*(arctan(abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)/abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_X<>Pac1[uttk].Position_X) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)/abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)))/3.1415926535897932385 else per22:=90;
      end;
      per1:=180-per21-per22;
      if (per<=per1) then
      begin
        for u:=ut2 to kt-2 do
        begin
          Kontr_T[u]:=Kontr_T[u+1];
        end;
      end
       else
       begin
         for u:=ut1 to kt-2 do
         begin
           Kontr_T[u]:=Kontr_T[u+1];
         end;
         ut1:=ut2-1;
       end;
        kt:=kt-1;
    end
    else begin ut1:=ut2; ut:=ut1+1; end;
    nvt:=2; ij:=ij+1;
    goto uttt51;
ee1: end;
//  }  Видалення кутових КТ (за дліною), які знаходяться поруч
/////  Визначення кількості КТ в кожному символі  {
for ww:=0 to simv do
begin
  KKontr_T[ww]:=0;
end;
for w:=0 to kt-1 do
begin
KKontr_T[Kontr_T[w].N_simv]:=KKontr_T[Kontr_T[w].N_simv]+1;
end;
/////  }  Визначення кількості КТ в кожному символі
/////////////////  }  Визначення КТ ////////////////////////////
for ww:=0 to simv do
begin
if (KKontr_T[ww]>30) then goto endf;
end;
if (simv=5) then
begin
sww:=0;
for ww:=0 to simv do
begin
if ((ww=0) or (ww=4)) then path:='Data_bukvy\i\i_'
+IntToStr(KKontr_T[ww]) +'.db';
if (ww=1) then path:='Data_bukvy\z\z_' +IntToStr(KKontr_T[ww]) +'.db';
if (ww=2) then path:='Data_bukvy\a\a_' +IntToStr(KKontr_T[ww]) +'.db';
if (ww=3) then path:='Data_bukvy\c\c_' +IntToStr(KKontr_T[ww]) +'.db';
if (ww=5) then path:='Data_bukvy\ya\ya_' +IntToStr(KKontr_T[ww]) +'.db';
f:=FindFirst(path,faAnyFile, SR1);
if (f<>0) then
begin
  with tb_raspozn do
    begin
      TableName := S +'\' + path;
      CreateTable;
    end;
end;
  tb_raspozn.TableName:=S +'\' + path;
```

```
  tb_raspozn.Open();  tb_raspozn.Insert();
  tb_raspozn.Fields[0].AsString:=tb_main.TableName;
  for i:=0 to KKontr_T[ww]-1 do
  begin
    tb_raspozn.Fields[3*i+3-2].AsInteger:=Kontr_T[sww+i].Position_X ;
    tb_raspozn.Fields[3*i+3-1].AsInteger:=Kontr_T[sww+i].Position_Y;
    tb_raspozn.Fields[3*i+3].AsInteger:=Kontr_T[sww+i].type_t;
  end;
  tb_raspozn.Next(); tb_raspozn.Close(); FindClose(SR1);
sww:=sww+KKontr_T[ww];
end;
end;
end;
endf:
iff:=FindNext(SR);
 end;
FindClose(SR);
end;
end;
procedure Tfm_raspozn.FormCreate(Sender: TObject);
begin
GetDir(0, S); rb_one.Checked:=true;
end;
procedure Tfm_raspozn.ed_kol_tChange(Sender: TObject);
begin
if (StrToInt(ed_kol_t.Text)<>0) then ed_x.Text:=IntToStr(0);
end;
procedure Tfm_raspozn.ed_kol_tKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_xChange(Sender: TObject);
begin
if (StrToFloat(ed_x.Text)<>0) then ed_kol_t.Text:=IntToStr(0);
end;
procedure Tfm_raspozn.ed_xKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_blKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_mKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_povKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.','-'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_sdv_xKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','-'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_sdv_yKeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','-'])  then Key:=#0;
end;
procedure Tfm_raspozn.ed_pov_1KeyPress(Sender: TObject; var Key: Char);
begin
if not (Key in ['0','1','2','3','4','5','6','7','8','9','.','-'])  then Key:=#0;
end;
procedure Tfm_raspozn.bt_ok_rClick(Sender: TObject);
var path:string;
iff, iffn, f,
pr,x,y,dx1,dx2,dy1,dy2,dx,dy,kt,i,ni,p,nii,j,j1,kz,w,nol,u,tt,ut,ut1,ut2,utt, ut12,
uttk, utt1:integer;
min_x_r, max_x_r, min_y_r, max_y_r, simv, min_rr, nmin_rr, jj, XS, YS, nn,
MM, XL, YL, XR, YR, ij, ww, sww:integer;
jjj, jjjn, kol1, kolf, ii, iin, kol_b, ib, iu, nmaxv, koln, kol1n, kolfn:integer;
ssn:longint;
maxv:real;
ssn1:double;
d, dn, SB:AnsiString;
kuo: array [0..50] of integer;
dr: array [0..32000] of AnsiString;
drn: array [0..32000] of AnsiString;
min_x_r1: array [0..100] of integer;
max_x_r1: array [0..100] of integer;
min_y_r1: array [0..100] of integer;
max_y_r1: array [0..100] of integer;
trr: array [0..100] of integer;
kx: array [0..9] of integer;
ky: array [0..9] of integer;
kp: array [0..9] of integer;
ss,ss1,ss2, ss3, ss4:Extended;
sss, ekt:boolean;
x11,x12,x21,x22,x0,y11,y12,y21,y22,y0,a1,a2,b1,b2,minx1,minx2,maxx1,
maxx2,miny1,miny2,maxy1,maxy2,tt1,per,per1,per11,per12,per21,per22,
sssss, PR2, PR1:real;
SR, SR1, SRn: TSearchRec;
Ver: array [0..50] of real;
Bukvy: array [0..50] of string;
Bukvy1: array [0..50] of string;
VBukvy1: array [0..50,0..50] of integer;
ED_X1, ED_Y1, ED_X11, ED_Y11, net, ij1, nnk, nin, jni, jni1,
jni3,nnkt,nvt,jn : integer;
ED_M1, ED_P1, ED_M11:real;
SVB, SVP1:integer;
SVP: array [0..50] of integer;
Label xv, xv1, rrr, rrr_1, rrr_2, rrr_3, k1, k1n, xv_, xv1_, rrr_3_, endo, endfn,
one, uttt, utttt, enen, enen1, slt2, nt, nt1, jni2, jni4, ee, uttt5, mu12;
begin
kol_b:=5;
Bukvy[0]:='и'; Bukvy[1]:='э'; Bukvy[2]:='a'; Bukvy[3]:='ц'; Bukvy[4]:='и';
Bukvy[5]:='я';
Bukvy1[0]:='a'; Bukvy1[1]:='c'; Bukvy1[2]:='i'; Bukvy1[3]:='z';
Bukvy1[4]:='ya';
ChDir(S+'\');
for ww:=0 to 50 do
begin
for i:=0 to 50 do
begin
VBukvy1[ww][i]:=0;
end;
end;
for i:=0 to 32000 do
begin
Pac1[i].Position_X:=0; Pac1[i].Position_Y:=0;
end;
if (rb_one.Checked=true) then begin
tb_main.TableName:=FileListBox2.FileName; goto one; end;
koln:=0; iin:=0; dn:=DirectoryListBox2.Directory;
if (Length(dn)=3) then
begin iffn:=FindFirst(dn +'*.*',faDirectory, SRn); drn[iin]:=dn; end
else begin iffn:=FindFirst(dn +'\*.*',faDirectory, SRn); drn[iin]:=dn +'\'; end;
while (iffn=0) do
 begin
if ((SRn.Name<>'.') and (SRn.Name<>'..') and (SRn.Attr=faDirectory)) then
 begin
if (Length(dn)=3) then
begin drn[iin]:=dn+SRn.Name; end
else begin drn[iin]:=dn +'\'+SRn.Name; end;
 iin:=iin+1;
 end;
 iffn:=FindNext(SRn);
 end;
FindClose(SRn); koln:=iin; kol1n:=0;
k1n: for jjjn:=kol1n to koln do
begin
dn:=drn[jjjn];
if (Length(dn)=3) then
begin iffn:=FindFirst(dn +'*.*',faDirectory, SRn); drn[iin]:=dn; end
else begin iffn:=FindFirst(dn +'\*.*',faDirectory, SRn); drn[iin]:=dn +'\'; end;
while (iffn=0) do
 begin
if ((SRn.Name<>'.') and (SRn.Name<>'..') and (SRn.Attr=faDirectory)) then
 begin
if (Length(dn)=3) then
begin drn[iin]:=dn+SRn.Name; end
else begin drn[iin]:=dn +'\'+SRn.Name; end;
 iin:=iin+1;
 end;
 iffn:=FindNext(SRn);
 end;
FindClose(SRn);
end;
```

```
kol1n:=koln; koln:=iin-1;
if (kol1n<>koln) then goto k1n;
for iin:=0 to koln do
begin
kolfn:=0;
if (Length(drn[iin])=3) then begin iffn:=FindFirst(drn[iin] +'*.db',faAnyFile,
SRn); end
else begin iffn:=FindFirst(drn[iin] +'\*.db',faAnyFile, SRn); end;
while (iffn=0) do
 begin
if ((SRn.Name<>'.') and (SRn.Name<>'..') and (SRn.Attr<>faDirectory)) then
 begin
kolfn:=kolfn+1;
if (Length(dn)=3) then
begin tb_main.TableName:=drn[iin]+SRn.Name; end
else begin tb_main.TableName:=drn[iin] +'\'+SRn.Name; end;
one:
Label7.Caption:='Невідомий екземпляр - '+ tb_main.TableName;
Label7.Refresh(); Edit1.Text:=''; Edit2.Text:=''; Edit3.Text:=''; Edit4.Text:='';
Edit5.Text:=''; Edit6.Text:=''; Edit1.Refresh(); Edit2.Refresh();
Edit3.Refresh(); Edit4.Refresh(); Edit5.Refresh(); Edit6.Refresh();
for i:=0 to 1000  do
begin
N_O[i]:=0; N_O_K[i]:=0;
end;
pr:=0; x:=0; y:=0; dx1:=0; dx2:=0; dy1:=0; dy2:=0; kt:=0; i:=0; nom_zap:=0;
p:=0; kz:=0;
tb_main.Open();
while tb_main.Eof<>true do
begin
////////////////// Створення масива точок, якщо виконувати згладжування  {
if (chb_s.Checked=true)
then
begin
if (tb_mainPressure.AsInteger=0) then
  begin
  if (kz>9) then
    begin
    for j1:=kz-5 to kz-2 do
    begin
    Pac1[i].ID:=i;  Pac1[i].Position_X:=kx[j1]; Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1]; i:=i+1; nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end
  else if(kz>0) then
    begin
    for j1:=0 to kz-1 do
    begin
    Pac1[i].ID:=i; Pac1[i].Position_X:=kx[j1];  Pac1[i].Position_Y:=ky[j1];
    Pac1[i].Pressure:=kp[j1];  i:=i+1; nom_zap:=nom_zap+1;
    end;
    kz:=0;
    end;
  Pac1[i].ID:=tb_mainID.AsInteger;
  Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
  Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
  Pac1[i].Pressure:=tb_mainPressure.AsInteger;
  i:=i+1;  nom_zap:=nom_zap+1;
  end
else
if(kz<=4) then
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger;
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger;
i:=i+1; nom_zap:=nom_zap+1; kz:=kz+1;
end
else if ((kz>4) and (kz<9)) then
begin
kx[kz]:=tb_mainPosition_X.AsInteger;
ky[kz]:=tb_mainPosition_Y.AsInteger;
kp[kz]:=tb_mainPressure.AsInteger; kz:=kz+1;
end
else if (kz>=9) then
begin
kx[9]:=tb_mainPosition_X.AsInteger;
ky[9]:=tb_mainPosition_Y.AsInteger;
kp[9]:=tb_mainPressure.AsInteger;
Pac1[i].Position_X:=0; Pac1[i].Position_Y:=0; Pac1[i].Pressure:=0;
for j1:=0 to 9 do
  begin
  Pac1[i].Position_X:=Pac1[i].Position_X+kx[j1];
  Pac1[i].Position_Y:=Pac1[i].Position_Y+ky[j1];
  Pac1[i].Pressure:=Pac1[i].Pressure+kp[j1];
  if (j1<>0) then
    begin
    kx[j1-1]:=kx[j1];  ky[j1-1]:=ky[j1]; kp[j1-1]:=kp[j1];
    end;
  end;
Pac1[i].ID:=i; Pac1[i].Position_X:=round(Pac1[i].Position_X/10);
Pac1[i].Position_Y:=round(Pac1[i].Position_Y/10);
Pac1[i].Pressure:=round(Pac1[i].Pressure/10); i:=i+1; nom_zap:=nom_zap+1;
if (kz=9) then kz:=kz+1;
end;
end
////////////////// }  Створення масива точок, якщо виконувати згладжування
////////////////// Створення масива точок, якщо не виконувати згладжування
}
else
begin
Pac1[i].ID:=tb_mainID.AsInteger;
Pac1[i].Position_X:=tb_mainPosition_X.AsInteger;
Pac1[i].Position_Y:=tb_mainPosition_Y.AsInteger;
Pac1[i].Pressure:=tb_mainPressure.AsInteger; i:=i+1; nom_zap:=nom_zap+1;
end;
////////////  }  Створення масива точок, якщо не виконувати згладжування
tb_main.Next();
end;
tb_main.Close();
////////////////  Видалення повторів  {  ////////////////////////////
if(chb_povt.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if ((Pac1[j].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].Position_X=Pac1[j+1].Position_X) and
(Pac1[j].Position_Y=Pac1[j+1].Position_Y) and (j<(i-1))) then
    begin
      u:=j+2;
      while ((Pac1[u].Pressure<>0) and
(Pac1[j].Position_X=Pac1[u].Position_X) and
(Pac1[j].Position_Y=Pac1[u].Position_Y) and (u<=(i-1))) do
      begin
        u:=u+1;
      end;
      nol:=u-1;
      begin
        i:=i-nol+j;
        nom_zap:=nom_zap-nol+j;
        for u:=j+1 to i-1 do
        begin
          Pac1[u]:=Pac1[u+nol-j];
        end;
      end;
    end;
  end;
end;
////////////////  }  Видалення повторів  ////////////////////////////
////////////////  Видалення випадкових точок {  ////////////////////////////
if(chb_t.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
    if (((Pac1[j].Pressure=0) and (j<(i-1))) or (j=0)) then
    begin
      if(Pac1[j].Pressure=0) then jn:=j else jn:=j-1;
      slt2:nol:=jn;
      for u:=jn+1 to (jn+1+StrToInt(ed_t.Text)) do
        if (u=i) then begin nol:=u-1; break; end else if (Pac1[u].Pressure=0) then
nol:=u;
      if(nol<>jn) then
      begin
        i:=i-nol+jn;
```

```
      nom_zap:=nom_zap-nol+jn;
      for u:=jn+1 to i-1 do
      begin
        Pac1[u]:=Pac1[u+nol-jn];
      end;
      if (jn<(i-1)) then goto slt2;
     end;
    end;
  end;
end;
////////////////  }  Видалення випадкових точок  ////////////////////////////
////////////////  Видалення хвостів за кількістю точок {  ////////////////////
tt:=StrToInt(ed_kol_t.Text);
if ((chb_x.Checked=true) and (StrToInt(ed_kol_t.Text)<>0)) then
begin
xv:  for j:=0 to i-1 do
  begin
   if ((Pac1[j].Pressure=0) and (j<(i-3))) then
   begin
    nol:=j;
    for u:=j+1 to (j+1+tt) do
     if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
      else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then nol:=u;
    if(nol<>j) then
    begin
     i:=i-nol+j;
     for u:=j+1 to i-1 do
     begin
       Pac1[u]:=Pac1[u+nol-j];
     end;
    end;
   end;
  nom_zap:=i;
end;
////////////////  }  Видалення хвостів за кількістю точок  ////////////////////
////////////////  Видалення хвостів за дліною {  /////////////////////////////
tt1:=StrToFloat(ed_x.Text);
if ((chb_x.Checked=true) and (StrToFloat(ed_x.Text)<>0)) then
begin
xv1:  for j:=0 to i-1 do
  begin
   if ((Pac1[j].Pressure=0) and (j<(i-3))) then
   begin
    nol:=j; u:=j+1; PR2:=0;
    while (PR2<=tt1) do
    begin
     if ((u>(i-2)) or (Pac1[u].Pressure=0) or (Pac1[u+1].Pressure=0) or
(Pac1[u+2].Pressure=0)) then break
      else if ((Sign(Pac1[u+1].Position_X-
Pac1[u].Position_X)<>Sign(Pac1[u+2].Position_X-Pac1[u+1].Position_X)) or
(Sign(Pac1[u+1].Position_Y-
Pac1[u].Position_Y)<>Sign(Pac1[u+2].Position_Y-Pac1[u+1].Position_Y)))
then begin nol:=u;  end;
      u:=u+1;
     PR2:=PR2+sqrt(sqr(Pac1[u].Position_X-Pac1[u-1].Position_X)+
sqr(Pac1[u].Position_Y-Pac1[u-1].Position_Y));
    end;
    if(nol<>j) then
    begin
     i:=i-nol+j;
     for u:=j+1 to i-1 do
     begin
       Pac1[u]:=Pac1[u+nol-j];
     end;
    end;
   end;
  nom_zap:=i;
end;
////////////////  }  Видалення хвостів за дліною  /////////////////////////////
////////////////  Визначення меж букв {  //////////////////////////////
begin
 simv:=0;
 for j:=0 to i-1 do
 begin
  if (Pac1[j].Pressure<>0) then
  begin
   min_x_r1[simv]:=Pac1[j].Position_X;
   max_x_r1[simv]:=Pac1[j].Position_X;
   min_y_r1[simv]:=Pac1[j].Position_Y;
   max_y_r1[simv]:=Pac1[j].Position_Y;
   break;
  end;
 end;
 for j:=0 to i-1 do
 begin
     if (Pac1[j].Pressure<>0) then
     begin
      if (Pac1[j].Position_X<min_x_r1[simv]) then
min_x_r1[simv]:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r1[simv]) then
max_x_r1[simv]:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r1[simv]) then
min_y_r1[simv]:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r1[simv]) then
max_y_r1[simv]:=Pac1[j].Position_Y;
     end
     else
     begin
      trr[simv]:=Pac1[j].ID;
      if((j<>0) and (j<>(i-1))) then
      begin
       simv:=simv+1; min_x_r1[simv]:=Pac1[j+1].Position_X;
       max_x_r1[simv]:=Pac1[j+1].Position_X;
       min_y_r1[simv]:=Pac1[j+1].Position_Y;
       max_y_r1[simv]:=Pac1[j+1].Position_Y;
      end;
     end;
 end;
if (simv<5) then goto endfn;
rrr_3: if (simv>5) then
 begin
  min_rr:=min_x_r1[1]-max_x_r1[0];  nmin_rr:=1;
  for j:=1 to simv do
  begin
   if ((min_x_r1[j]-max_x_r1[j-1])<min_rr) then begin
    min_rr:=min_x_r1[j]-max_x_r1[j-1]; nmin_rr:=j; end;
  end;
  max_x_r1[nmin_rr-1]:=Max(max_x_r1[nmin_rr-1],max_x_r1[nmin_rr]);
  min_x_r1[nmin_rr-1]:=Min(min_x_r1[nmin_rr-1],min_x_r1[nmin_rr]);
  max_y_r1[nmin_rr-1]:=Max(max_y_r1[nmin_rr-1],max_y_r1[nmin_rr]);
  min_y_r1[nmin_rr-1]:=Min(min_y_r1[nmin_rr-1],min_y_r1[nmin_rr]);
  for j:=nmin_rr to simv do
  begin
   max_x_r1[j]:=max_x_r1[j+1];  min_x_r1[j]:=min_x_r1[j+1];
   max_y_r1[j]:=max_y_r1[j+1];  min_y_r1[j]:=min_y_r1[j+1];
   trr[j-1]:=trr[j];
  end;
  simv:=simv-1;
  goto rrr_3;
 end;
end;
 nn:=Pac1[0].ID;  ij:=0;
 for jj:=0 to simv do
 begin
 for j:=nn to trr[jj]-1 do
 begin
  while (ij<=(i-1)) do
  begin
  if (Pac1[ij].ID>=j) then break;
  ij:=ij+1;
  end;
  Pac1[ij].N_simv:=jj;
     end;
 nn:=trr[jj];
 end;
////////////////  }  Визначення меж букв  ////////////////////////////
////////////////  Зсув  {  ///////////////////////////
if((chb_sdv_x.Checked=true) or (chb_sdv_y.Checked=true)) then
begin
  for j:=0 to i-1 do
 begin
```

```
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
   end;
   for j:=0 to i-1 do
   begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
   end;
   ED_X1:=StrToInt(ed_sdv_x.Text);  ED_Y1:=StrToInt(ed_sdv_y.Text);
   if ((chb_sdv_x.Checked=true) and (ED_X1=777)) then
ED_X1:=round((8000-(max_x_r-min_x_r))/2)-min_x_r;
   if ((chb_sdv_y.Checked=true) and (ED_Y1=777)) then
ED_Y1:=round((6000-(max_y_r-min_y_r))/2)-min_y_r;
   for j:=0 to i-1 do
   begin
    if(chb_sdv_x.Checked=true) then
Pac1[j].Position_X:=Pac1[j].Position_X+ED_X1;
    if(chb_sdv_y.Checked=true) then
Pac1[j].Position_Y:=Pac1[j].Position_Y+ED_Y1;
   end;
end;
////////////////  }  Зсув  ////////////////////////////
////////////////  Масштабування  {  ////////////////////////////
if((chb_m.Checked=true) and (StrToFloat(ed_m.Text)>0)) then
begin
    for j:=0 to i-1 do
   begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
   end;
   for j:=0 to i-1 do
   begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
   end;
   ED_M1:=StrToFloat(ed_m.Text);
   if (ED_M1=777) then
   ED_M1:=Min (Min ((0-((max_x_r-min_x_r)/2+min_x_r))/(min_x_r-
((max_x_r-min_x_r)/2+min_x_r)),(8000-((max_x_r-min_x_r)/2+
min_x_r)/(max_x_r-((max_x_r-min_x_r)/2+min_x_r)))), Min ((0-((max_y_r-
min_y_r)/2+min_y_r))/(min_y_r-((max_y_r-min_y_r)/2+ min_y_r)),(6000-
((max_y_r-min_y_r)/2+min_y_r)/(max_y_r-((max_y_r-
min_y_r)/2+min_y_r))))   );
   for j:=0 to i-1 do
   begin
    Pac1[j].Position_X:=round((Pac1[j].Position_X-((max_x_r-min_x_r)/2+
min_x_r))*ED_M1+((max_x_r-min_x_r)/2+min_x_r));
    Pac1[j].Position_Y:=round((Pac1[j].Position_Y-((max_y_r-min_y_r)/2+
min_y_r))*ED_M1+((max_y_r-min_y_r)/2+min_y_r));
   end;
end;
////////////////  }  Масштабування  ////////////////////////////
////////////////  Поворот  {  ////////////////////////////
if(chb_pov.Checked=true) then
begin
    for j:=0 to i-1 do
   begin
    if (Pac1[j].Pressure<>0) then
    begin
      min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
      min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
      break;
    end;
   end;
   for j:=0 to i-1 do
   begin
    if (Pac1[j].Pressure<>0) then
    begin
      if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
      if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
      if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
    end;
   end;
   for j:=0 to i-1 do
   begin
    XS:=Pac1[j].Position_X; YS:=Pac1[j].Position_Y;
    Pac1[j].Position_X:=round((XS-((max_x_r-
min_x_r)/2+min_x_r))*cos(StrToFloat(ed_pov.Text)*
3.14159265358979323385/180.0)+(YS-((max_y_r-min_y_r)/2+min_y_r))*
sin(StrToFloat(ed_pov.Text)*PI/180.0))+round(((max_x_r-min_x_r)/2+
min_x_r));
    Pac1[j].Position_Y:=round((YS-((max_y_r-min_y_r)/2+min_y_r))*
cos(StrToFloat(ed_pov.Text)* 3.14159265358979323385/180.0)-(XS-
((max_x_r-min_x_r)/2+min_x_r))*sin(StrToFloat(ed_pov.Text)*
PI/180.0))+round(((max_y_r-min_y_r)/2+min_y_r));
   end;
end;
////////////////  }  Поворот  ////////////////////////////
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
////////////////  Посимвольний поворот  {  ////////////////////////////
Label8.Caption:='';
if(chb_pov_1.Checked=true) then
begin
ED_P1:=StrToFloat(ed_pov_1.Text);
if (ED_P1=777) then
ED_P1:=180.0*(arctan((YR-YL)/(XR-XL)))/3.14159265358979323385;
  nn:=Pac1[0].ID;
  ij:=0;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
   while (ij<=(i-1)) do
   begin
   if (Pac1[ij].ID>=j) then break;
   ij:=ij+1;
   end;
   if (Pac1[ij].Pressure<>0) then
   begin
     min_x_r1[jj]:=Pac1[ij].Position_X; max_x_r1[jj]:=Pac1[ij].Position_X;
     min_y_r1[jj]:=Pac1[ij].Position_Y; max_y_r1[jj]:=Pac1[ij].Position_Y;
     ij1:=ij;
     break;
   end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=ij1;
```

```
        while (ij<=(i-1)) do
        begin
          if (Pac1[ij].ID>=j) then break;
          ij:=ij+1;
        end;
        XS:=Pac1[ij].Position_X;  YS:=Pac1[ij].Position_Y;
        Pac1[ij].Position_X:=round((XS-((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]))*cos(ED_P1*3.1415926535897932385/180.0)+(YS-
((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]))*sin(ED_P1*PI/180.0))+
round(((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]));
        Pac1[ij].Position_Y:=round((YS-((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]))*cos(ED_P1* 3.1415926535897932385/180.0)-(XS-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))*sin(ED_P1*PI/180.0))+
round(((max_y_r1[jj]-min_y_r1[jj])/2+min_y_r1[jj]));
        if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
       end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольний поворот  ////////////////////////////
////////////////  Посимвольний зсув по Y  {  ////////////////////////////
if(chb_sdv_y_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X; max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y; max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  MM:=max_y_r1[0];
  for jj:=0 to simv do
  begin
    MM:=Max(MM,max_y_r1[jj]);
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          Pac1[ij].Position_Y:=Pac1[ij].Position_Y+(MM-max_y_r1[jj]);
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
end;
////////////////  }  Посимвольний зсув по Y  ////////////////////////////
////////////////  Посимвольне масштабування  {  ////////////////////////////
if(chb_simv_m.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
    if (Pac1[ij].ID>=j) then break;
    ij:=ij+1;
    end;
    if (Pac1[ij].Pressure<>0) then
    begin
      min_x_r1[jj]:=Pac1[ij].Position_X; max_x_r1[jj]:=Pac1[ij].Position_X;
      min_y_r1[jj]:=Pac1[ij].Position_Y; max_y_r1[jj]:=Pac1[ij].Position_Y;
      break;
    end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_X11:=round((8000-(max_x_r1[jj]-min_x_r1[jj]))/2)-min_x_r1[jj];
  ED_Y11:=round((6000-(max_y_r1[jj]-min_y_r1[jj]))/2)-min_y_r1[jj];
j:=nn;
while (j<=trr[jj]-1) do
        begin
         ij:=0;
          while (ij<=(i-1)) do
          begin
            if (Pac1[ij].ID>=j) then break;
            ij:=ij+1;
          end;
          Pac1[ij].Position_X:=Pac1[ij].Position_X+ED_X11;
          Pac1[ij].Position_Y:=Pac1[ij].Position_Y+ED_Y11;
          if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
        end;
  nn:=trr[jj];
  end;
    nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
    ij:=0;
    while (ij<=(i-1)) do
    begin
```

```
   if (Pac1[ij].ID>=j) then break;
   ij:=ij+1;
   end;
   if (Pac1[ij].Pressure<>0) then
   begin
     min_x_r1[jj]:=Pac1[ij].Position_X; max_x_r1[jj]:=Pac1[ij].Position_X;
     min_y_r1[jj]:=Pac1[ij].Position_Y; max_y_r1[jj]:=Pac1[ij].Position_Y;
     break;
   end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  ED_M11:=Min (Min ((0-((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]))/(min_x_r1[jj]-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj])),
(8000-((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj])/(max_x_r1[jj]-
((max_x_r1[jj]-min_x_r1[jj])/2+min_x_r1[jj]))), Min ((0-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))/(min_y_r1[jj]-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj])),(6000-((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj])/(max_y_r1[jj]-((max_y_r1[jj]-min_y_r1[jj])/2+ min_y_r1[jj]))))
);
j:=nn;
while (j<=trr[jj]-1) do
       begin
        ij:=0;
         while (ij<=(i-1)) do
         begin
           if (Pac1[ij].ID>=j) then break;
           ij:=ij+1;
         end;
           Pac1[ij].Position_X:=round((Pac1[ij].Position_X-((max_x_r1[jj]-
min_x_r1[jj])/2+min_x_r1[jj]))*ED_M11+((max_x_r1[jj]-min_x_r1[jj])/2+
min_x_r1[jj]));
           Pac1[ij].Position_Y:=round((Pac1[ij].Position_Y-((max_y_r1[jj]-
min_y_r1[jj])/2+min_y_r1[jj]))*ED_M11+((max_y_r1[jj]-min_y_r1[jj])/2+
min_y_r1[jj]));
           if (Pac1[ij].ID<>j) then j:=Pac1[ij].ID+1 else j:=j+1;
       end;
  nn:=trr[jj];
  end;
end;
//////////////// } Посимвольне масштабування ////////////////////////////
//////////////// Загальна рамка { ////////////////////////////
if(chb_ramka.Checked=true) then
begin
  for j:=0 to i-1 do
  begin
   if (Pac1[j].Pressure<>0) then
   begin
     min_x_r:=Pac1[j].Position_X;  max_x_r:=Pac1[j].Position_X;
     min_y_r:=Pac1[j].Position_Y;  max_y_r:=Pac1[j].Position_Y;
     break;
   end;
  end;
  for j:=0 to i-1 do
  begin
   if (Pac1[j].Pressure<>0) then
   begin
     if (Pac1[j].Position_X<min_x_r) then       min_x_r:=Pac1[j].Position_X;
     if (Pac1[j].Position_X>max_x_r) then       max_x_r:=Pac1[j].Position_X;
     if (Pac1[j].Position_Y<min_y_r) then       min_y_r:=Pac1[j].Position_Y;
     if (Pac1[j].Position_Y>max_y_r) then       max_y_r:=Pac1[j].Position_Y;
   end;
  end;
end;
//////////////// } Загальна рамка ////////////////////////////
//////////////// Посимвольна рамка { ////////////////////////////
if(chb_ramka_1.Checked=true) then
begin
  nn:=Pac1[0].ID;
  for jj:=0 to simv do
  begin
  for j:=nn to trr[jj]-1 do
  begin
   ij:=0;
   while (ij<=(i-1)) do
   begin
   if (Pac1[ij].ID>=j) then break;
   ij:=ij+1;
   end;
   if (Pac1[ij].Pressure<>0) then
   begin
     min_x_r1[jj]:=Pac1[ij].Position_X;  max_x_r1[jj]:=Pac1[ij].Position_X;
     min_y_r1[jj]:=Pac1[ij].Position_Y;  max_y_r1[jj]:=Pac1[ij].Position_Y;
     break;
   end;
  end;
  for j:=nn to trr[jj]-1 do
  begin
       ij:=0;
       while (ij<=(i-1)) do
       begin
       if (Pac1[ij].ID>=j) then break;
       ij:=ij+1;
       end;
       if (Pac1[ij].Pressure<>0) then
       begin
         if (Pac1[ij].Position_X<min_x_r1[jj]) then
min_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_X>max_x_r1[jj]) then
max_x_r1[jj]:=Pac1[ij].Position_X;
         if (Pac1[ij].Position_Y<min_y_r1[jj]) then
min_y_r1[jj]:=Pac1[ij].Position_Y;
         if (Pac1[ij].Position_Y>max_y_r1[jj]) then
max_y_r1[jj]:=Pac1[ij].Position_Y;
       end;
  end;
  nn:=trr[jj];
  end;
  XL:=max_x_r1[0]; YL:=max_y_r1[0];
  XR:=max_x_r1[simv]; YR:=max_y_r1[simv];
end;
//////////////// } Посимвольна рамка ////////////////////////////
//////////////// Визначення КТ { ////////////////////////////
nnk:=-1; jj:=-1;
for j:=0 to i-1 do
begin
dx:=x-Pac1[j].Position_X; dy:=y-Pac1[j].Position_Y;
//////////////// Визначення начальної або кінцевої КТ {
ekt:=false;
if ((nnk=-1) or (Pac1[j].N_simv<>jj)) then
begin
  jj:=jj+1;  nin:=nnk+1;
  while (nin<=(i-1)) do
        begin
          if ((nin=0) or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin-1].N_simv<Pac1[j].N_simv))) then begin nn:=nin; nnkt:=kt; end;
          if ((nin=(i-1))or ((Pac1[nin].N_simv=Pac1[j].N_simv) and
(Pac1[nin+1].N_simv>Pac1[j].N_simv))) then begin nnk:=nin; break; end;
          nin:=nin+1;
        end;
end;
if ((chb_k.Checked=true) and (Pac1[j].Pressure<>0) and ((j=0) or (j=i-1) or
(Pac1[j-1].Pressure=0) or (Pac1[j+1].Pressure=0)))  then
begin
if (nnkt<>kt) then
```

```
begin
  net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
    if ((net<>kt) and ((Kontr_T[net].type_t=2) or (Kontr_T[net].type_t=3)))
then
    begin
        for ni:=net to kt-2 do
       begin
         Kontr_T[ni]:=Kontr_T[ni+1];
       end;
       kt:=kt-1;
    end
    else if ((net<>kt) and (Kontr_T[net].type_t=1)) then goto nt;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=1;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt: end;
//////////  }  Визначення начальної або кінцевої КТ
/////////////////  Визначення кутової КТ {  /////////////////
  if ( (chb_r.Checked=true) and (ekt=false) and (Pac1[j].Pressure<>0) and
(Pac1[j-1].Pressure<>0) and (Pac1[j+1].Pressure<>0) and
(Pac1[j].N_simv=Pac1[j-1].N_simv)  and
(Pac1[j].N_simv=Pac1[j+1].N_simv) and (j<>(i-1)) and (j<>0) and
((Sign(Pac1[j].Position_X-
Pac1[j-1].Position_X)<>Sign(Pac1[j+1].Position_X-Pac1[j].Position_X)) or
 (Sign(Pac1[j].Position_Y-
Pac1[j-1].Position_Y)<>Sign(Pac1[j+1].Position_Y-Pac1[j].Position_Y))))
 then
  begin
if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
    if ((net<>kt) and (Kontr_T[net].type_t=3)) then
    begin
        for ni:=net to kt-2 do
       begin
         Kontr_T[ni]:=Kontr_T[ni+1];
       end;
       kt:=kt-1;
    end
    else if ((net<>kt) and ((Kontr_T[net].type_t=1) or
(Kontr_T[net].type_t=2))) then goto nt1;
end;
ekt:=true; Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=2;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1;
nt1: end;
/////////////////  }  Визначення кутової КТ  /////////////////
/////////////////  Визначення КТ перетинання  {  //////////
if ((chb_p.Checked=true) and (Pac1[j].Pressure<>0)) then
begin
if (ekt=false) then
begin
  for ni:=nn to (j-1) do
  begin
    if ( (Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y=Pac1[ni].Position_Y) and  (Pac1[ni].Pressure<>0) )  then
    begin break;   end;
  end;
  if(ni<>j) then
  begin
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
    if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
  end
  else
  begin
  for ni:=nn to (nnk-1) do
  begin
       if ((Pac1[ni].Pressure<>0) and (Pac1[ni+1].Pressure<>0)  and
(Pac1[j].ID<>Pac1[ni].ID) and (Pac1[j].ID<>Pac1[ni+1].ID))  then
       begin
         if(Pac1[ni].Position_X=Pac1[ni+1].Position_X) then
         begin
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_X=Pac1[ni].Position_X) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
         end
         else if(Pac1[ni].Position_Y=Pac1[ni+1].Position_Y) then
         begin
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           if((Pac1[j].Position_Y=Pac1[ni].Position_Y) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1)) then
           begin break; end;
         end
         else
         begin
           x11:=Pac1[ni].Position_X;  x12:=Pac1[ni+1].Position_X;
           y11:=Pac1[ni].Position_Y;  y12:=Pac1[ni+1].Position_Y;
           if (x11=0) then
           begin
            b1:=y11;  a1:=(y12-b1)/x12;
           end
           else
           begin
             b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
           end;
           minx1:=Min(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           maxx1:=Max(Pac1[ni].Position_X,Pac1[ni+1].Position_X);
           miny1:=Min(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           maxy1:=Max(Pac1[ni].Position_Y,Pac1[ni+1].Position_Y);
           if((Pac1[j].Position_Y=(a1*Pac1[j].Position_X+b1)) and
(Pac1[j].Position_X>minx1) and (Pac1[j].Position_X<maxx1) and
(Pac1[j].Position_Y>miny1) and (Pac1[j].Position_Y<maxy1)) then
           begin break; end;
         end;
       end;
  end;
  if(ni<>nnk) then
  begin
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=Pac1[j].Position_X) and
(Kontr_T[net].Position_Y=Pac1[j].Position_Y)) then break;
end;
    if ((net=kt) or (nnkt=kt)) then begin Kontr_T[kt].ID:=Pac1[j].ID;
Kontr_T[kt].Position_X:=Pac1[j].Position_X;
Kontr_T[kt].Position_Y:=Pac1[j].Position_Y;
Kontr_T[kt].Pressure:=Pac1[j].Pressure; Kontr_T[kt].type_t:=3;
Kontr_T[kt].N_simv:=Pac1[j].N_simv; kt:=kt+1; p:=p+1; end;
  end;
  end;
  end;
  for ni:=nn+1 to (j-2) do
  begin
  if ((Pac1[ni].Pressure<>0) and (Pac1[ni-1].Pressure<>0) and (Pac1[j-
1].Pressure<>0) )  then
  begin
    x11:=Pac1[ni].Position_X; x12:=Pac1[ni-1].Position_X;
    y11:=Pac1[ni].Position_Y; y12:=Pac1[ni-1].Position_Y;
    x21:=Pac1[j].Position_X;  y21:=Pac1[j].Position_Y;
    if (j<>(nn)) then begin x22:=Pac1[j-1].Position_X; y22:=
Pac1[j-1].Position_Y end  else begin x22:=Pac1[j].Position_X;
y22:=Pac1[j].Position_Y; end;
```

```
    if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
    if (x11=0) then
    begin
      b1:=y11;  a1:=(y12-b1)/x12;
    end
    else
    begin
      b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
    end;
    if (x21=0) then
    begin
      b2:=y21; a2:=(y22-b2)/x22;
    end
    else
    begin
      b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
    end;
    if (a1<>a2) then
      begin
        y0:=(b2*a1-b1*a2)/(a1-a2);  x0:=(y0-b1)/a1;
        minx1:=Min(x11,x12);  minx2:=Min(x21,x22);  maxx1:=Max(x11,x12);
        maxx2:=Max(x21,x22); miny1:=Min(y11,y12); miny2:=Min(y21,y22);
        maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
        if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then
          begin goto enen1;   end;
      end;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21; a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      x0:=x11; y0:=a2*x0+b2;
      miny1:=Min(y11,y12);  miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then goto enen1;
    end
    else if ((x11<>x12) and (x21=x22) and (y11<>y12) and (y21<>y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12);  a1:=(y11-b1)/x11;
      end;
      x0:=x21;  y0:=a1*x0+b1;
      miny1:=Min(y11,y12); miny2:=Min(y21,y22);
      maxy1:=Max(y11,y12); maxy2:=Max(y21,y22);
      if ((y0>(miny1+0.000001)) and (y0<(maxy1-0.000001)) and
(y0>(miny2+0.000001)) and (y0<(maxy2-0.000001))) then    goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11=y12) and (y21<>y22)) then
    begin
      if (x21=0) then
      begin
        b2:=y21;  a2:=(y22-b2)/x22;
      end
      else
      begin
        b2:=(x21*y22-x22*y21)/(x21-x22);  a2:=(y21-b2)/x21;
      end;
      y0:=y11;  x0:=(y0-b2)/a2;
      minx1:=Min(x11,x12);  minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then   goto enen1;
    end
    else if ((x11<>x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      if (x11=0) then
      begin
        b1:=y11; a1:=(y12-b1)/x12;
      end
      else
      begin
        b1:=(x11*y12-x12*y11)/(x11-x12); a1:=(y11-b1)/x11;
      end;
      y0:=y21; x0:=(y0-b1)/a1;
      minx1:=Min(x11,x12); minx2:=Min(x21,x22);
      maxx1:=Max(x11,x12); maxx2:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(x0>(minx2+0.000001)) and (x0<(maxx2-0.000001))) then      goto enen1;
    end
    else if ((x11=x12) and (x21<>x22) and (y11<>y12) and (y21=y22)) then
    begin
      y0:=y21; x0:=x11;
      miny1:=Min(y11,y12);  minx1:=Min(x21,x22);
      maxy1:=Max(y11,y12); maxx1:=Max(x21,x22);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1;
    end
    else if ((x11<>x12) and (x21=x22) and (y11=y12) and (y21<>y22)) then
    begin
      y0:=y11; x0:=x21;
      miny1:=Min(y21,y22); minx1:=Min(x11,x12);
      maxy1:=Max(y21,y22); maxx1:=Max(x11,x12);
      if ((x0>(minx1+0.000001)) and (x0<(maxx1-0.000001)) and
(y0>(miny1+0.000001)) and (y0<(maxy1-0.000001))) then      goto enen1;
    end;
    goto enen;
    enen1:
 if (nnkt<>kt) then
begin
    net:=round(sin(0));
    for net:=nnkt to kt-1 do
     if ((Kontr_T[net].Position_X=x0) and (Kontr_T[net].Position_Y=y0)) then
break;
    if (net<>kt) then begin goto enen; end;
end;
    jni3:=j;
jni4:    if ((sqrt(sqr(round(x0)-Pac1[jni3].Position_X)+sqr(round(y0)-
Pac1[jni3].Position_Y)))>=(sqrt(sqr(round(x0)-Pac1[jni3-1].Position_X)+
sqr(round(y0)-Pac1[jni3-1].Position_Y)))) then
    begin jni:=jni3; end else begin jni:=jni3-1; end;
    jni1:=1;
jni2:    net:=round(sin(0));
    for net:=nnkt to kt-1 do
      if (Kontr_T[net].ID=Pac1[jni].ID) then break;
    if ((net=kt) or (nnkt=kt)) then
    begin
      Kontr_T[kt].ID:=Pac1[jni].ID; Kontr_T[kt].Position_X:=round(x0);
Kontr_T[kt].Position_Y:=round(y0); if (jni3=j) then begin
Kontr_T[kt].Pressure:=round((Pac1[j].Pressure+Pac1[j-1].Pressure)/2); end
else if (jni3=ni) then begin
Kontr_T[kt].Pressure:=round((Pac1[ni].Pressure+Pac1[ni-1].Pressure)/2);
end; Kontr_T[kt].type_t:=3;  Kontr_T[kt].N_simv:=Pac1[jni].N_simv;
      kt:=kt+1; p:=p+1; goto enen;
    end;
    if (jni1=1) then begin jni1:=jni+1; if (jni=jni3) then begin jni:=jni-1; end
else if (jni=jni3-1) then begin jni:=jni+1; end; goto jni2; end
    else if((jni1=2) and (jni3=j)) then begin jni3:=ni; goto jni4; end else goto
enen;
enen:
  end;
  end;
end;
pr:=Pac1[j].Pressure;
end;
////////////////   }   Визначення КТ перетинання  ////////
//   Видалення кутових КТ  (за длиною), які знаходяться поруч  {
if(chb_bl.Checked=true) then
begin
  ut:=0; ut1:=0; ut2:=0;  ij:=0;  nvt:=1;
  uttt5: while (ut<=(kt-1)) do
          begin
            if (Kontr_T[ut].type_t=2) then break;
```

```
          ut:=ut+1;
        end;
  if(ut=kt) then goto ee;
if(nvt=1) then begin ut1:=ut; nvt:=2; ut:=ut+1; goto uttt5; end else ut2:=ut;
  if (Kontr_T[ut1].N_simv<>Kontr_T[ut2].N_simv) then begin ut1:=ut2;
nvt:=2; ut:=ut1+1; goto uttt5; end;
      ut12:=ut1;
      mu12: while (ij<=(i-1)) do
      begin
      if (Pac1[ij].ID>=Kontr_T[ut12].ID) then break;
      ij:=ij+1;
      end;
      if(ut12=ut1) then begin utt:=ij; utt1:=ij; ut12:=ut2; ij:=ij+1; goto mu12;
end else uttk:=ij;
  PR1:=0;
  while (utt1<uttk) do
  begin
     if(Pac1[utt1].Pressure=0) then break;
     PR1:=PR1+sqrt(sqr(Pac1[utt1].Position_X-Pac1[utt1+1].Position_X)+
sqr(Pac1[utt1].Position_Y-Pac1[utt1+1].Position_Y));
  utt1:=utt1+1;
  end;
    if(utt1<>uttk) then begin ut1:=ut2; nvt:=2; ut:=ut1+1; goto uttt5; end;
    if ((PR1<=StrToFloat(ed_bl.Text)) and (PR1>0)) then
    begin
      if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)<>
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-Pac1[utt-1].Position_Y)=
Sign(Pac1[utt+1].Position_Y-Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_Y<>Pac1[utt-1].Position_Y) then
per11:=180*(arctan(abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)/abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_Y<>Pac1[utt].Position_Y) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)/abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))/3.1415926535897932385 else per12:=90;
      end;
      if ((Sign(Pac1[utt].Position_X-Pac1[utt-1].Position_X)=
Sign(Pac1[utt+1].Position_X-Pac1[utt].Position_X)) and
((Sign(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)<>Sign(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)))) then
      begin
      if (Pac1[utt].Position_X<>Pac1[utt-1].Position_X) then
per11:=180*(arctan(abs(Pac1[utt].Position_Y-
Pac1[utt-1].Position_Y)/abs(Pac1[utt].Position_X-
Pac1[utt-1].Position_X)))/3.1415926535897932385 else per11:=90;
      if (Pac1[utt+1].Position_X<>Pac1[utt].Position_X) then
per12:=180*(arctan(abs(Pac1[utt+1].Position_Y-
Pac1[utt].Position_Y)/abs(Pac1[utt+1].Position_X-
Pac1[utt].Position_X)))/3.1415926535897932385 else per12:=90;
      end;
      per:=180-per11-per12;
      if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-1].Position_X)<>
Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-Pac1[uttk-1].Position_Y)=
Sign(Pac1[uttk+1].Position_Y-Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_Y<>Pac1[uttk-1].Position_Y) then
per21:=180*(arctan(abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)/abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_Y<>Pac1[uttk].Position_Y) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)/abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))/3.1415926535897932385 else per22:=90;
      end;
      if ((Sign(Pac1[uttk].Position_X-Pac1[uttk-1].Position_X)=
Sign(Pac1[uttk+1].Position_X-Pac1[uttk].Position_X)) and
((Sign(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)<>Sign(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)))) then
      begin
      if (Pac1[uttk].Position_X<>Pac1[uttk-1].Position_X) then
per21:=180*(arctan(abs(Pac1[uttk].Position_Y-
Pac1[uttk-1].Position_Y)/abs(Pac1[uttk].Position_X-
Pac1[uttk-1].Position_X)))/3.1415926535897932385 else per21:=90;
      if (Pac1[uttk+1].Position_X<>Pac1[uttk].Position_X) then
per22:=180*(arctan(abs(Pac1[uttk+1].Position_Y-
Pac1[uttk].Position_Y)/abs(Pac1[uttk+1].Position_X-
Pac1[uttk].Position_X)))/3.1415926535897932385 else per22:=90;
      end;
      per1:=180-per21-per22;
      if (per<=per1) then
      begin
        for u:=ut2 to kt-2 do
        begin
          Kontr_T[u]:=Kontr_T[u+1];
        end;
      end
       else
       begin
         for u:=ut1 to kt-2 do
        begin
          Kontr_T[u]:=Kontr_T[u+1];
        end;
        ut1:=ut2-1;
       end;
        kt:=kt-1;
    end
    else begin ut1:=ut2; ut:=ut1+1; end;
    nvt:=2;   ij:=ij+1;
    goto uttt5;
ee: end;
//   }   Видалення кутових КТ  (за длиною), які знаходяться поруч
/////   Визначення кількості КТ в кожному символі {
for ww:=0 to simv do
begin
  KKontr_T[ww]:=0;
end;
for w:=0 to kt-1 do
begin
KKontr_T[Kontr_T[w].N_simv]:=KKontr_T[Kontr_T[w].N_simv]+1;
end;
/////   }   Визначення кількості КТ в кожному символі
////////////////   }   Визначення КТ   ////////////////////////////
for ww:=0 to simv do
begin
if (KKontr_T[ww]>30) then goto endfn;
end;
sww:=0;
for ww:=0 to simv do
begin
  for i:=0 to 1000  do
  begin
    N_O[i]:=0;
  end;
  N_O_K[ww]:=KKontr_T[ww];
  for i:=0 to KKontr_T[ww]-1 do
  begin
     N_O[3*i+3-3]:=Kontr_T[sww+i].Position_X ;
     N_O[3*i+3-2]:=Kontr_T[sww+i].Position_Y;
     N_O[3*i+3-1]:=Kontr_T[sww+i].type_t;
  end;
   ssn:=0;
  for i:=0 to 3*KKontr_T[ww]-1 do
  begin
    ssn:=ssn+sqr(N_O[i]);
  end;
  for i:=0 to 3*KKontr_T[ww]-1 do
  begin
    N_O_N[i]:=N_O[i]/sqrt(ssn);
  end;
  for ib:=0 to kol_b-1 do
  begin
  kuo[ib]:=0;
  SB:=S+'\Data_bukvy\' + Bukvy1[ib] + '\' + Bukvy1[ib] + '_' +
IntToStr(N_O_K[ww]) + '.db';
  iff:=FindFirst(SB,faAnyFile, SR);
  if (iff<>0) then begin  end
  else
  begin
    tb_raspozn.TableName := SB; tb_raspozn.Open();
   while tb_raspozn.Eof<>true do
    begin
      for i:=0 to 3*KKontr_T[ww]-1 do
```

```
    begin
     U_O[i][kuo[ib]][ib]:=tb_raspozn.Fields[i+1].AsInteger;
    end;
    kuo[ib]:=kuo[ib]+1;  tb_raspozn.Next();
   end;
   tb_raspozn.Close();
  end;
  FindClose(SR);
  end;
  for ib:=0 to kol_b-1 do
  begin
  for iu:=0 to kuo[ib]-1 do
  begin
   ssn:=0;
   for i:=0 to 3*KKontr_T[ww]-1 do
    begin
     ssn:=ssn+sqr(U_O[i][iu][ib]);
    end;
   for i:=0 to 3*KKontr_T[ww]-1 do
    begin
     U_O_N[i][iu][ib]:=U_O[i][iu][ib]/sqrt(ssn);
    end;
  end;
  end;
  for ib:=0 to kol_b-1 do
  begin
  Ver[ib]:=0;
  for iu:=0 to kuo[ib]-1 do
  begin
   ssn1:=0;
   for i:=0 to 3*KKontr_T[ww]-1 do
    begin
     ssn1:=ssn1+U_O_N[i][iu][ib]*N_O_N[i];
    end;
   Ver[ib]:=Ver[ib]+exp((ssn1-1)/0.01);
  end;
  end;
  maxv:=Ver[0];  nmaxv:=0;
  for ib:=0 to kol_b-1 do
  begin
  if (Ver[ib]>maxv) then begin maxv:=Ver[ib]; nmaxv:=ib; end;
  if (ww=0) then begin Edit1.Text:=Edit1.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit1.Refresh(); end
  else if (ww=1) then begin Edit2.Text:=Edit2.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit2.Refresh(); end
  else if (ww=2) then begin Edit3.Text:=Edit3.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit3.Refresh(); end
  else if (ww=3) then begin Edit4.Text:=Edit4.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit4.Refresh(); end
  else if (ww=4) then begin Edit5.Text:=Edit5.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit5.Refresh(); end
  else if (ww=5) then begin Edit6.Text:=Edit6.Text + Bukvy1[ib] + ' - ' +
FloatToStr(Ver[ib]) + ';  '; Edit6.Refresh(); end;
  end;
  VBukvy1[ww][nmaxv]:=VBukvy1[ww][nmaxv]+1;
  sww:=sww+KKontr_T[ww];
end;
Edit7.Text:=''; Edit8.Text:=''; Edit9.Text:=''; Edit10.Text:=''; Edit11.Text:='';
Edit12.Text:=''; Edit13.Text:=''; Edit14.Text:=''; Edit15.Text:='';
Edit16.Text:=''; Edit17.Text:=''; Edit18.Text:=''; Edit19.Text:='';
Edit21.Text:='';
Edit7.Refresh(); Edit8.Refresh(); Edit9.Refresh(); Edit10.Refresh();
Edit11.Refresh(); Edit12.Refresh(); Edit13.Refresh(); Edit14.Refresh();
Edit15.Refresh(); Edit16.Refresh(); Edit17.Refresh(); Edit18.Refresh();
Edit19.Refresh(); Edit20.Refresh(); Edit21.Refresh();
SVP1:=0;
for ww:=0 to simv do
begin
SVB:=0;
for ib:=0 to kol_b-1 do
 begin
 SVB:=SVB+VBukvy1[ww][ib];
 if (ww=0) then begin Edit7.Text:=Edit7.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit7.Refresh(); end
 else if (ww=1) then begin Edit8.Text:=Edit8.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit8.Refresh(); end
 else if (ww=2) then begin Edit9.Text:=Edit9.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit9.Refresh(); end
 else if (ww=3) then begin Edit10.Text:=Edit10.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit10.Refresh(); end
 else if (ww=4) then begin Edit11.Text:=Edit11.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit11.Refresh(); end
 else if (ww=5) then begin Edit12.Text:=Edit12.Text + Bukvy1[ib] + ' - ' +
IntToStr(VBukvy1[ww][ib]) + ';  '; Edit12.Refresh(); end;
 end;
if (ww=0) then begin if (SVP[ww]<>VBukvy1[ww][2]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][2]; if (VBukvy1[ww][2]=0) then
Edit13.Text:=IntToStr(0) else
Edit13.Text:=FloatToStr(100*VBukvy1[ww][2]/SVB); Edit13.Refresh(); end
else if (ww=1) then begin if (SVP[ww]<>VBukvy1[ww][3]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][3]; if (VBukvy1[ww][3]=0) then
Edit14.Text:=IntToStr(0) else
Edit14.Text:=FloatToStr(100*VBukvy1[ww][3]/SVB); Edit14.Refresh(); end
else if (ww=2) then begin if (SVP[ww]<>VBukvy1[ww][0]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][0]; if (VBukvy1[ww][0]=0) then
Edit15.Text:=IntToStr(0) else
Edit15.Text:=FloatToStr(100*VBukvy1[ww][0]/SVB); Edit15.Refresh(); end
else if (ww=3) then begin if (SVP[ww]<>VBukvy1[ww][1]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][1]; if (VBukvy1[ww][1]=0) then
Edit16.Text:=IntToStr(0) else
Edit16.Text:=FloatToStr(100*VBukvy1[ww][1]/SVB); Edit16.Refresh(); end
else if (ww=4) then begin if (SVP[ww]<>VBukvy1[ww][2]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][2]; if (VBukvy1[ww][2]=0) then
Edit17.Text:=IntToStr(0) else
Edit17.Text:=FloatToStr(100*VBukvy1[ww][2]/SVB); Edit17.Refresh(); end
else if (ww=5) then begin if (SVP[ww]<>VBukvy1[ww][4]) then
SVP1:=SVP1+1; SVP[ww]:=VBukvy1[ww][4]; if (VBukvy1[ww][4]=0) then
Edit18.Text:=IntToStr(0) else
Edit18.Text:=FloatToStr(100*VBukvy1[ww][4]/SVB); Edit18.Refresh();
end;
end;
if (SVP1=simv+1) then begin if (Edit20.Text<>'') then
Edit20.Text:=IntToStr(StrToInt(Edit20.Text)+1) else
Edit20.Text:=IntToStr(1); end;
if (Edit22.Text<>'') then Edit22.Text:=IntToStr(StrToInt(Edit22.Text)+1) else
Edit22.Text:=IntToStr(1);
if (Edit20.Text<>'') then
Edit21.Text:=FloatToStr(100*StrToInt(Edit20.Text)/SVB);
Edit19.Text:=FloatToStr((StrToFloat(Edit13.Text)+StrToFloat(Edit14.Text)+
StrToFloat(Edit15.Text)+StrToFloat(Edit16.Text)+StrToFloat(Edit17.Text)+
StrToFloat(Edit18.Text))/(simv+1));
Edit19.Refresh(); Edit20.Refresh(); Edit21.Refresh(); Edit22.Refresh();
if (rb_one.Checked=true) then goto endo;
 end;
endfn:
iffn:=FindNext(SRn);
 end;
FindClose(SRn);
end;
endo:
end;
end.
```