МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНЖЕНЕРІЇ ТА УПРАВЛІННЯ
НАЦІОНАЛЬНОГО АВІАЦІЙНОГО УНІВЕРСИТЕТУ»

ДОПУСТИТИ ДО ЗАХИСТУ
Заступник директора з НР
_____ О.В. Родіонова
«____»_____ 2021 р.

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ

«БАКАЛАВР»

Тема: _____ Програмна реалізація обробки сюрреальних чисел

Автор: Помін Д.А.

Керівник проєкту: Куц К.Б.

Нормконтролер: В.М. Кругляк

Київ 2021

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНЖЕНЕРІЇ ТА УПРАВЛІННЯ
НАЦІОНАЛЬНОГО АВІАЦІЙНОГО УНІВЕРСИТЕТУ»

Циклова комісія: Інженері програмного забезпечення
Освітнього ступеня: «Бакалавр»
Спеціальність: 123 Комп'ютерна інженерія
Освітньо-професійна програма: «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Голова циклової комісії
_____ Н.А. Рябчук
«____»_____ 2021 р.

ЗАВДАННЯ

на виконання дипломного проєкту

ПІБ

1. Тема роботи: «Програмна реалізація обробки сюрреальних чисел»

затверджена наказом від «15» березня 2021 року № 28-Ст.

2. Термін виконання: з 12.04.2021р. по 20.06.2021р.

3. Вихідні дані: Розробити калькулятор для роботи з сюрреальними числами.

4. Зміст пояснювальної записки:

У 1 розділі проаналізувати завдання на проєкт та основні методи його розв'язання.

У 2 розділі наведести опис основних етапів розробки.

У 3 розділі описати основні вимоги охорони праці.

КАЛЕНДАРНИЙ ПЛАН

виконання дипломної роботи

№ п/п	Етапи виконання кваліфікаційної роботи	Термін виконання етапів	Примітка
1.	Уточнення постановки задачі	12.04.2021	Виконано
2.	Аналіз літературних джерел	15.04.2021	Виконано
3.	Обґрунтування рішення	21.04.2021	Виконано
4.	Збір інформації	01.05.2021	Виконано
5.	Аналіз існуючих методів. Обгрунтування вибору мови програмування	06.05.2021	Виконано
6.	Виконання проєкту	08.06.2021	Виконано
7.	Оформлення і друк пояснювальної записки	15.06.2021	Виконано
8.	Оформлення презентації	17.06.2021	Виконано
9.	Отримання рецензій	20.06.2021	Виконано
10.	Захист проєкту		

Дипломник Помін Д.А.

(підпис, дата) (П.І.Б.)

Дипломний керівник Куц К.Б.

(підпис, дата) (П.І.Б.)

Консультант з охорони праці Пешков І.В.

(підпис, дата) (П.І.Б.)

Зміст

Вступ

Дослідження теоретичних засад розв’язання задач, що володіють властивостями слабоструктурованості або неструктурованості взагалі пов’язане з низкою проблем. Такі задачі характеризуються відсутністю методів розв’язання на основі безпосередніх перетворень даних. Неструктуровані задачі містять неформалізовані процедури, які характеризуються високою степеню невизначеності вхідної інформації, а також її представлення для подальшої обробки і використання. Функціональні залежності в вказаному класі задач можуть характеризуватись:

1. нелінійністю;
2. недифиринціовністю;
3. багатокритеріальністю (для аналітичних задач);
4. овражністю;
5. відсутністю аналітичного вираження;
6. складною топологією області допустимих значень;
7. високою обчислювальною складністю оптимізуючих функцій;
8. високою розмірністю простору пошуку, тощо…

Прикладом задачі, що володіє вказаними властивостям може стати задача побудови системи підтримки прийняття рішень, або системи з інтелектуальною складовою, яка володіє ситуаційними пораджувальними можливостями. На сьогодні такі системи побудовані на основі штучного інтелекту (високого і низького рівнів), зокрема, з використанням нечітких ситуаційних алгоритмів обробки інформації, евристичних алгоритмів (наприклад алгоритмів, побудованих біоінспірованими методами: еволюційними, популяційними, штучними методами систем організмів). Але, серед всіх підзадач вказаної задачі можна виділити окремий підклас, для якого застосовні ігрові моделі. Для цього підкласу підзадач може стати корисним

спеціальне представлення числових даних, яке використовуються, і більш того, бере свій початок в комбінаторній теорії ігор – сюрреальне представлення.

Теорію сюрреальних чисел, в певній мірі, можна сприймати як певне теоретико-ігрове узагальнення теорії дійних чисел. Але це узагальнення є альтернативним визначенням чисел, в якому класичний ланцюг включень $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C} \subset \cdots$ замінюється на ланцюг $\mathbb{N} \subset \mathbb{Q}_2 \subset \mathbb{S}$, де $\mathbb{Q}_2$-множина двійково-раціональних чисел, $\mathbb{S}$-множина сюрреальних чисел, що включає, зокрема, множину дійсних чисел, всіх трансфінітних чисел Кантора, множину чисел, обернених числам Кантора, а також нескінченні класи чисел, пов'язаних з першим кардинальним числом Кантора ω.

Питання про використання сюрреальних чисел для комп'ютерної обробки в термінах визначення Конвея є досить складним і глибоким. В рамках вивчення цього питання існує ідея побудови моделі зображення сюрреальних чисел певним двійковим кодом. На сьогодні, для представлення сюрреальних чисел використовують три основні символи $1, 0,$ і символ числа нуль Z.

Метою дипломного проекту є вивчення теоретичних аспектів теорії сюрреальних чисел, розробка комп'ютерного представлення, та розробка алгоритмів побудови сюрреального числа, що відповідає певному дійсному числу і алгоритмів виконання арифметичних операцій над сюрреальними числами.

1. Загальна частина

1.1 Постановка задачі

Функціональні вимоги для програмного забезпечення складають собою основу, котра повинна бути задовільнена під час розробки платформи, тому займає найбільшу частину часу розробника. Функціонал системи повинен мати найкращі характеристики існуючих рішень та пропонувати покращення чи аналоги функціоналу з інших типів систем, тим самим бути агрегатором найкращих рис існуючих комерційних рішень у суміжних сферах.

Значною проблемою при розробці функціональних вимог до даного програмного засобу стало те, що на теперешній час практично не існує проектів, що реалізують виконання арифметичних операцій над сюрреальними числами. Хоча ця область відносно не надто нова в математиці, проте дослідження її з точки зору комп'ютерної реалізації дуже обмежені. Існує лише декілька реалізацій, що частково реалізують арифметичні дії з сюрреальними числами, проте всі вони є розширеннями для спеціалізованих середовищ математичного моделювання, та, як заявлено у їх описах, використовують функціональність, специфічну для середовища виконання додаткову функціональність, яку досить складно відтворити в неспеціалізованих середовищах програмування.

Задачею розробки даного програмного засобу є дослідити контрукцію та властивості сюрреальних чисел, та арифметичних операцій над ними, і знайти таке їх представлення, яке буде найбільш зручним для використання при комп'ютерній обробці, та матиме гарні характеристики обчислювальної складності та ресурсозатратності виконання арифметичних операцій над сюрреальними числами в такому представлені, та реалізувати алгоритми основних арифметичних операцій над сюрреальними числами, найбільш оптимальним чином, при цьому не втрачаючи математичну коректність, та, по можливості, загальність алгоритму.

Крім цього була поставлена задача реалізувати необхідні алгоритми, та структури даних таким чином, щоб аналогічний функціонал можливо було легко відтворити на будь якій мові програмування загального призначення, що реалізує об'єктно-орієнтовану або процедурну парадигму програмування, що дозволить використовувати данні розробки в більш широкому наборі випадків, майже незалежно від середовища створення (окрім синтаксичних особливостей конкретної мови програмування) та виконання програми, на відміну від існуючих аналогів.

1.2 Теоретичні відомості

Сюрреальні числа є результатом певного узагальнення звичайних дійсних чисел та нескінченних порядкових чисел, що вперше було використане в роботах англійського математика Джона Конвея для дослідження деяких аспектів теорії ігор.

Розгляд множини сюрреальних чисел логічно буде почати з введення основних понять теорії ігор для дослідження яких початково використовувалися сюрреальні числа.

Розглянемо гру Червоно-синій хакенбуш. Гра складається з ліній червоно та синього кольорів, та базової лінії, що не має кольору и з нею не можуть бути здійсненні жодні дії. Можлива модифікація правил гри в якій базова лінія не проводиться і її функції виконує початок ігрового простору. Ігровим простором може бути як звичайний аркуш паперу, так і будь яка обмежена в одному напрямку площина. Саме це обмеження площини може грати роль базової лінії. В інших напрямках ігровий простір може бути нескінченним та містити нескінченну кількість червоних та синіх ліній. Лінії можуть бути з'єднанні між собою незалежно від кольору. В одній точці може бути з'єднане необмежена кількість ліній. Всі лінії мають бути з'єднанні з базовою лінією безпосередньо, або через

інші лінії, з якими вони з'єднуються. Якщо між лінією немає жодного зв'язку з базовою лінією, то вона автоматично видаляється В гру грають два гравці, Червоний та Синій, що можуть видаляти лише лінії відповідного до них кольору. Кожен гравець під час свого ходу має видалити одну лінію свого кольору. Програє той гравець, що першим не може зробити хід, тобто перший для кого не залишиться ліній відповідного кольору.

Для аналізу цієї гри а також ігор такого типу використовуються методи комбінаторної теорії ігор. Для початку назвемо гравця, що може видаляти лише сині лінії Лівим гравцем, а гравця, що може видаляти тільки червоні лінії Правим гравцем.

В комбінаторній теорії ігор гра визначається як $\{G^L|G^R\}$, де G^L – множина всіх можливих ходів які може зробити Лівий гравець, а G^R – множина всіх можливих ходів які може зробити Правий гравець. Кожен хід в цих множинах представляє собою гру виду $\{G_1^L|G_1^R\}$, в яку трансформується початкова гра $\{G^L|G^R\}$, після певної дії гравця, що робить визначення гри рекурентним.

Розглянемо гру, що складається з двох порожніх множин : $\{\emptyset|\emptyset\}$. Згідно з вище приведеним визначенням можна сказати що в цій грі жоден з гравців не має жодної можливості зробити хід. Розглядаючи клас ігор в яких поразку зазнає той гравець, який перший не може зробити хід, про таку гру можна сказати, що в ній завжди перемагатиме гравець, який ходить другим, адже гравець який робить перший хід, не може зробити жодної дії і згідно з правилами гри зазнає поразки, тоді як інший гравець автоматично отримує перемогу. Будемо називати таку гру *кінцем гри.* На прикладі Червоно-синього хакенбушу цю ситуацію можна проілюструвати як пустий ігровий простір, що містить лише базову лінію. Надалі будемо розуміти під грою саме Червоно-синій хакенбуш та розглядати всі твердження на цьому прикладі.

Розглянемо інші можливі стани гри. Припустимо, що обидва гравця мають однакові можливості, тобто можуть робити однакові ходи, тобто приводити гру до однакових станів. В нашому прикладі це означає, що Лівий та Правий гравець мають однакові фігури складені з ліній їх кольору (хоча можливі і більш складні випадки розміщення ліній при якому гравці мають однакові можливості). Розглянемо як буде протікати така гра. Гравець що робить перший хід, прибираючи одну лінію зі своєї фігури. Припустимо, що обидва гравця діють за оптимальною стратегією, тоді другий гравець, що має таку ж саму фігуру повторить хід першого. Таке повторення буде продовжуватися до того моменту коли в гравців залишиться лише по одній лінії відповідного кольору. Тоді перший гравець прибере свою останню лінію, і так само зробить другий гравець. Це приведе нас до гри $\{\emptyset|\emptyset\}$ і як тільки перший гравець зробить свій наступний хід, він програє, бо не буде мати жодної можливості прибрати лінію, і перемога автоматично перейде до гравця, який ходив другий. Будемо вважати, що значення такої гри дорівнює нулю. Якщо Лівий гравець, наприклад, матиме 8 синіх ліній безпосередньо поєднаних з базовою лінією, а Правий матиме 5 аналогічно розміщених ліній, то Лівий гравець буде мати перевагу в 3 лінії, і значення гри буде 3. Якщо ж складеться протилежна ситуація і правий гравець матиме 8 ліній, а Лівий 5, то перевага у 3 лінії буде в Правого гравця. Для зручності будемо вимірювати перевагу Правого гравця у від'ємних величинах, тоді значення такої гри буде -3. З цього можна отримати три прості правила, завдяки яким можна проаналізувати будь яку гру:

- Якщо $G < 0$, то перемагає Правий гравець;
- Якщо $G = 0$, то перемагає гравець, що ходить другим;
- Якщо $G > 0$, то перемагає Лівий гравець;

В цих правилах «перемагає», означає, що гравець має переможну стратегію, але можливий випадок, що він зробить помилковий, неоптимальний хід і може

програти. Досліджуючи значення гри таким чином, введемо позначення $\{\emptyset|\emptyset\} = 0$, що погоджено з нашім попереднім визначенням нульової гри, так як кожен гравець має однакову множину ходів, і в цьому конкретному випадку це порожня множина. Тоді якщо ми розглянемо гру $\{0|\emptyset\}$ ми отримаємо, що лівий гравець має один єдиний хід, яким може привести гру до стану $\{\emptyset|\emptyset\}$, в той час як Правий гравець не має жодного ходу. В такій грі Лівий гравець має перевагу в один хід, тому $\{0|\emptyset\} = 1$. ЗА таким самим принципом аналізуючи гру $\{\emptyset|0\}$ ми маємо перевагу Правого гравця в один хід, тому отримаємо, що $\{\emptyset|0\} = -1$, відповідно до домовленості вимірювати перевагу Правого гравця в від'ємних величинах. Так само можливо проаналізувати і більш складну гру, яка представлена на рисунку 1.2.1.

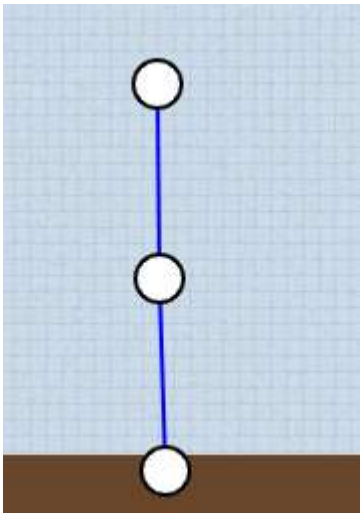

Рис. 1.2.1

В цій грі Правий гравець не має жодного ходу, а лівий має дві можливості здійснити хід. Він може видалити найнижчу лінію, чим привести гру до стану $\{\emptyset|\emptyset\} = 0$, чи верхню лінію, що приведе гру до стану $\{0|\emptyset\} = 1$. Представлення цієї гри в прийнятій нами формі буде таким : $\{\{\emptyset|\emptyset\}, \{0|\emptyset\}|\emptyset\}$, що іншими словами виглядає як $\{0,1|\emptyset\}$. Якщо змінити колір обох ліній цієї гри на червоний, ми отримаємо гру $\{\emptyset|\{\emptyset|\emptyset\}, \{\emptyset|0\}\}$, що іншими совами виглядає як $\{\emptyset|0, -1\}$.

Будемо вважати, що оптимальною стратегією у грі є та, яка залишає в гравця більшу перевагу, тобто більше ліній відповідного кожному гравцеві кольору. Тоді

розглядаючи дві вище приведені гри можна спростити їх представлення таким чином, що для Лівого гравця обрати хід який має найбільше значення, серед усіх можливих в множині ходів Лівого гравця, а для Правого гравця обрати хід, що має найменше значення, і дає найбільшу перевагу Правому гравцю, серед усіх можливих ходів. Спрощуючи за таким правилом ми отримаємо $\{0,1|\emptyset\} = \{1|\emptyset\}$, та $\{\emptyset|0,-1\} = \{\emptyset|-1\}$. Отримані ігри матимуть значення $\{1|\emptyset\}$=2, та $\{\emptyset|-1\} = -2$, тому що в кожній з них Лівий або Правий гравець відповідно матимуть перевагу у два ходи.

Далі корисно буде проаналізувати значення гри представленої на рисунку 1.2.2.

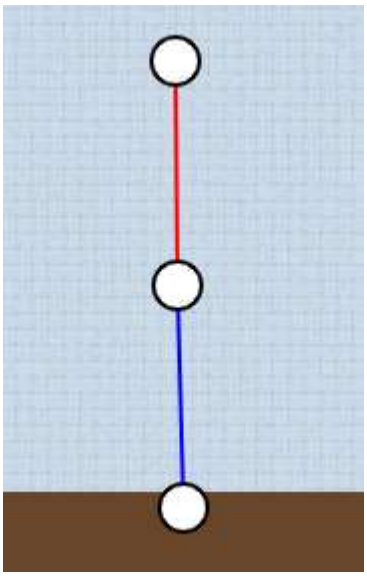

Рис.1.2.2

Цю гру можна представити як $\{0|1\}$, так як Лівий гравець має один єдиний хід – прибрати нижню лінію свого кольору, після чого червона лінія видалиться автоматично і гра перейде в стан $\{\emptyset|\emptyset\}$. Правий гравець може прибрати лише верхню лінію, чим приведе гру в стан $\{0|\emptyset\}$. Чому дорівнює значення такої гри. Воно більше за 0, тому що Лівий гравець перемагає в будь якому випадку. Але воно не дорівнює 1, що можна довести дослідивши значення цієї гри після

додавання іншої гри, що складається з однієї червоної лінії, як показано на рисунку 1.2.3.

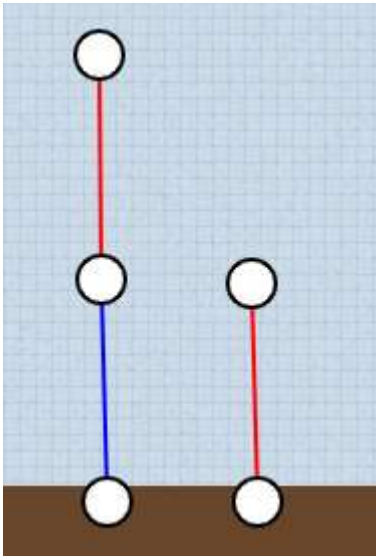

Рис.1.2.3

Так як значення доданої гри дорівнює -1, то якщо б значення гри на рисунку 1.2.2 дорівнювало б 1, то ми отримали би нульову гру. Проте очевидно що в цій грі завжди перемагає Правий гравець, що суперечить означенню нульової гри. Припустимо, що значення гри на рисунку 1.2.2 дорівнює $\frac{1}{2}$. Це можна довести,якщо після додавання до гри на рисунку 1.2.3 ще однієї гри на рисунку 1.2.2, ми отримаємо нульову гру, так як $\frac{1}{2} + \frac{1}{2} - 1 = 0$. Справді при дослідженні гри на рисунку 1.2.4, приходимо до висновку що при використанні оптимальної стратегії перемогу завжди одержить гравець, що починає грати другим.

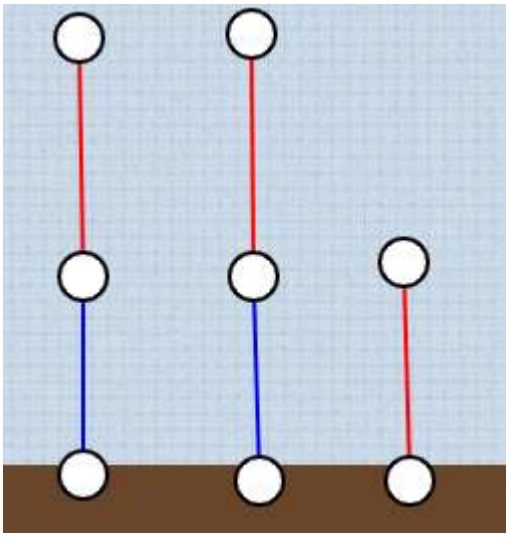

Рис. 1.2.4

Отже гра може мати як ціле, так і дробове значення. Аналогічно можна довести, що гра, яка утвориться зміною кольорів ліній в грі на рисунку 1.2.3, на протилежні, ми отримаємо значення $\{-1|0\} = -\frac{1}{2}$.

Саме приведені вище поняття значення гри, його властивості, та методи його обчислення стали основою формування поняття про множину сюрреальних чисел.

Сюрреальне число – це гра $\{L|R\}$. Для побудови множини сюрреальних чисел $\mathbb{S}$ (так надалі будемо позначати множину сюрреальних чисел) Джон Конвей вводить дві наступні аксіоми (далі під числом будемо розуміти сюрреальне число):

- Аксіома 1: кожне число співвідноситься з двома множинами попередньо створених чисел, таких що, в лівій множині не існує такого елемента, що був би більший, або дорівнював будь якому елементу в правій множині. Математично це можна записати так : $x \equiv \{X_L|X_R\}$, $\forall x_L \in X_L, x_R \in X_R,\ \nexists\ x_L \geq x_R$
- Аксіома 2: одне число менше або дорівнює другому, якщо, і тільки якщо, не існує числа в лівій множині першого числа, такого, що більше або дорівнює другому числу, та не існує такого числа в правій множині другого числа, такого, що менше, або дорівнює першому числу.

Математично це можна записати так:

$x \leq y, x \equiv \{X_L|X_R\}\, y \equiv \{Y_L|Y_R\},\ \forall\, x_L \in X_L \nexists\, y \leq x_l$

$\forall\, y_R \in Y_R \nexists\, y_R \leq x$

Виходячи з аксіоми 1, що кожне число співвідноситься з двома множинами попередньо створених чисел, перше число, яке можливо отримати є число $\{\emptyset|\emptyset\}$, тому що мі не маємо жодного попередньо створеного числа, тому обидві множини пусті. Як вже було сказано таку гру (або число) ми будемо позначати $\{\emptyset|\emptyset\} \equiv 0$. Це число Відповідає аксіомі 1, тому що обидві множини пусті, тому не існує елемента здатного порушити її правила.

Надалі говорячи про сюрреальні числа, мі будемо використовувати знак «≡» для позначення того, що два числа є ідентичними. Запис $\{\emptyset|\emptyset\} \equiv 0$ означає, що 0 є просто формою запису числа $\{\emptyset|\emptyset\}$. Важливо зауважити, що «≡» означає не теж саме, що і «=». Ми будемо вважати, що два сюрреальних числа y та x є рівними і записувати $y = x$, якщо наступне твердження є істинним: $x \leq y \wedge y \leq x$. Відповідно до цього визначення ми будемо казати, що два сюрреальні числа не рівні між собою, якщо наступне твердження є істинним: $x \nleq y \vee y \nleq x$.

Маючи число 0 ми можемо побудувати три наступні гри: $\{0|\emptyset\}$, $\{\emptyset|0\}$, та $\{0|0\}$. Проте остання побудована нами гра не є числом, так як не відповідає аксіомі 1. З цього можна зробити висновок, що *кожне число є грою, проте не кожна гра є числом*. Що стосується гри Червоно-синій хакенбуш, на прикладі якого в цій роботі розглядається побудова та використання множини сюрреальних чисел Джоном Конвеєм була доведена наступна теорема, що складається з двох тверджень:

- Якщо синя лінія видаляється з гри, то значення гри строго зменшується, а якщо червона лінія видаляється з гри, то значення гри строго збільшується.
- Будь яка гра Червоно-синього хакенбушу є числом.

Отже після побудови числа 0, на наступному кроці ми можемо побудувати числа $\{0|\emptyset\} \equiv 1$ та $\{\emptyset|0\} \equiv -1$, як це вже було доведено для ігор. Тепер скористаємося аксіомою 2, та порівняємо отримані два числа з 0. Досить легко довести твердження, що $\{\emptyset|\emptyset\} \leq \{0|\emptyset\}$, бо справді, в лівій множині числа $\{\emptyset|\emptyset\}$ не існує жодного числа, що більше, або дорівнює 1, так як це порожня множина, і також в правій множині числа $\{0|\emptyset\}$ не існує жодного числа, що менше, або дорівнює 0, бо це порожня множина. Також ми можемо показати, що твердження $\{0|\emptyset\} \leq \{\emptyset|\emptyset\}$, є хибним, тому що ліва множина першого числа містить число 0, яке дорівнює другому числу тому умови аксіоми 2 не задовольняються. Таким чином ми отримаємо, що *0 менше і не дорівнює 1*. Таким самим методом ми можемо довести, що дійсне твердження $\{\emptyset|0\} \leq \{\emptyset|\emptyset\}$, а також хибність твердження $\{\emptyset|\emptyset\} \leq \{\emptyset|0\}$, і це означатиме, що *-1 менше та не дорівнює 0.* Таким чином ми обґрунтували використання додатних позначень для чисел виду $\{n|\emptyset\}$, та від'ємні позначення для чисел виду $\{\emptyset|n\}$.

Ґрунтуючись на цих міркуваннях можна встановити наступні відношення для двох сюрреальних чисел. Будемо вважати, що для двох сюрреальних числа y та x, число x є строго меншbм за y, і записувати $x < y$ якщо істинне наступне твердження: $x \leq y \wedge y \nleq x$. При істинності цього ж твердження ми можемо казати, що число y строго більше за x, і записувати це $y > x$.

Ми встановили, що $-1 < 0 < 1$, проте ми також можемо сказати, що 0 був створений раніше за -1 та 1. Це вказує на те, що на множині сюрреальних чисел існує окремий порядок, не пов'язаний з порядком «бути більше або бути менше», і це порядок «бути раніше».

Ми будемо називати число x *простішим* за число y, якщо число x було створено раніше ніж y.

Далі, маючи три попередньо створених числа, ми можемо створити значну кількість чисел, проте більшість з них будуть такими, що не задовольняють аксіому

1, так як, наприклад, число $\{1|-1\}$, та інші. Відкинувши такі числа, м отримаємо 17 чисел, що створені з трьох попередньо створених чисел 0, 1 та -1, та задовольняють аксіому 1. Перерахуємо ці числа: $\{-1\mid\},\{\mid -1\},\ \{1\mid\},\{\mid 1\},\ \{-1,0\mid\},\{-1\mid 0\},\{\mid -1,0\},$ $\{0,1\mid\},\{0\mid 1\},\{\mid 0,1\},\{-1,1\mid\},\{-1\mid 1\},\{\mid -1,1\},\ \{-1,0,1\mid\},\{-1,0\mid 1\},\{-1\mid 0,1\},\quad \{\mid -1,0,1\}.$

Користуючись аксіомою 2 та вище наведеними домовленостями, легко довести, що $\{1|\emptyset\} > 1$, а також те, що $\{\emptyset|-1\} < -1$. Назвемо ці числа 2, та -2 відповідно.

Також можна довести, що $\{0|1\} > 0$ і одночасно $\{0|1\} < 1$. Як це було вже зроблено раніше в дослідженні значення гри на рисунку 2, Назвемо це число $\frac{1}{2}$. Далі, ми зможемо навести більш детальній доказ обґрунтованості такого позначення, дослідивши операцію додавання сюрреальних чисел, та проілюструвавши, що $\{0|1\} + \{0|1\} = 1$.

Розглянемо ще одне число, із останніх створених – число $\{-1|1\}$. Використовуючи аксіому 2, ми можемо довести, що $\{-1|1\} \leq 0$. В лівій множині числа $\{-1|1\}$ не існує жодного числа, що було б більше, або дорівнювало числу 0 , і в правій множині числа $\{\emptyset|\emptyset\} \equiv 0$, не існує жодного числа, що було б менше, або дорівнювало числу $\{-1|1\}$, так як це порожня множина. Спираючись на ці два доведених тверджень та попереднє визначення рівності двох чисел, ми можемо зробити висновок, що число $\{-1|1\}$ дорівнює 0, тобто $\{-1|1\} = 0$. Зауважимо, що саме «=» - дорівнює , а не «≡» - ідентичне, бо ліва і права множини в з цих числах відповідно, кардинально відрізняються. Це приводить нас до того, що $\{-1|1\}$ та $\{\emptyset|\emptyset\}$, є різними відображеннями одного й того самого значення 0. Узагальнюючи цю думку, можна зробити висновок, що одному значенню можуть відповідати декілька представлень, як наприклад:

$\{-1 \mid\} = 0\ \{\mid 1\} = 0\ \{-1, 0 \mid\} = 1\ \{\mid -1, 0\} = -2\ \{0, 1 \mid\} = 2\ \{\mid 0, 1\} = -1\ \{-1, 1 \mid\} = 2\ \{\mid -1, 1\} = -2\ \{-1, 0, 1 \mid\} = 2\ \{-1, 0 \mid 1\} = 1/2\ \{-1 \mid 0, 1\} = -1/2\ \{\mid -1, 0, 1\} = -2.$

Далі наведено доведення того, що коли ми визначаємо значення сюрреального числа, нам необхідно враховувати лише найбільше значення в його правій множині, та найменше значення в його лівій множині. Це було проілюстровано при дослідженні значення гри, проте надалі отримає узагальнене доведення для чисел.

Основні властивості сюрреальних чисел.

В цьому пункті буде наведено доведення усіх основних властивостей сюрреальніх чисел які необхідні як для їх дослідження, так і для практичного використання. Більшість цих властивостей добре відомі для дійсних чисел, проте при побудові множини сюрреальних чисел вони потребують відповідного доведення.

Теорема 1.2.1 Якщо x сюрреальне число, то $x \leq x$.

Доведення. Для доведення цієї теореми скористаємося методом індукції. Він полягає в доведенні теореми через доведення двох наступних тверджень:

- Доведення, що теорема дійсна для числа $\{\emptyset|\emptyset\}$
- Доведення, що теорема дійсна для числа x, якщо вона дійсна для чисел, що є батьківськими для числа x, тобто всіх елементів лівої та правої множини, які визначають це число.

Доведемо теорему 1.2.1 для числа $\{\emptyset|\emptyset\}$, яке є батьківським для всіх сюрреальних чисел. Не важко довести, що відповідно до аксіоми 2 дійсне твердження $0 \leq 0$, тому що не існує жодного елемента в лівій множині числа $\{\emptyset|\emptyset\}$, такого, що більший або дорівнює 0, бо це порожня множина. Так само в правій множині числа $\{\emptyset|\emptyset\}$ не існує жодного елемента, такого що менше, або дорівнює 0. Отже $0 \leq 0$.

Тепер доведемо, що теорема дійсна для батьківських чисел числа x, тобто для усіх елементів з правої та лівої множини, якими визначене це число. Твердження $x \leq x$ є дійсним тоді і тільки тоді, коли дійсним є наступне твердження : $\nexists x_L \in X_L : x \leq x_L \wedge \nexists x_R \in X_R : x_R \leq x$.

Ліва частина цього твердження вимагає довести, що x не менше і не дорівнює жодному елементу з лівої множини числа x. Іншими словами це твердження означає, що $\exists a \in X_L : x_L \leq a \vee \exists b \in X_{LR} : b \leq x$. X_{LR} в цьому твердженні позначає прву множину числа x_L. Ми припустили, що теорема 1.2.1 вірна для всіх батьківських чисел для x. Отже ми можемо припустити, що $x_L \leq x_L$. Обравши значення a таке, що $a \equiv x_L$ ліва частина цього твердження стає дійсною і це робить дійсним все твердження. Повертаючись до доведення теореми 1.2.1, доведемо праву частину твердження $\nexists x_L \in X_L : x \leq x_L \wedge \nexists x_R \in X_R : x_R \leq x$. Права чистина цього твердження означає, що не x_R не менше і не дорівнює x, для будь яких значень $x_R \in X_R$. Іншими словами це твердження можна записати так: $\exists c \in X_{\mathrm{RL}} : x \leq c \vee \exists d \in X_R : d \leq x_R$, де X_{RL} позначає ліву множину числа x_R. Ми припустили, що теорема 1.2.1 вірна для всіх батьківських чисел для x. Отже ми можемо припустити, що $x_R \leq x_R$. Обравши значення d таке, що $d \equiv x_R$ права частина цього твердження стає дійсною і це робить дійсним все твердження. Таким чином дійсним стає твердження $\nexists x_L \in X_L : x \leq x_L \wedge \nexists x_R \in X_R : x_R \leq x$, з чого витікає дійсність теореми 1.2.1.

Як наслідок цієї теореми можна довести, що дійсним є твердження $x = x$. Доведенням цього твердження є визначення відношення «=».

Доведення цієї простої теореми може добре проілюструвати загальний принцип доведення тверджень про сюрреальні числа методом індукції. Цей метод дуже часто використовується в досліджені властивостей сюрреальних чисел через те, що він є рекурсивним, що дуже добре співвідноситься з принципом побудови сюрреальних чисел.

Аналогічно до теореми 1.2.1 доводиться наступна теорема.

Теорема 1.2.2 Якщо A, A_0, B, та B_0 множини сюрреальних чисел такі, що $\forall a \in A\ \exists a_0 \in A_0 : a \leq a_0 \wedge \forall b_0 \in B_0\ \exists b \in B : b \leq b_0$, то $\{A|B\} \leq \{A_0|B_0\}$.

Теорема 1.2.3 Якщо $A = A_0$ та $B = B_0$, тоді $\{A|B\} = \{A_0|B_0\}$.

Звернемо увагу, що знак «=» між множинами у визначені цієї теореми означає, що дві множини складаються з однакових елементів. Тоді теорема 1.2.3 доводиться як наслідок з теореми 1.2.2, та визначення відношення «=» для сюрреальних чисел.

Важливою для подальшого дослідження сюрреальних чисел є наступна теорема.

Теорема 1.2.4 Сюрреальне число строго більше будь якого елемента його лівої множини, та строго менше будь якого елемента його правої множини, іншими словами $\forall a \in A : a < \{A|B\}$ та $\forall b \in B : \{A|B\} < b$. Теж саме можна записати так: A < {A|B} ∧ {A|B} < B, де «<» означає , що всі елементи множини менше ніж число, або число менше ніж всі елементи множини.

З вище наведених теорем можна довести наступні властивості відношення «≤»:

- *Теорема 1.2.5 (транзитивний закон)* Якщо $x \leq y$ та $y \leq z$, тоді x ≤ z
- *Теорема 1.2.6* Якщо для сюрреальних чисел x та y x ≰ y тоді y ≤ x
- *Теорема 1.2.7* Якщо для сюрреальних чисел x та y $x < y \wedge y < z \Rightarrow x < z$

Теорема 1.2.8 В сюрреальному числі $x = \{X_L|X_R\}$ можна прибрати будь яке число ξ з X_L без зміни значення числа x, якщо множина X_L містить елемент

більший за ξ. Так само можна прибрати будь яке число η з X_R, не змінюючи значення числа x, якщо множина X_R містить елемент менший за η.

Наслідок теореми 1.2.8 Якщо множина A містить найбільший елемент $A_{\max}$, а множина B містить найменший елемент $B_{\min}$, тоді дійсне твердження, що число $\{A|B\} = \{A_{\max}|B_{\min}\}$.

Теорема 1.2.9 Якщо сюрреальне число $x = \{X_L|X_R\}$ більше ніж будь яке число в множині A, та менше ніж будь яке число в множині B, тоді дійсним є твердження, що $x = \{X_L \cup A|X_R \cup B\}$.

Наступною важливою концепцією в побудові множини сюрреальних чисел є *теорема спрощення*, та поняття дня народження числа. В попередньому пункті було розглянуто декілька випадків коли при створені нового «покоління» сюрреальних чисел з попередньо створенних ми отримували різні форми одного й того самого числа. Використовуючи аксіому 2 було проілюстровано, що $\{-1|1\}$ має значення 0, тобто це число має те саме значення, що і число $\{\emptyset|\emptyset\}$, Хоча неможна стверджувати, що це два ідентичних числа через те, що множини, які визначають ці числа є повністю різними. Повертаючись до визначення сюрреального числа, як деякої гри, що визначаетьс двома множинами ігор, що створені попередньо, тобто є попереднім станом даної гри, якщо конструювати її від стану «кінець гри» (що позначається $\{\emptyset|\emptyset\}$). Тоді можливо змінити саму гру, привевши її до наступного стану, наприклад, додавши до Червоно-синього хакенбушу декілька нових ліній, отримати нову гру, якій відповідає новому, раніше не створеному числу, яке створено з попередньо відомих, тобто з попереднього стану гри. Проте, якщо додавати певну кількість ліній певного кольору, розмістивши їх в певній конфігурації, можна не дивлячись на змінену гру, залишити сталою перевагу кожного гравця, отже не змінити значення гри, і це значення буде визначатися найпершим станом гри, що існував до рівносильної зміни переваги кожного з гравців. Формалізація цього принципу для чисел і

приводить нас до визначення теореми спрощення, проте для її формулювання буде корисним ввести точну міру того, яке з чисел було створено раніше.

Будемо називати «днем народження» числа x, номер ітерації, на якій це число було створено, тобто скільки чисел необхідно створити попередньо, щоб отримати дане число. Починати рахунок будемо з числа $\{\emptyset|\emptyset\}$, бо це перше число, яке може бути створено. Вважатимемо, що ітерація на якій створено це число має номер 0. Часто розглядаючи сюрреальні числа кажуть, що число було створено «в день» з певним номером, тому наше визначення номеру ітерації для числа $\{\emptyset|\emptyset\}$ можна сказати так: число $\{\emptyset|\emptyset\}$ було створено в день 0. Таким чином числа $\{0|\emptyset\}$ та $\{\emptyset|0\}$, що мають значення 1 та -1 відповідно, були створені в день 1. Далі приводяться дві теореми, які формалізують концепцію «дня народження» та знаходження найпростішого числа, значенню якого відповідає значення даного.

Теорема 1.2.10 Якщо в день з номером m де $m \in \mathbb{N}$, існують наступні різні сюрреальні числа: $x1 < x2 < x3 < \ldots < xn$, то всі нові створені числа на наступний день з номером $m+1$ можуть бути представлені наступною послідовністю: $\{|x1\}, \{x1|x2\}, \{x2|x3\}, \ldots, \{xn-1|xn\}, \{xn|\}$. Тоді після дня $m+1$ порядок відомих нам різних сюрреальних чисел буде таким: $\{|x1\} < x1 < \{x1|x2\} < x2 < \{x2|x3\} < x3 < \ldots < \{xn-1|xn\} < xn < \{xn|\}$

Теорема 1.2.11 (Теорема спрощення) Якщо дано число $y = \{Y_L|Y_R\}$, та число x, найперше створене з таких що, $\forall y_L \in Y_L: y_L < x$ та $\forall y_R \in Y_L: x < y_R$, тобто число, що має мінімальний «день народження», з тих, що відповідають висунутим вимогам, то можна казати, що $x = y$ (числа мають однакове значення, проте не обов'язково ідентичні).

Вже було описано досить багато властивостей сюрреальних чисел, проте найважливішою властивістю для їх практичного використання є те, що множина сюрреальних чисел включає в себе множину дійсних чисел $\mathbb{R}$, і доповнює її числами, що неможливо побудувати в $\mathbb{R}$. Тоді має існувати можливість ставити у

відповідність елементи множини $\mathbb{R}$ елементам множини $\mathbb{S}$, а точніше лише деякої її частиніи яку ми назвемо $\mathbb{S}_{\mathbb{R}}$. До цього ми вже співвідносили деякі сюрреальні числа їх дійсним відповідникам, користуючись відношенням «менше або дорівнює». Так ми проілюстрували, що $\{\emptyset|\emptyset\} = 0, \{0|\emptyset\} = 1, \{0|1\} = \frac{1}{2}$, та інші. Проте досі не було введене чітке правило знаходження сюрреального числа, яке б відповідало довільному дійсному значенню.

Для формалізації такого правила визначимо функцію $\delta(x)$, яка дійсному числу $x_{\mathbb{R}} \in \mathbb{R}$ ставить у відповідність сюрреальне число $x_{\mathbb{S}} \in \mathbb{S}$ і визначається так:

$$\delta(x) = \begin{cases} \{\emptyset|\emptyset\}, if\ x = 0 \\ \{\delta(x-1)|\emptyset\}, if\ |x| \in \mathbb{N}\ \wedge x > 0 \\ \{\emptyset|\delta(x\ +\ 1)\}, if\ |x| \in \mathbb{N}\ \wedge x < 0 \\ \left\{\delta\left(\frac{j-1}{2^k}\right)\middle|\delta\left(\frac{j+1}{2^k}\right)\right\}, if\ x = \frac{j}{2^k}, j \in \mathbb{Z}, k \in \mathbb{N} \end{cases}$$

Проте, виходячи з такого визначення функції $\delta(x)$ стає зрозуміло, що її область визначення обмежена двійково-раціональними числами, тобто числами виду $\frac{j}{2^k}, j \in \mathbb{Z}, k \in \mathbb{N} \cup \{0\}$, що представляє всю множину цілих чисел, та раціональні числа, знаменник яких є степінню числа 2. Приведення до сюрреальної форми інших дійсних чисел буде розглянуте далі, і буде доведено, що дійсне твердження $\mathbb{R} \subset \mathbb{S}$.

Операція додавання.

Перед тим, як дати визначення операції додавання для сюрреальних чисел необхідно ввести декілька позначень, які будуть використані в подальшому визначені та дослідженні операції додавання.

Будемо вважати, що вираз $n + S$, де n – число, а S – числова множина, означає додавання числа n до кожного елемента множини S. Результатом цієї дії буде множина $\acute{S} = \{s_0 + n,\ s_1 + n,\ s_2 + n \ldots\},\quad s_i \in S$. Аналогічно вирази $n - S$ та $S - n$.

Як приклад можна привести наступні випадки, з використанням звичайних дійсних чисел:

- $6 + \{3;\ 5;\ 8\} = \{9;\ 11;\ 14\}$
- $6 - \{3;\ 5;\ 8\} = \{3;\ 1; -2\}$
- $\{3;\ 5;\ 8\} - 6 = \{-3; -1;\ 2\}$

Важливо зазначити, що коли в ролі множини S виступає порожня множина $\emptyset$ результатом будь якої дії буде порожня множина $\emptyset$.

Вважатимемо, що вираз $S + T$ де S та T – числові множини, означає суму кожного елемента множини S з кожним елементом множини T. Як приклад розглянемо наступний вираз:

$$\{10;\ 20\} + \{3;\ 5;\ 8\} = \{13;\ 15;\ 18;\ 23;\ 25;\ 28\}$$

Тепер можна дати визначення додавання двох сюрреальних чисел. Сумою двох сюрреальних чисел a та b є сюрреальне число виду $\{A_L + b;\ a + B_L | A_R + b;\ a + B_R\}$, та можемо записати це так: $a + b = \{A_L + b;\ a + B_L | A_R + b;\ a + B_R\}$.

Визначення операції додавання є рекурсивним, тобто нове число визначається через попередньо створені числа.

Як приклад розглянемо таку суму $1+\frac{1}{2}$, де $1 \equiv \{0|\emptyset\}$, $\frac{1}{2} \equiv \{0|1\}$. Отже відповідно із попередньо приведеним визначенням додавання сюрреальних чисел отримаємо $1+\frac{1}{2}=\left\{0+\frac{1}{2}, 1+0\middle|\emptyset+\frac{1}{2}, 1+1\right\}=\left\{0+\frac{1}{2}, 1+0\middle|1+1\right\}$. Таким чином отримаємо число, що визначається іншими сумами сюрреальних чисел.

Тепер необхідно обчислити суму $0+\frac{1}{2}$. Так як $0 \equiv \{\emptyset|\emptyset\}$, з визначення додавання отримаємо наступне число

$$0+\frac{1}{2}=\left\{\emptyset+\frac{1}{2}, 0+0\middle|\emptyset+\frac{1}{2}, 0+1\right\}=\{0+0|0+1\}$$

І знову отримане число визначається сумами інших чисел, створених раніше, за шукане число, що повністю відповідає рекурсивній природі як операції додавання, так і рекурсивній природі безпосередньо сюрреальних чисел. Далі знаходячи відповідні суми ми отримаємо такі результат:

$$0+0=\{\emptyset+0,0+\emptyset|\emptyset+0,0+\emptyset\}=\{\emptyset|\emptyset\}=0$$

$$0+1=\{\emptyset+1,0+0|\emptyset+1,0+\emptyset\}=\{0|\emptyset\}=1$$

Таким чином отримаємо, що $0+\frac{1}{2}=\{0+0|0+1\}=\{0|1\}=\frac{1}{2}$.

Тепер знайшовши суму $1+1=\{0+1,1+0|\emptyset+1,0+\emptyset\}=\{1|\emptyset\}=2$ можна отримати, значення суми $1+\frac{1}{2}=\{1|2\}=1\frac{1}{2}$.

Далі буде наведено декілька теорем, щодо властивостей додавання сюрреальних чисел.

Теорема 1.2.12 Якщо для сюрреальних чисел $x, \acute{x}, y$ та $\acute{y}$ дійсні такі твердження: $x \leq \acute{x}$ та $\acute{y} \leq y$, то дійсне твердження $x+y \leq \acute{x}+\acute{y}$.

Теорема 1.2.13 Якщо для сюрреальних чисел $x, \acute{x}, y$ та $\acute{y}$ дійсні такі твердження: $x+y \geq \acute{x}+\acute{y}$ та $y \leq \acute{y}$, то дійсне твердження $x \geq \acute{x}$.

Наслідок з теореми 1.2.12 Якщо для сюрреальних чисел $x, \acute{x}, y$ та $\acute{y}$ дійсні такі твердження: $x=\acute{x}$ та $\acute{y}=y$, то дійсне твердження

$$x + y = \acute{x} + \acute{y}$$

Теорема 1.2.14 Якщо для сюрреальних чисел $x, \acute{x}, y$ та $\acute{y}$ дійсні такі твердження: $x < \acute{x}$ та $\acute{y} < y$, то дійсне твердження

$$x + y < \acute{x} + \acute{y}$$

Три вище приведених теореми доводяться методом індукції, що був розглянутий раніше. Тепер використовуючи ці теореми, щодо властивостей операції додавання сюрреальних чисел, можна довести важливу властивість додавання відносно відповідності аксіомі 1.

Теорема 1.2.15 Якщо два числа $a \in \mathbb{S}$ та $b \in \mathbb{S}$, то їх сума $a + b = \{A_L + b; a + B_L | A_R + b; a + B_R\}$ відповідає аксіомі 2.

Досліджуючи операцію додавання сюрреальних чисел можна довести, що для неї дійсні властивості, які має додавання дійсних чисел, та, в більш загальному випадку, операція додавання сюрреальних чисел відповідає операції додавання дійсних чисел. Це є дуже важливим фактом для побудови відображення множини дійсних чисел $\mathbb{R}$ на множину сюрреальних чисел $\mathbb{S}$. Для цього необхідно послідовно довести два твердження:

1. Множина сюрреальних чисел відносно операції додавання, тобто $(\mathbb{S}, +)$ є комутативною, або, що теж саме, Абелевою групою.
2. Функція $\delta(x)$ є гомоморфним відображенням $(\mathbb{R}, +_r)$ у $(\mathbb{S}, +_s)$, де $+_r$ позначає операцію додавання визначену на множині дійсних чисел, а $+_s$ позначає операцію додавання визначену на множині сюрреальних чисел.

Для того, щоб розглянути перше з цих тверджень, спочатку розглянемо поняття групи. Якщо э деяка множина X, та алгебраїчна операція *, що визначена для членів множини X, можна казати, що $(X, *)$ є групою, тоді, і тільки тоді, якщо дійсні наступні твердження:

- Асоціативний закон, який означає, що для будь яких елементів множини X, $a \in X$, $\mathrm{b} \in X$ та $c \in X$ дійсна рівність $(a * b) * c = a * (b * c)$.
- Існує нейтральний елемент множини X, $z \in X$, такий, що для нього дійсна рівність $a * z = z * a = a$.
- Для кожного елемента множини X, $a \in X$, існує протилежний елемент множини X, $-a \in X$, такий що дійсна рівність $a * -a = z$

Якщо крім цих тверджень, для (X,∗) дійсний також і комутативний закон, тобто дійсна рівність $\mathrm{a} * \mathrm{b} = \mathrm{b} * \mathrm{a}$, то така група є комутативна, або, що теж саме, Абелева.

Наступні теореми доводять, що множина сюрреальних чисел є Абелевою групою відносно операції додавання.

Теорема 1.2.16 0 є нейтральним елементом множини сюрреальних чисел, відносно операції додавання.

Доводиться ця теорема досить легко, використовуючи визначення операції додавання $0 + x = \{\emptyset + x, 0 + X_L \mid \emptyset + x, 0 + X_R\} = \{0 + X_L \mid 0 + X_R\}$, що дорівнює x, якщо теорема 2.16 дійсна для батьківських елементів x. Далі це доводиться за індукцією. Таким чином ми отримаємо $0 + x = x$, і аналогічно можемо довести, що $x + 0 = x$.

Теорема 1.2.17 (асоціативний закон для операції додавання) Якщо є три сюрреальних числа $x \in \mathbb{S}$, $y \in \mathbb{S}$ та $z \in \mathbb{S}$ то для них дійсна наступна рівність: $(x + y) + z = x + (y + z)$.

Ця теорема доводиться порівнянням результатів операції додавання, отриманих з правої та лівої частин нерівності:

$$(x + y) + z = \{(x + y)_L + z, (x + y) + Z_L \,|\, (x + y)_R + z, (x + y) + Z_R\}$$

$$= \{(X_L + y) + z, (x + Y_L) + z, (x + y) + Z_L \,|\, (X_R + y) + z, (x + Y_R) + z, (x + y) + Z_R\}.$$

$$x + (y + z) = \{X_L + (y + z), x + (y + z)_L \,|\, X_R + (y + z), x + (y + z)_R\}$$

$$= \{X_L + (y + z), x + (Y_L + z), x + (y + Z_L) |\, X_R + (y + z), x + (Y_R + z), x + (y + Z_R)\}$$

Результати виконаня додавання в лівій і в правій частинах рівності будуть рівними, якщо теорема 1.2.17 дійсна для батьківських елементів розглянутих чисел.

Теорема 1.2.18 Якщо є сюрреальне число $x \in \mathbb{S}$, то для нього обов'язково існує протилежний, відносно операції додавання, елемент множини сюрреальних чисел $-x \in \mathbb{S}$, таких, що $x + (-x) = 0$, і визначається як $-x = \{-X_L | -X_R\}$.

Надалі для скорочення запису вираз $a + (-b)$, тобто «до сюрреального числа a, додати сюрреальне число, що є протилежним сюрреальному числу b», будемо записувати у формі $a - b$.

Теореами 1.2.19 (комутативний закон для операції додавання) Якщо є два сюрреальних числа $x \in \mathbb{S}$ та $y \in \mathbb{S}$, то для них дійсна наступна рівність: $x + y = y + x$

Виходячи з визначення операції додавання, ця теорема є дійсною для будь яких $x \in \mathbb{S}$ та $y \in \mathbb{S}$, якщо вона дійсна для їх батьківських елементів, і таким чином доведення зводиться до доведення рівності $0 + x = x + 0$, яка випливає з вже розглянутої теореми 1.2.16.

Теореми 1.2.16, 1.2.17, 1.2.18 та 1.2.19 доводять твердження, що множина сюрреальних чисел утворює комутативну групу відносно операції додавання. Це означає, що операція додавання сюрреальних чисел має ті ж самі алгебраїчні

властивості, що і операція додавання дійсних чисел, бо множина дійсних чисел також утворює комутативну групу відносно операції додавання.

Тепер розглянемо друге твердження, яке має бути дійсним для доведення відповідності операції додавання сюрреальних чисел операції додавання дійсних чисел. Це твердження говорить про те, що функція $\delta(x)$ є гомоморфним відображенням $(\mathbb{R}, +_r)$ у $(\mathbb{S}, +_s)$ де $+_r$ позначає операцію додавання визначену на множині дійсних чисел, а $+_s$ позначає операцію додавання визначену на множині сюрреальних чисел. Теж саме твердження можна записати як рівність наступного вигляду: $\delta(a + b) = \delta(a) + \delta(b)$. Важливість такої властивості полягає в тому, що функція $\delta(x)$ використовується для відображення деякого дійсного числа на відповідне йому сюрреальне число. Таким чином, для $\delta(2 + 3)$, ми хочемо отримати сюрреальне число, яке представляє суму сюрреальних відповідників дійсних чисел 2 та 3.

Теорема 1.2.20 Якщо a та b дійсні числа, то для них дійсною є рівність $\delta(a + b) = \delta(a) + \delta(b)$.

Доведення цієї теореми не наводиться у цій роботі, через досить великий об’єм, та відсутність практичного значення самого доведення в подальшому розгляді.

Операція множення

Добутком двох сюрреальних чисел a та b э інше сюрреальне число, яке позначається наступним чином: $a * b = \{A_L b + aB_L - A_L B_L, A_R b + aB_R - A_R B_R \mid A_L b + aB_R - A_L B_R, A_R b + aB_L - A_R B_L\}$

Як і для додавання, для операції множення нам важливо довести відповідність множення сюрреальних чисел множенню дійсних чисел. Для цього мають бути дійсними два наступні твердження:

1. Множина сюрреальних чисел, без числа 0, відносно операції множення, тобто $(\mathbb{S}/0,*)$ є комутативною, або, що теж саме, Абелевою групою. Число 0 відіграє особливу роль для операції множення сюрреальних чисел (як і в множені дійсних чисел)
2. Функція $\delta(x)$ є гомоморфним відображенням $(\mathbb{R},*_r)$ у $(\mathbb{S},*_s)$, де $*_r$ позначає операцію множення визначену на множині дійсних чисел, а $*_s$ позначає операцію множення визначену на множині сюрреальних чисел.

Теорема 1.2.21 $0 * x = x * 0 = 0$

Доведення цієї теореми проводиться підстановкою числа нуль згідно з визначенням операції множення. При такій підстановці на будь яке місце (правий чи лівий множник), так не залежно від x, в результаті множення буде отримано 0.

Доведення всіх наступних теорем проводиться тимим самими методами, що і доведення аналогічних теорем для операції додавання сюрреальних чисел.

Теорема 1.2.22 Для операції множення сюрреальних чисел нейтральним елементом є число 1. Таким чином : $1 \times x = x \times 1 = x$

Теорема 1.2.23(комутативний закон для операції множення) Якщо є два сюрреальних числа x та y, то для них дійсна рівність $xy = yx$

Теорема 1.2.24(ассоціативний закон для операції множення) Якщо є три сюрреальних числа x, y та z то для них дійсна рівність $(xy)z = x(yz)$

Теорема 1.2.25(дистрибутивний закон) Якщо є три сюрреальних числа x, y та z то для них дійсна рівність $(x + y)z = xz + yz$

Теорема 1.2.26 Відносно операції множення, кожне сюрреальне число x, окрім числа 0, має зворотній елемент $\frac{1}{x}$, такий, що $x * \frac{1}{x} = \frac{1}{x} * x = 1$

Вище приведені теореми доводять твердження, що множина $\mathbb{S}/0$ утворює Абелеву групу відносно операції множення.

Теорема 1.2.20 Якщо a та b дійсні числа, то для них дійсною є рівність $\delta(a * b) = \delta(a) * \delta(b)$.

Розширення множини сюрреальних чисел

До цього моменту вже були досліджені всі основні властивості сюрреальних чисел, та арифметичних дій із ними, проте самі сюрреальні числа мають однозначне відображення (функція $\delta(x)$) тільки з цілими числами та дробами, знаменник яких є стелінню числа два. Також можливо представити у сюрреальній формі будь яке дійсне число, яке не можливо записати як дріб знаменником якого є певна степінь двійки, проте для цього необхідно буде побудувати ліву та праву множину цього числа з нескінченно кількості наближень з недостачею та з надлишком, відповідно. Проте далі буде проілюстровано, що множина сюрреальних чисел не тільки повністю включає в себе множину дійсних чисел, але й доповнює її, узагальнюючи дійсні, та нескінченні порядкові числа.

Для початку покажемо, як визначається множина цілих чисел у термінах сюрреальних чисел. Далі будемо множину цілих чисел позначати $\mathbb{Z}$, а підмножину множини сюрреальних чисел, яка включає в себе всі сюрреальні відповідники цілих чисел будемо позначати $\mathbb{S}_{\mathbb{Z}}$.

Таким чином ми отримаємо наступне визначення підмножини $\mathbb{S}_{\mathbb{Z}}$:

- $0 \in \mathbb{Z} \Rightarrow \{\emptyset|\emptyset\} \in \mathbb{S}_{\mathbb{Z}}$
- $n \in \mathbb{Z} \Rightarrow \begin{cases} \{n|\emptyset\} \in \mathbb{S}_{\mathbb{Z}} \\ \{\emptyset|n\} \in \mathbb{S}_{\mathbb{Z}} \end{cases}$

Отримана множина $\mathbb{S}_{\mathbb{Z}}$ є множиною сюрреальних чисел, і відповідно до аксіоми 1, можливо побудувати сюрреальне число наступного виду:

$$\{\mathbb{S}_{\mathbb{Z}}|\emptyset\}$$

Отримане повністю відповідає аксіомі 1, так як в лівій множині цього числа (множина $\mathbb{S}_{\mathbb{Z}}$) не існує жодного елементу, що був би більший або дорівнював будь якому елементу з правої множини цього числа, через те, що права множина числа є пустою множиною.

Відповідно до теореми 1.2.4 отримане число, буде більшим за всі елементи його лівої множини, тобто більше будь якого елемента множини $\mathbb{S}_{\mathbb{Z}}$. З визначення множини $\mathbb{S}_{\mathbb{Z}}$, ми бачимо, що вона містить в собі всі сюрреальні числа, значення яких є цілими числами, і таким чином, отримаємо, що число $\{\mathbb{S}_{\mathbb{Z}}|\emptyset\}$ є більшим ніж будь яке ціле число. Тобто значенням отриманого числа є нескінченність. Надалі будемо позначати це число грецькою літерою ω. Так само позначається ординал, або іншими словами нескінченне порядкове число, що за своїми властивостями відповідає числу $\{\mathbb{S}_{\mathbb{Z}}|\emptyset\}$. Надалі ми не будемо заглиблюватися в теорію порядкових чисел, проте слід сказати, що за допомогою сюрреальних чисел можливо представити усі порядкові числа.

Використовуючи розглянуті вище властивості сюрреальних чисел, та дії з ними, можна побудувати наступні числа, які, так само як і число ω, не можуть бути побудовані використовуючи дійсні числа. Наприклад $\omega + 1 = \{\omega|\emptyset\}$, або 2ω, або ω^{ω}.

За анлогічним принципом можливо побудувати нескінченно мале число, яке буде більше за 0, проте менше ніж будь яке додатне дійсне число: $\varepsilon = \{0|\mathbb{R}_{+}\}$.

Можливість оперувати нескінченно великими та нескінченно малими величинами, а головне відрізняти різні нескінченно великі чи малі значення, проводити з ними арифметичні операції, є дуже важливою для теорії ігор і аналізу наявності переможної стратегії в певній грі, бо не маючи цього інструмента не можливо було б сказати, який з гравців червоно-синього хакенбушу, наприклад, отримає перемогу, коли на ігровому полі була б нескінченна кількість ліній різних кольорів, проте використовуючи математичний апарат, який надають сюрреальні числа, роблять можливим аналізувати який з гравців початково має виграшну стратегію, при цьому абстрагуючись від того, що для досягнення перемоги треба зробити нескінченну кількість ходів.

1.3 Обґрунтування вибору мови програмування

Метою дипломного проекту є розробка настільної програми для операційної системи Microsoft Windows. Вибір цієї ОС обумовлений передусім тим, вона займає близько 79% ринку для застосунків такого типу у світі, а в Україні її частка сягає 86%.

Серед переваг Windows можна виділити наступні:

— під керуванням цієї ОС працюють неймовірна кількість пристроїв, практично будь-яка програма або її аналоги;
— апаратна та програмна сумісність;
— повне використання апаратних ресурсів;
— багатозадачність;
— захищений режим;
— підтримка та легкість встановлення;
— обмін даними між застосунками;
— функціональність та зручний та інтуїтивно зрозумілий інтерфейс;
— легке відновлення видаленої інформації.

Програмне забезпечення, виконане в рамках даного дипломного проекту, розроблене за допомогою програмної платформи .NET Framework від компанії Microsoft — технології, що розрахована на роботу під ОС Microsoft Windows. Ця платформа представляє собою всебічну і узгоджену модель програмування для побудови застосунків, що володіють привабливим інтерфейсом користувача, прозорими і безпечними засобами зв'язку, а також можливістю створення різноманітних бізнес-процесів.

Дана платформа представлена у декількох версіях, для розроблюваної системи мінімально необхідною версією була обрана 4.0, випущена у 2010 році, що підходить навіть для Windows XP.

Мінімальні рекомендовані апаратні вимоги для .NET Framework 4.0:

1) процесор Pentium з тактовою частотою 1 ГГц;

2) 512 МБ оперативної пам’яті;

3) мінімальне місце на диску:

— x86 – 850 МБ;

— x64 – 2 ГБ.

Microsoft .NET Framework 4 працює разом зі своїми попередніми версіями. Застосунки, засновані на попередніх версіях .NET Framework, будуть продовжувати виконуватися на платформі, для якої вони призначені за замовчуванням.

Платформа містить наступні нові можливості і удосконалення [3], які використовувалися під час розробки програми:

— вдосконалення в CLR (Common Language Runtime) і BCL (Base Class Library);

— нововведення в мові C#, наприклад лямбда-оператори, неявні продовження рядків, а також іменовані і необов'язкові параметри.

— удосконалення в доступі до даних і моделюванні;

— вдосконалення в Windows Presentation Foundation (WPF).

Для розробки програмного забезпечення під платформу .NET була обрана така високорівнева мова програмування, як C#. Мова C# розроблена під егідою Microsoft, вона є об’єктно-орієнтованою з безпечною системою типізації та може бути використана для створення кроссплатформених застосунків.

На даний момент мова програмування C# дуже стрімко розвивається. Зокрема, вона налічує потужну кількість бібліотек різної функціальності, які працюють з максимальною швидкодією, та характеризується безліччю переваг, як для програміста з точки зору зручності написання коду, так і з точки зору ефективності її роботи.

Так, прийнято виділяти наступні найважливіші переваги, які надає дана мова:

— автоматичне керування пам'яттю;

— можливість створення багатопотокових застосунків;

— підтримка узагальнень;

— підтримка лямбда виразів та замикань, а також функціонального програмування;
— розширені можливості обробки виняткових ситуацій;
— механізм абстракцій, наслідування, поліморфізм, інкапсуляція;
— вказівники на функції-члени класів, делегати;
— велика стандартна бібліотека класів, зокрема для роботи з файлами та колекціями;
— JIT-компіляція;
— коментарі у форматі XML.

В якості технології для розробки системи була обрана така система побудови клієнтських застосунків, як Windows Forms. Windows Forms - інтерфейс програмування додатків (API), що відповідає за графічний інтерфейс користувача і є частиною Microsoft .NET Framework.

Даний інтерфейс спрощує доступ до елементів інтерфейсу Microsoft Windows за рахунок створення обгортки для існуючого Win32 API в керованому коді. Причому керований код - класи, що реалізують API для Windows Forms, що не залежать від мови розробки. Тобто програміст однаково може використовувати Windows Forms як при написанні програмного забезпечення на C #, C ++, так і на VB.Net, J# тощо.

2. Спеціальна частина

2.1 Опис алгоритму створення програмного засобу

Комп’ютерне представлення сюрреального числа за визначення Конвея обумовлює високу обчислювальну складність як алгоритму приведення дійсного числа до сюрреального представлення, так і алгоритмів виконання операцій над сюрреальними числами. Окрім цього, через конструкцію сюрреальних чисел, сюрреальні представлення дійсних чисел, що не належать до множини двійково-раціональних чисел є нескінченними, що робить комп’ютерну обробку таких даних майже неможливою. Для вирішення цих проблем було вирішено використовувати квазідвійкове представлення сюрреального числа. Таке представлення дозволяє значно оптимізувати алгоритми побудови сюрреального числа, та виконання операцій, а також, хоча і не дозволяє скінченно представити число, яке не є двійково-раціональним, але значно спрощує приблизне представлення таких чисел при комп’ютерній обробці.

Квазідвійкове представлення сюрреальних чисел є певною моделлю сюрреального числа, визначеного Конвеєм, тобто квазідвійкове представлення можна сприймати як набір об’єктів, що задовольняють одинадцяти аксіомам (або тринадцяти, якщо вимоги замкненості суми і добутку сюрреального числа вважати окремими аксіомами), яким задовольняють сюрреальні числа.

Розглядаючи формальну теорію $S = \langle A, P, D, V \rangle$, де A – алфавіт, P – основні правила та визначення, D – дії з сюрреальними числами, V – властивості чисел та дій над ними, можна зробити висновок, що для задання, зокрема дійсної підмножини (а взаягалі кажучи і не тільки її) всесвіту сюрреальних чисел в алфавіті достатньо мати три символи, тобто $A = \{Z, 1, 0\}$. Набори символів взятих з A будемо називати квазідвійковим представленням сюрреальних чисел. Якщо набір не містить символів,

крім символа Z, то він відіграє роль нуля. Набори символів, що відповідають сюрреальним числам, відмінним від нуля, можуть не містити в своєму записі символа Z. Будь-який набір символів, що відповідає сюрреальному числу x розглядається як протокол пошуку цього числа x на числовій прямій.

Набір символів може бути скінченним, наприклад 01, 1010, нескінченним, наприклад 11011…, 1(010), більш ніж нескінченним, наприклад (0)0, де дужки () означають нескінченне повторення впорядкованого набору символів, який знаходиться в них. Індуктивна побудова (народження) сюрреальних чисел передбачає наступне. Найпершим (початковим в протоколі) є число нуль (символ Z). Якщо число x більше нуля (тобто є додатнім) то наступний символ його запису 1. Запис $Z1$ (або просто 1) означає число 1. Якщо сюрреальне число більше 1, то наступний символ його запису буде також 1. Запис 11 відповідає числу 2, і так далі для підмножини натуральних чисел всесвіту сюрреальних чисел. Якщо сюрреальне число від'ємне, то наступним символом в його записі після Z буде символ 0. Запис $Z0$ (або просто 0) буде відповідати числу -1. Якщо сюрреальне число менше ніж -1, то наступним символом в його записі буде також 0. Запис 00 відповідає числу -2, і так далі для всіх цілих чисел. Таким чином, якщо вважати символ Z позначення початкової точки на числовій прямій, з якої починається пошук деякого сюрреального числа, то символ 1 буде позначати рух праворуч вздовж числової прямої на одиничний відрізок, а символ 0 – рух ліворуч на одиничний відрізок. Після першої зміни символа 1 на 0 або 0 на 1 в записі (протоколі пошуку) сюрреального числа, відбувається зміна напрямку і «масштабу» руху. Після першої зміни символа кожний крок пошуку зменшується вдвічі.

Як було сказано раніше, вказаний процес пошуку може бути скінченним або нескінченним для підмножини дійсних чисел. Будь-який квазідвійковий запис (відповідно протокол пошуку) відповідає деякому сюрреальному числу, причому різні записи відповідають різним сюрреальним числам.

В всесвіті сюрреальних чисел існує кілька порядкових відношень (<,=,>) і (символи «˅» - бути раніше, «˄» - бути пізніше).

Відношення порядку (<,=,>) в квазідвійковому представленні сюрреальних чисел діє лексикографічно. Якщо перший з неоднакових символі квазідвійкового запису сюрреального числа a буде 1, а у числа b буде 0, і числа a і b записуються однаковою кількістю символів, то $a > b$. Якщо число a має меншу кількість символів запису ніж b: $a = a_1 a_2 \dots a_n$ $b = b_1 b_2 \dots b_n b_{n+1}$ і $a_k = b_k$ $\forall k = \overline{1,n}$, і $b_{n+1} = 0$, то $a > b$.

Будемо вважати, що для сюрреальних чисел a і b має місце відношення $a \check{} b$, якщо ці числа мають наступні квазідвійкові представлення: $a = a_1 a_2 \dots a_n$ $b = a_1 a_2 \dots a_n a_{n+1}$ $\forall a_i \in \{0,1\}$ $(i = \overline{1,n})$. Очевидно, що в цьому разі $b \hat{} a$.

Поняття двійково-раціонального числа в строгому сенсі пов'язане з поняттям класичної двійкової системи числення. В широкому сенсі під двійково-раціональними числами розуміють раціональні числа, знаменник яких дорівнює степені двійки. Очевидно, що всім двійково-раціональним числам в широкому сенсі відповідають скінченні набори одиниць і нулів в квазідвійковому представленні сюрреальних чисел.

Для загальних визначень операції додавання і множення за Конвеєм сюрреальних чисел важливу роль відіграє поняття зрізу і частинних зрізів сюрреального числа. Вказане пояснюється тим, що Конвей задає операції над сюрреальними числами керуючись принципами почерговості і простоти. Тому правила дії над числами визначаються, як і самі числа, індуктивно: спочатку для більш «ранніх», потім для більш «пізніх». В якості результату вибирається найбільш раннє можливе число.

Зрізом для деякого сюрреального числа a (зріз позначають символом $|a|$) називають множину всіх сюрреальних чисел x більш ранніх ніж a, тобто $|a| = \{x | x \check{} a\}$.

До частинних зрізів відносять нижній зріз $\left|\underline{a}\right| = \{x|x \in |a|, x < a\}$ і верхній зріз $|\overline{a}| = \{x|x \in |a|, x > a\}$.

Для визначення дій з сюрреальними числами суттєву роль відіграє наступна лема.

Лема (про відокремлююче сюрреальне число) Нехай S_1 і S_2 – дві підмножини всесвіту сюрреальних чисел. Якщо $S_1 < S_2$, тобто якщо $\forall x \in S_1$ і $\forall y \in S_2\ x < y$, то існує таке найбільш раннє сюрреальне число a, що $x < a < y\ \forall x \in S_1$ та $\forall y \in S_2$.

Зауваження. Кажуть, що сюрреальне число z відокремлює множини S_1 і S_2, якщо $x < z < y\ \forall x \in S_1$ та $\forall y \in S_2$. Взагалі кажучи таких відокремлюючих чисел множин S_1 і S_2 багато, але зазначене в лемі найбільш раннє відокремлююче число a – єдине. Таке найбільш раннє відокремлююче число a позначають $\{S_1|S_2\}$.

Додавання двійково-раціональних сюрреальних чисел, заданих своїм квазідвійковим представленням.

Одне з загальних визначень суми двох сюрреальних чисел за Конвеєм, побудоване на принципах почерговості і простоти формулюється наступним чином:

$$a + b = \left\{\left\{\left|\underline{a}\right| + b\right\} \cup \left\{a + \left|\underline{b}\right|\right\}\middle|\left\{|\overline{a}| + b\right\} \cup \left\{a + \left|\overline{b}\right|\right\}\right\}$$

Додавання сюрреальних чисел відповідно принципам Конвея є досить складним і трудомістким.

Проілюструємо додавання двох сюрреальних чисел, заданих своїм квазідвійковим представленням за Конвеєм на наступному прикладі: нехай $a = 101; b = 1001;$ $|a| = \{Z, 1, 10\};\ |b| = \{Z, 1, 10, 100\};\ \left|\underline{a}\right| = \{Z, 10\};\ \left|\underline{b}\right| = \{Z, 100\};\ |\overline{a}| = \{1\};\ \left|\overline{b}\right| = \{1, 10\};$

$$a + b =$$

$$\left\{\left\{\{Z + 1001, 10 + 1001\} \cup \{101 + Z, 101 + 100\}\right\}\middle|\left\{\{1 + 1001\} \cup \{101 + 1, 101 + 10\}\right\}\right\} =$$

$$\left\{\{1001, 1011, 101, 1\}\middle|\{11001, 1101, 100\}\right\} =$$

$\{\{1,101,1001,1011\}|\{1100,1101,11001\}\} = \{1011|1100\} = 11000$ – найбільш раннє відокремлююче число.

В окремих випадках, зокрема для двійково-раціональних сюрреальних чисел можна знайти простіші, але не претендуючі на загальність схеми додавання сюрреальних чисел, заданих своїм квазідвійковим представленням. Для запропонованого прикладу одна з таких схем може бути наступною:

$$\begin{array}{r} +\dfrac{1\,\overbrace{01}}{10\,\overbrace{01}} \\ 11000 \end{array}$$

Таку схему можна пояснити наступним чином:

$$+\frac{1-\overbrace{\frac{1}{2}+\frac{1}{4}}}{1-\frac{1}{2}-\overbrace{\frac{1}{4}+\frac{1}{8}}}$$
$$1+1-\frac{1}{2}-\frac{1}{4}-\frac{1}{8}$$

Для отримання сюрреального числа протилежного сюрреальному числу a, необхідно в його квазідвійковому записі всі одиниці замінити нулями, а нулі – одиницями. Тому природнім є зведення дії віднімання сюрреальних чисел до додавання з протилежним знаком числа, яке віднімається. Наприклад: нехай $a = 1101; b = 1001$. Тоді $a - b = a + (-b)$, тобто $1101 - 1001 = 1101 + 0110$. Додавання вказаних чисел в стовпчик дає наступний результат:

$$\begin{array}{r} +\dfrac{11\,\overbrace{01}}{\overbrace{01}\,\overbrace{10}} \\ 11001 \end{array}$$

Зауваження 2. Для квазідвійкового представлення двійково-раціональних сюрреальних чисел віднімання не обов'язково має зводитися до додавання з протилежним знаком числа, що віднімається. Його можна здійснювати в стовпчик безпосередньо, користуючись, зокрема, запропонованою схемою. Для нашого прикладу вказане здійснюється наступним чином:

$$-\frac{11\overbrace{01}}{10\overbrace{01}}$$
$$11001$$

що пояснюється можливими міркуваннями:

$$-\frac{1+1-\overbrace{\frac{1}{2}+\frac{1}{4}}^{-\frac{1}{4}}}{\frac{\overbrace{1-\frac{1}{2}}^{\frac{1}{2}}-\overbrace{\frac{1}{4}+\frac{1}{8}}^{-\frac{1}{8}}}{1+1-\frac{1}{2}-\frac{1}{4}+\frac{1}{8}}}$$

Множення двійково-раціональних сюрреальних чисел, заданих своїм квазідвійковим представенням

Відповідно принципам почерговості і простоти одне з загальних визначень добутку двох сюрреальних чисел за Конвеєм має наступне формулювання: $a*b=\left\{\left\{\{|\underline{a}|*b+a*|\underline{b}|-|\underline{a}|*|\underline{b}|\}\cup\{|\overline{a}|*b+a*|\overline{b}|-|\overline{a}|*|\overline{b}|\}\right|\left\{\{|\underline{a}|*b+a*|\overline{b}|-|\underline{a}|*|\overline{b}|\}\cup\{a*|\underline{b}|+|\overline{a}|*b-|\overline{a}|*|\underline{b}|\}\right\}\right\}$

Множення сюрреальних чисел відповідно принципам Конвея здійснюється ще складніше ніж додавання, тому первісна задумка проілюструвати його реалізацію на значеннях $a=101;b=1001$ ($a*b=101*1001=100100$) була замінена більш простим прикладом

Нехай $a=101;b=01$. Очевидно, що добуток $a*b$ є від'ємним тому можна міркувати, наприклад, наступним чином: $a*b=a*(-\acute{b})=-(a*\acute{b})$, де $\acute{b}=-b=-(01)=10$. Дужки () в цьому випадку визначають послідовність дій, а не період

сюрреального числа. Для того, щоб знайти добуток $a * \acute{b}$ покроково, кожний крок може записуватись наступним чином:

1. $|a| = \{Z, 1{,}10\};\ \left|\acute{b}\right| = \{Z, 1\};\ \left|\underline{a}\right| = \{Z, 10\};\ \left|\underline{\acute{b}}\right| = \{Z\};\ |\overline{a}| = \{1\};\ \left|\overline{\acute{b}}\right| = \{1\};$
2. $\left\{\left|\underline{\mathrm{a}}\right| * \acute{\mathrm{b}} + \mathrm{a} * \left|\underline{\mathrm{b}}\right| - \left|\underline{\mathrm{a}}\right| * \left|\underline{\mathrm{b}}\right|\right\} = \{\mathrm{Z} * 10 + 101 * \mathrm{Z} - \mathrm{Z} * \mathrm{Z}, 10 * 10 + 101 * \mathrm{Z} - 10 * \mathrm{Z}\} = \{100\}$
3. $\left\{|\overline{a}| * \acute{b} + a * \left|\overline{\acute{b}}\right| - |\overline{a}| * \left|\overline{\acute{b}}\right|\right\} = \{1 * 10 + 101 * 1 - 1 * 1\} = \{100\}$
4. $\left\{\left\{\left|\underline{a}\right| * \acute{b} + a * \left|\underline{\acute{b}}\right| - \left|\underline{a}\right| * \left|\underline{\acute{b}}\right|\right\} \cup \left\{|\overline{a}| * \acute{b} + a * \left|\overline{\acute{b}}\right| - |\overline{a}| * \left|\overline{\acute{b}}\right|\right\}\right\} = \{\{100\} \cup \{100\}\} = \{100\}$
5. $\left\{\left|\underline{a}\right| * \acute{b} + a * \left|\overline{\acute{b}}\right| + \left|\underline{a}\right| * \left|\overline{\acute{b}}\right|\right\} = \{Z * 10 + 101 * 1 - Z * 1{,}10 * 10 + 101 * 1 - 10 * 1\} = \{101{,}100 + 101 - 10\} = \{101{,}10\}$
6. $\left\{a * \left|\underline{\acute{b}}\right| + |\overline{a}| * \acute{b} - |\overline{a}| * \left|\underline{\acute{b}}\right|\right\} = \{101 * Z + 1 * 10 - 1 * Z\} = \{10\}$
7. $\left\{\left\{\left|\underline{a}\right| * \acute{b} + a * \left|\overline{\acute{b}}\right| + \left|\underline{a}\right| * \left|\overline{\acute{b}}\right|\right\} \cup \left\{a * \left|\underline{\acute{b}}\right| + |\overline{a}| * \acute{b} - |\overline{a}| * \left|\underline{\acute{b}}\right|\right\}\right\} = \{\{101{,}10\} \cup \{10\}\} = \{10{,}101\}$
8. $a * \acute{b} = \{\{100\}|\{10{,}101\}\} = 1001$ – найбільш раннє відокремлююче сюрреальне число
9. Тоді $a * b = -(a * \acute{b}) = 0110.$

Запропонований приклад множення сюрреальних чисел заданих своїм квазідвійковим представленням навіть для таких простих двійково-раціональних співмножників демонструє непросту реалізацію вказаної дії за визначенням Конвея. Але це визначення є як загальним, так і фундаментальним. З іншої сторони існують окремі випадки, для яких правила множення є достатньо простими і очевидними. Зокрема добуток двох натуральних співмножників заданих своїм квазідвійковим представленням складається з одиниць, кількість яких дорівнює добутку їх кількостей в кожному з співмножників. Наприклад:

$$
\begin{array}{r}
\overbrace{111}^{3} \\
* \quad \overbrace{}^{2} \\
\overbrace{}^{6=2*3} \\
\hline
\overbrace{111111}
\end{array}
$$

Такими ж простими виявляються правила добутку натурального і цілого від’ємного співмножників, заданих своїм квазідвійковим представленням та достатньо велика низка досить простих і очевидних правил які мають місце завдяки специфічним властивостям квазідвійкового представлення двійково-раціональних сюрреальних чисел.

Виходячи з індуктивного характеру побудови не тільки сюрреальних чисел, але і дій з ними, корисними можуть стати таблиці множення, які спрощують в окремих випадках вказану операцію. Прикладом таких таблиць може бути наступна таблиця:

	1	10	11	100	101	110	…
1	1	10	11	100	101	110	
10	10	100	1	1000	1001	101	
11	11	1	1111	10	110	111	
100	100	1000	10	10000	10001	1001	
101	101	1001	110	10001	10100	11000	
110	110	101	111	1001	11000	11100	
⋮							

Таблиця 2.1.1

Для найпростіших випадків множення сюрреальних чисел (таких, як множення натуральних чисел, цілих чисел, тощо) правила отримання результатів є очевидними, досить простими.

Ділення сюрреальних чисел, заданих своїм квазідвійковим представленням

Ділення сюрреальних чисел можна зводити до множення чисельника на число обернене до знаменника, тобто: $\frac{a}{b} = a * \frac{1}{b}$.

Тоді ділення передбачає вміння знаходити число, обернене до заданого. Правило Конвея знаходження оберненого сюрреального числа передбачає ґрунтовне розуміння індуктивного підходу до визначення дії з двома сюрреальними числами, понять числової форми, зрізів, відокремлюючого числа і розуміння змісту самого правила.

Якщо числова форма сюрреального числа $b = \{b_L | b_R\}$, то правило Конвея, побудоване індуктивним способом за принципами почерговості і простоти знаходження числа $\frac{1}{b}$ формулюється наступним чином: $\frac{1}{b} = \left\{ \left\{ Z, \frac{1+(b_R-b)*\left(\frac{1}{b}\right)_L}{b_R}, \frac{1+(b_L-b)*\left(\frac{1}{b}\right)_R}{b_L} \right\} \middle| \left\{ \frac{1+(b_L-b)*\left(\frac{1}{b}\right)_L}{b_L}, \frac{1+(b_R-b)*\left(\frac{1}{b}\right)_R}{b_R} \right\} \right\}$

Де $\left\{ \left(\frac{1}{b}\right)_L \middle| \left(\frac{1}{b}\right)_R \right\}$ – числова форма сюрреального числа $\frac{1}{b}$, $b_L > 0, b_R > 0$.

Для $b = Z, \frac{1}{b}$ – не визначається, для $b < 0, \frac{1}{b} = -\left(-\frac{1}{b}\right) = -\left(\frac{1}{-b}\right)$.

Знаходження сюрреального числа, оберненого до заданого за загальним правилом Конвея можна проілюструвати на наступному прикладі.

Нехай сюрреальне число $a = 5$, тоді його числова форма має вигляд $a = \{4|\emptyset\}$

1. $\left(\frac{1}{a}\right)_L = Z,\ \frac{1}{5} = \{\{Z\}|\emptyset\}$
2. $b_L = 4;\ \left(\frac{1}{a}\right)_R = \frac{1+(1111-11111)*Z}{1111} = 100;\ \frac{1}{5} = \{\{Z\}|\{100\}\}$

3. $\left(\frac{1}{a}\right)_L = \frac{1+(1111-11111)*100}{100} = 10001;\ \frac{1}{5} = \{\{Z, 10001\}|\{100\}\}$
4. $\left(\frac{1}{a}\right)_R = \frac{1+0*10001}{1111} = 1000110;\ \frac{1}{5} = \{\{Z, 10001\}|\{100,1000110\}\}$
5. $\left(\frac{1}{a}\right)_L = \frac{1+0*1000110}{1111} = 100011001;\ \frac{1}{5} = \{\{Z, 10001,100011001\}|\{100,1000110\}\}$
6. $\left(\frac{1}{a}\right)_R = \frac{1+0*100011001}{1111} = 10001100110;\ \frac{1}{5} = \{\{Z, 10001,100011001\}|\{100,1000110,10001100110\}\}$

Гіпотетично не важко помітити появу періодичності в квазідвійковому записі шуканого сюрреального числа, тобто: $\frac{1}{5} = \{\{Z, 10001,100011001\ldots\}|\{100,1000110,10001100110\ \ldots\}\}$, і тоді виникає гіпотеза, що $\frac{1}{5} = 10(0011)$.

7. Перевіремо (доведемо) гіпотезу $\frac{1}{5} = 10(0011)$. Квазідвійковому запису сюрреального числа 10(0011) відповідає наступний нескінченний числовий ряд:

$$1 - \frac{1}{2} - \frac{1}{2^2} - \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^5} - \frac{1}{2^6} - \frac{1}{2^7} + \frac{1}{2^8} + \frac{1}{2^9} - \frac{1}{2^{10}} - \frac{1}{2^{11}} + \frac{1}{2^{12}} + \frac{1}{2^{13}} - \cdots =$$
$$1 - \frac{1}{2} - \left(\frac{1}{2^2} + \frac{1}{2^6} + \frac{1}{2^{10}} + \frac{1}{2^{14}} + \cdots\right) - \left(\frac{1}{2^3} + \frac{1}{2^7} + \frac{1}{2^{11}} + \cdots\right) + \left(\frac{1}{2^4} + \frac{1}{2^8} + \frac{1}{2^{12}} + \cdots\right) + \left(\frac{1}{2^5} + \frac{1}{2^9} + \frac{1}{2^{13}} + \cdots\right) = \frac{1}{2} - \frac{1}{2^2}\left(1 + \frac{1}{2^4} + \frac{1}{2^8} + \frac{1}{2^{12}} + \cdots\right) - \frac{1}{2^3}\left(1 + \frac{1}{2^4} + \frac{1}{2^8} + \frac{1}{2^{12}} + \cdots\right) + \frac{1}{2^4}\left(1 + \frac{1}{2^4} + \frac{1}{2^8} + \frac{1}{2^{12}} + \cdots\right) + \frac{1}{2^5}\left(1 + \frac{1}{2^4} + \frac{1}{2^8} + \frac{1}{2^{12}} + \cdots\right) =$$
$$\frac{1}{2} - \left(\frac{1}{2^2} + \frac{1}{2^3} - \frac{1}{2^4} - \frac{1}{2^5}\right) * \sum_{k=0}^{\infty} \frac{1}{2^{4k}}$$

Послідовність $\frac{1}{2^{4k}}$ де $k = 0,1,\ldots$ є нескінченною спадною геометричною прогресією, тому : $\sum_{k=0}^{\infty} \frac{1}{2^{4k}} = \frac{1}{1-\frac{1}{2^4}} = \frac{16}{15}$

Таким чином отримаємо : $\frac{1}{4} + \frac{1}{8} - \frac{1}{16} - \frac{1}{32} = \frac{9}{32};\quad \frac{1}{2} - \frac{9}{32} * \frac{16}{15} = \frac{1}{2} - \frac{3}{2*5} = \frac{1}{5}$

2.2 Опис засобів реалізації

Для програної реалізації описаного метода представлення та обробки сюрреальних чисел була розроблена структура даних SurrealNode (далі об'єкт такої структури будемо називати вузлом), що представляє собою один крок в протоколі пошуку числа. Протокол пошуку повністю представляється зв'язним списком вузлів, таким, що кожен вузол, окрім початкового в протоколі, знає про вузол, що передував йому. Для отримання квазідвійкового представлення сюрреального числа, що відповідає певному дійсному числу використовується рекурсивний алгоритм, який починаючи з вузла, що представляє число 0 (сюрреальне число $\{\emptyset|\emptyset\}$, або Z в квазідвійковому представленні). Кожен вузол знає, якому дійсному значенню відповідає крок в протоколі пошуку, який представляє цей вузол. Алгоритм порівнює це значення із значенням, представлення якого необхідно отримати, і визначає яку дію необхідно виконати далі (рух праворуч, рух ліворуч, зупинка алгоритму). Алгоритм зупиняється тоді, коли значення оброблюємого вузла дорівнює значенню, представлення якого необхідно побудувати. В іншому випадку, обирається один з напрямків руху. Ліворуч, якщо значення вузла більше ніж значення для якого необхідно побудувати представлення, або праворуч, якщо значення вузла більше ніж значення для якого необхідно побудувати представлення. Кожен рух збільшує значення вузла на $\frac{1}{2^k}$ $k = 0,1,2, \ldots$, де k – номер кроку протоколу пошуку починаючи з першої зміни напрямку. До зміни напрямку $k = 0$. На кроці, де вперше змінюється напрямок руху $k = 1$. На всіх наступних кроках $k_n = k_{n-1} + 1$, тобто збільшується з кожним наступним кроком. Результатом виконання алгоритму буде вузол, значення якого відповідає значенню, для якого необхідно будо побудувати представлення, та який є останнім елементом зв'язного списку, що представляє протокол пошуку числа.

Для виконання арифметичних операцій базовими поняттям є верхній та нижній зрізи сюрреального числа, та найбільш раннє розділяюче сюрреальне число.

Завдяки представленню протоколу пошуку як зв’язного списку, кожне сюрреальне число знає про всі числа, які були отримані раніше за нього. Таким чином отримання зрізів зводиться до циклічної оброки всіх вузлів передуючих кінцевому вузлу, та визначення, чи відповідає це число умові для конкретного зрізу (менше ніж кінцеве, для нижнього, та більше ніж кінцеве, для верхнього). Для спрощення цього алгоритму, при побудові протоколу пошуку, в кожен вузол заноситься інформація про те, як він відноситься до числа, представлення якого необхідно побудувати (більший, менший, або дорівнює, для кінцевого вузла).

Розділяюче число, взагалі кажучи, визначається як число, що розділяє дві множини сюрреальнгих чисел, але очевидно, що найбільш строга умова для розділяючого числа бути більшим найбільшого числа з лівої множини, та меншим найменшого в правій множині. Таким чином, побудова розділяючого сюрреального числа виконується за алгоритмом, що аналогічний до алгоритму побудови сюрреального представлення дійсного числа, проте умовоювиходу з алгоритму є не досягнення заданого значення, а відповідність умові відносно двох значень. Таким чином результатом алгоритму буде найбільш раннє розділяюче число, що і необхідно для подальшого виконання арифметичних операцій.

Першою операцією, яку необхідно реалізувати для можливості подальшої реалізації наступних операцій є операція додавання сюрреальних чисел. Визначення цієї операції є рекурсивним (як і визначення самого сюрреального числа), та спирається на те, що операція додавання була визначена для більш ранніх чисел, ніж розглядаємі доданки. Таким чином, основою алгоритму є доведений факт, що додавання сюрреального числа {∅|∅} (Z в квазідвійковому представленні) до будь якого сюрреального числа, дорівнює цьому числу. В іншому випадку виконується повний алгоритм додавання двох сюрреальних чисел. Основою цього алгоритму є додавання сюрреального числа, одного доданку, до всіх сюрреальних чисел, що формують один із зрізів іншого доданку. Таким чином отримуються ліва та права множини форми сюрреального числа, Для яких в подальшому можна знайти

розділяючий елемент, який і буде результатом операції додавання. Проте, такий алгоритм має досить високу обчислювальну складність, через те, що при додаванні двох сюрреальних чисел, через рекурсивний характер опреації, багато раз повторюються додавання однакових доданків. Зі збільшенням значення, або точності доданків, і відповідно збільшенням кількості елементів їх зрізів, швидкість виконання операції додавання знижується на декілька порядків. Для вирішення цієї проблеми реалізований алгоритм операції додавання використовує кеш проміжних значень. Кеш побудовано на основі хеш-таблиці, ключом в якій є дійсне значення, а об'єктом є протокол пошуку, що відповідає цьому дійсному значенню. При виконанні будь якого додавання, алгоритм спочатку перевіряє, чи був знайдений протокол пошуку числа, для суми значень отриманих доданків, якщо таке значення знаходиться в кеші, то відповідний протокол пошуку просто повертається з кешу, і безпосередній алгоритм додавання не виконується. Така оптимізація дозволила значно збільшити швидкість виконання операції додавання двох сюрреальних чисел, при цьому не втративши коректність виконання операції.

Алгоритм операції множення реалізується аналогічним чином, з алгоритмом операції додавання. Цей алгоритм так само рекурсивний (як і операція множення), та використовує доведений факт, що множення сюрреального числа {∅|∅} (Z в квазідвійковому представленні) на будь яке сюрреальне число дорівнює сюрреальному числу {∅|∅}. Але алгоритм операції множення двох сюрреальних чисел має одну ключову відмінність від алгоритму операції додавання. Якщо в операції додавання кожна з підмножин лівої та правої множини форми сюрреального числа формувалася виконанням пеіних дій для всіх елементів одного із зрізів певного доданку, так іншим доданком, то в опеарції множення відповідні підмножини формуються з результатів певних дій не для множини, а для декартового добутку двох множин, а точніше певних зрізів обох співмножників, та безпосередньо співмножників. Таким чином обчислювальна складність алгоритму операції множення значно більша ніж обчислювальна складність операції додавання.

Важливим для розуміння обчислювальної складності цього алгоритму є і те, що в ньому на кожній ітерації виконується по декілька операцій додавання. Як і в алгоритмі операції додавання, алгоритм операції множення виконується значно довше при збільшені значення, або точності співмножників, через багаторазове виконання однакових операцій множення. Проте, використана в алгоритмі операції додавання оптимізація з використанням кешу проміжних значень не є можливою в алгоритмі операції множення, через певні особливості її математичного визначення та особливості реалізуючого її алгоритму.

Найскладнішою арифметичною операцією над сюрреальними числами є ділення двох сюрреальних чисел, яке зводиться до множення на число оберненен до дільника, і таким чином до знаходження оберненого числа взагалі. Формула знаходження оберненого числа до заданого сюрреального числа, є сама по собі досить складною, та базується на попередньо визначених операціях додавання та множення. Алгоритм, що реалізує операцію знаходження оберненого сюрреального числа, виявився настільки ресурсоємним та повільним, що не можна вважати його коректним, та таким, що виконує поставлену задачу. В більшості випадків виконання цього алгоритму призводить до переповнення стеку виконавчого потоку, через дуже велику глибину рекурсії, багато в чому обумовлену виконанням на кожній операції і без цього ресурсозатратної операції множення сюрреальних чисел. Можливі оптимізації цього алгоритму потребують окремого дослідження, та, можливо, використання іншого, відмінного від квазідвійкового, представлення сюрреального числа, або використання специфічних, проте ще не досліджених, властивостей квазідвійкового представлення сюрреального числа, що дозволить при комп’ютерній обробці сюрреальних даних користуватися спрощеними схемами виконання операцій, які були наведені вище, та побудувати спрощену схему безпосередньо операції знаходження числа оберненого до заданого сюрреального числа. В такому випадку можливо буде відмовитися від використання повного протоколу пошуку при виконанні операції, та звести їх до алгоритмів заснованих на двійкових

операціях над даними. Тоді проблема з глибиною рекурсії, та з багатократним повторенням однакових операцій буде вирішена на рівні визначення певної операції над сюрреальними числами.

2.3 Порівняльний аналіз реалізованого програмного засобу та програм аналогів

Здійснення порівняльного аналізу отриманого програмного засобу виявилося досить складним, через те, що існує лише декілька програмних засобів реалізуючих арифметичні дії над сюрреальними числами, та всі ці засоби представляють з себе бібліотеки для використання у спеціалізованих середовищах математичного моделювання. З опису цих бібліотек можна зрозуміти лише те, що для представлення сюрреальних чисел, та виконання арифметичних операцій над ними використовується визначення сюрреального числа за Конвеєм, на відміну від розробленого програмного засобу, що використовує квазідвійкове представлення сюрреальних чисел. З існуючих програмних продуктів виконання арифметичних операцій над сюрреальними числами, лише в одному заявляється часткова можливість знаходження оберненого сюрреального числа, проте ця можливість досягається за рахунок вбудованих можливостей середовища виконання.

2.4 Інструкція роботи користувача

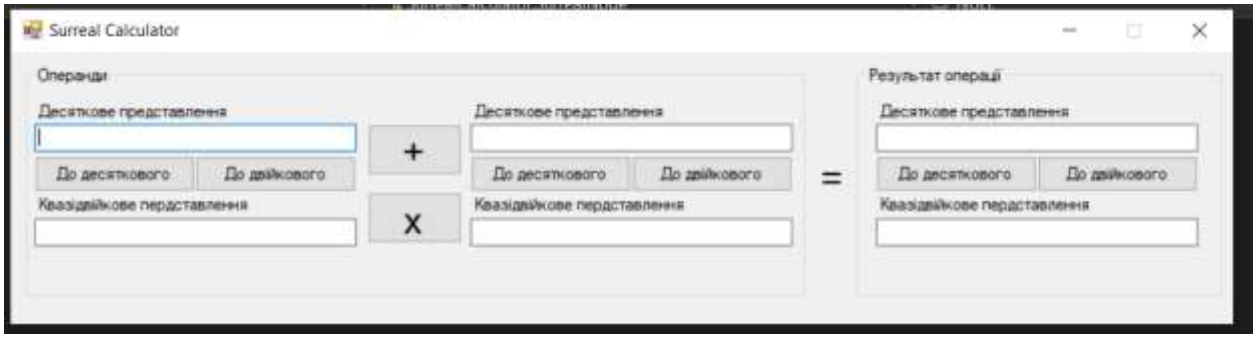

Рис. 2.4.1 Користувацький інтерфейс програмного засобу

На рисунку 2.4.1 представлено зовнішній вигляд користувацького інтерфейсу розробленого програмного засобу. Для виконання операції множення чи додавання необхідно ввести в відповідні поля в області «Операнди» квазідвійкове представлення сюрреальних чисел, над якими буде виконуватися операція. Можливо

також ввести дійсне число, та натиснувши кнопку «До двійкового» отримати квазідвійкове представлення сюрреального числа, що відповідає введеному дійсному числу. Після вводу операндів необхідно натиснути кнопку «+» для виконання додавання, чи «х» для виконання множення. Після цього в області «Результат» буде відображений результат виконаної операції. При необхідності дізнатися дійсне число, що відповідає сюрреальному числу, квазідвійкове представлення якого введене в одне з полів «Квазідвійкове представлення», необхідно натиснути кнопку «До десяткового», результат буде виведений в відповідному полі «Десяткове представлення».

При необхідності, основні алгоритми та структури даних, що являють собою ядро розробленого програмного засобу, і безпосередньо здійснюють всі перетворення та обчислення, можуть бути винесені в окрему бібліотеку, та повторно використовуватися у інших проектах, що засновані на платформі .NET Core. Завдяки використанню середовища виконання, яке надається платформою .NET Core, ядро програмного засобу є кросплатформеним, та може використовуватися незалежно від апаратного забезпечення та операційної системи. Також можливо використовувати розроблене ядро в додатках не заснованих на .NET Core, проте лише в рамках операційної системи Windows, за допомогою технології COM.

2.5 Тестування роботи програмного модуля

Мета: перевірка програмного засобу на відповідність вимог та оцінювання рівня якості програмного продукту.

Рівень плану: Тест План (*Test Plan*).

Склад документа: опис тестувальних робіт, об'єкта тестування, стратегії тестування, розкладу, критеріїв початку і закінчення тестування.

План проекту: тестування програмного засобу як повноцінного продукту.

Опис тестованого продукту: представляє програму для роботи з сюрреальними числами. Програму написано мовою C# в середовищі Microsoft Visual Studio.

Відповідний стандарт: *IEEE 829-1998 Format*.

Тестові завдання. Тестування якості програмних модулів продукту та всього програмного продукту.

Повинно бути проведено тестування таких частин:

1. Програмний засіб.

2. Коректність введення інформації.

3. Коректність виконання операцій.

До аспектів, що перевіряються, з точки зору користувачів, основні функції програмного засобу повинні тестуватися на:

1. Функціональні можливості (правильність, здатність до взаємодії).

2. Надійність (стабільність, стійкість до помилки).

3. Практичність (зрозумілість, простота використання).

4. Ефективність (характер зміни в часі, характер зміни ресурсів).

5. Супроводжуваність (аналізованість).

6. Мобільність (адаптованість, простота впровадження).

Нетестовані аспекти. Модулі, які рідко перевіряються користувачами:

1. Узгодженість.

2. Захищеність.

3. Відновлюваність.

4. Стійкість.

5. Взаємозамінність.

Підхід до тестування. Рівень тестування: системний, з точки зору кінцевого користувача.

Спеціальні засоби тестування: відсутні, тестування буде проводитися вручну.

Метрики: в рамках даного плану передбачається створити комплект тестів, повний щодо метрики за тестовими випадками (*Test Cases*) (відповідно до міжнародного стандарту *ISO 14598*).

Особливі вимоги до тестування: відсутні, тестування проводиться в звичайному режимі.

Сегмент компонентів: певний сегмент компонентів повинен бути протестований разом.

Обмеження для тестування: істотним обмеженням є проведення тестування вручну.

Рекомендована методика тестування: ручна, тому що зменшує матеріальні та програмні витрати на проведення тестування.

Критерії успішності та припинення тестування. Критерії успішності тестування:

1. система передається в експлуатацію, коли розроблений повний комплект тестів і всі розроблені тести виконуються без помилок;

2. вдале тестування на виправлену помилку при повторній передачі на тестування.

Критерії припинення тестування:

1. після виконання тестового сценарію.
2. після завершення кінцевого терміну перевірки (дедлайну).
3. програма має серйозні недоліки, що подальше тестування просто не має жодного сенсу.
4. основні баги знайдені, шукати далі економічно не вигідно.

Тест кейси зображені на таблицях 2.1, 2.2, 2.3.

Таблиця 2.1 – Тест кейс для вибору файлів

Дія	**Очікуваний результат**	**Фактичний результат**
Передумова		
Відкрити програмний засіб	Програмний засіб відкритий та доступний для використання	Пройдено
Кроки тесту		
Ввести інформацію	Інформацію успішно введено	Пройдено
Постумова		
Завершити роботу програмного засобу	Робота програмного засобу завершена	Пройдено

Таблиця 2.2 – Тест кейс для основних функцій модуля

Дія	**Очікуваний результат**	**Фактичний результат**
Передумова		
Відкрити програмний засіб	Програмний засіб відкритий та доступний для використання	Пройдено
Кроки тесту		
Виконати операцію додавання	Операцію виконано	Пройдено
Виконати операцію добутку	Операцію виконано	Пройдено

Постумова		
Завершити роботу програмного засобу	Робота програмного засобу завершена	Пройдено

Висновком виконання тестового випадку (табл. 2.1) є те, що програмний засіб може коректно приймати інформацію для подальшого виконання функцій.

Результатом виконання наступного тестового випадку (табл. 2.2) є те, що основні функції програмного засобу виконуються коректно, а саме – виконання кодування та декодування.

3. Охорона праці

Безпека виконання людьми своїх трудових обов’язків виходить на перший план, особливо з розвитком науково-технічного прогресу та новітніх технологій. Наука про безпеку праці і життєдіяльності людини була створена і розвивається саме у зв'язку з цим.

Безпека життєдіяльності (БЖД) - це комплекс заходів, спрямованих на забезпечення безпеки людини в середовищі проживання, збереження його здоров'я, розробку методів і засобів захисту шляхом зниження впливу шкідливих і небезпечних факторів до допустимих значень, вироблення заходів по обмеженню збитку в ліквідації наслідків надзвичайних ситуацій мирного і воєнного часу .

Мета і зміст БЖД:

- виявлення і вивчення чинників навколишнього середовища, що негативно впливають на здоров'я людини;
- ослаблення дії цих чинників до безпечних меж або виключення їх якщо це можливо;
- ліквідація наслідків катастроф і стихійних лих.

Коло практичних задач БЖД перш за все обумовлений вибором принципів захисту, розробкою і раціональним використанням засобів захисту людини і природного середовища від впливу техногенних джерел і стихійних явищ, а також коштів, що забезпечують комфортний стан середовища життєдіяльності.

Охорона здоров'я трудящих, забезпечення безпеки умов праці, ліквідація професійних захворювань і виробничого травматизму складає одну з головних турбот людського суспільства. Звертається увага на необхідність широкого застосування прогресивних форм наукової організації праці, зведення до мінімуму ручної, малокваліфікованої праці, створення обстановки, що виключає професійні захворювання і виробничий травматизм.

На робочому місці повинні бути передбачені заходи захисту від можливого впливу небезпечних і шкідливих факторів виробництва. Рівні цих чинників не повинні перевищувати граничних значень, обумовлених правовими, технічними та санітарно-технічними нормами. Ці нормативні документи зобов'язують до створення на робочому місці умов праці, при яких вплив небезпечних і шкідливих чинників на працюючих або усунуто зовсім, або знаходиться в допустимих межах.

Даний розділ дипломного проекту присвячений розгляду наступних питань:

- визначення оптимальних умов праці програміста;
- вимоги до виробничих приміщень;
- розрахунок освітленості та рівня шуму.

3.1 Характеристика умов праці програміста

Науково-технічний прогрес вніс серйозні зміни в умови виробничої діяльності робітників розумової праці. Їх праця стала більш інтенсивним, напруженим, які вимагають значних витрат розумової, емоційної і фізичної енергії. Це зажадало комплексного рішення проблем ергономіки, гігієни і організації праці, регламентації режимів праці та відпочинку.

В даний час комп'ютерна техніка широко застосовується у всіх областях діяльності людини. При роботі з комп'ютером людина піддається дії ряду небезпечних і шкідливих виробничих факторів: електромагнітних полів (діапазон радіочастот: ВЧ, УВЧ і СВЧ), інфрачервоного і іонізуючого випромінювань, шуму і вібрації, статичної електрики і ін.

Робота з комп'ютером характеризується значною розумовою напругою і нервово-емоційним навантаженням операторів, високою напруженістю зорової роботи і достатньо великим навантаженням на м'язи рук при роботі з клавіатурою ЕОМ. Велике значення має раціональна конструкція і розташування елементів

робочого місця, що важливо для підтримки оптимальної робочої пози людини-оператора.

У процесі роботи з комп'ютером необхідно дотримувати правильний режим праці та відпочинку. В іншому випадку у персоналу наголошуються значна напруга зорового апарату з появою скарг на незадоволеність роботою, головні болі, дратівливість, порушення сну, втому і хворобливі відчуття в очах, в поясниці, в області шиї і руках.

3.2 Вимоги до виробничих приміщень

3.2.1 Забарвлення і коефіцієнти віддзеркалення

Забарвлення приміщень і меблів повинні сприяти створенню сприятливих умов для зорового сприйняття, гарного настрою.

Джерела світла, такі як світильники і вікна, які дають віддзеркалення від поверхні екрану, значно погіршують точність знаків і тягнуть за собою перешкоди фізіологічного характеру, які можуть виразитися в значній напрузі, особливо при тривалій роботі. Віддзеркалення, включаючи віддзеркалення від вторинних джерел світла, повинне бути зведено до мінімуму. Для захисту від надмірної яскравості вікон можуть бути застосовані штори і екрани .

Залежно від орієнтації вікон рекомендується наступна фарбування стін і підлоги:

- вікна орієнтовані на південь: - стіни зеленувато-блакитного або світло-блакитного кольору; підлога - зелений;
- вікна орієнтовані на північ: - стіни світло-оранжевого або оранжево-жовтого кольору; підлога - червонувато-оранжевий;
- вікна орієнтовані на схід: - стіни жовто-зеленого кольору;

- підлога зелена або червонувато-оранжевий;
- вікна орієнтовані на захід: - стіни жовто-зеленого або голубувато-зеленого кольору; підлога зелена або червонувато-оранжевий.

У приміщеннях, де знаходиться комп'ютер, необхідно забезпечити наступні величини коефіцієнта віддзеркалення: для стелі: 60 ... 70%, для стін: 40 ... 50%, для підлоги: близько 30%. Для інших поверхонь і робочих меблів: 30 ... 40%.

3.2.2 Освітлення

Правильно спроектоване і виконане виробниче освітлення покращує умови зорової роботи, знижує стомлюваність, сприяє підвищенню продуктивності праці, благотворно впливає на виробниче середовище, надаючи позитивну психологічну дію на працюючого, підвищує безпеку праці і знижує травматизм.

Недостатність освітлення приводить до напруги зору, ослабляє увагу, приводить до настання передчасної стомленості. Надмірно яскраве освітлення викликає засліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Всі ці причини можуть призвести до нещасного випадку або профзахворювань, тому такий важливий правильний розрахунок освітленості.

Існує три види освітлення - природне, штучне і поєднане (природне і штучне разом).

Природне освітлення - освітлення приміщень денним світлом, що потрапляє через світлові прорізи в зовнішніх огороджуючих конструкціях приміщення. Природне освітлення характеризується тим, що змінюється в широких межах залежно від часу дня, пори року, характеру області і ряду інших чинників.

Штучне освітлення застосовується при роботі в темний час доби і вдень, коли не вдається забезпечити нормовані значення коефіцієнта природного освітлення (похмура погода, короткий світловий день). Освітлення, при якому недостатнє за

нормами природне освітлення доповнюється штучним, називається змішаним освітленням.

Штучне освітлення підрозділяється на робоче, аварійне, евакуаційне, охоронне. Робоче освітлення, у свою чергу, може бути загальним або комбінованим. Загальне - освітлення, при якому світильники розміщуються у верхній зоні приміщення рівномірно, або, як розташоване устаткування. Комбіноване - освітлення, при якому до загального додається місцеве освітлення.

Згідно СНіП II-4-79 в приміщень обчислювальних центрів необхідно застосувати систему комбінованого освітлення.

При виконанні робіт категорії високої зорової точності (найменший розмір об'єкту розрізнення 0,3 ... 0,5 мм) величина коефіцієнта природного освітлення (КЕО) повинна бути не нижче 1,5%, а при зоровій роботі середньої точності (найменший розмір об'єкту розрізнення 0,5 ... 1,0 мм) КЕО повинен бути не нижче 1,0%. В якості джерел штучного освітлення звичайно використовуються люмінесцентні лампи типа ЛБ, або ДРЛ, які попарно об'єднуються в світильники, які повинні розташовуватися рівномірно над робочими поверхнями .

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300лк, а комбінована - 750лк; аналогічні вимоги при виконанні робіт середньої точності - 200 і 300лк відповідно.

Крім того все поле зору повинне бути освітлено достатньо рівномірно - ця основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими, оскільки яскраве світло в районі периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності.

3.2.3 Параметри мікроклімату

Параметри мікроклімату можуть мінятися в широких межах, у той час як необхідною умовою життєдіяльності людини є підтримка постійності температури тіла завдяки терморегуляції, тобто здатності організму регулювати віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату - створення оптимальних умов для теплообміну тіла людини з навколишнім середовищем.

Обчислювальна техніка є джерелом істотних тепловиділень, що може привести до підвищення температури і зниження відносної вологості в приміщенні. У приміщеннях, де встановлені комп'ютери, повинні дотримуватися певні параметри мікроклімату. У санітарних нормах СН-245-71 встановлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від пори року, характеру трудового процесу і характеру виробничого приміщення (див. табл. 7.1) .

Об'єм приміщень, в яких розміщені працівники обчислювальних центрів, не повинен бути меншим 19,5 м 3 / людини з урахуванням максимального числа одночасно працюючих в зміну. Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери, приведені в табл. 3.2.3.1.

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря в приміщенні	22 ... 24 ° С
	Відносна вологість	40 ... 60%
	Швидкість руху повітря	до 0,1 м / с
Теплий	Температура повітря в приміщенні	23 ... 25 ° С
	Відносна вологість	40 ... 60%
	Швидкість руху повітря	0,1 ... 0,2 м / с

Таблиця 3.2.3.1 Параметри мікроклімату для приміщень, де встановлені комп'ютери

Характеристика приміщення	Об'ємна витрата подається в приміщення свіжого повітря, м 3 / на одну людину в годину
Об'єм до 20м 3 на особу	Не менше 30
20 ... 40м 3 на особу	Не менше 20
Більш 40м 3 на особу	Природна вентиляція

Таблиця 3.2.3.2 Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери

Для забезпечення комфортних умов використовуються як організаційні методи (раціональна організація проведення робіт залежно від пори року і доби, чергування праці і відпочинку), так і технічні засоби (вентиляція, кондиціювання повітря, опалювальна система).

3.2.4 Шум і вібрація

Шум погіршує умови праці надаючи шкідливу дію на організм людини. Працюючі в умовах тривалої шумової дії випробовують дратівливість, головні болі, запаморочення, зниження пам'яті, підвищену стомлюваність, зниження апетиту, біль у вухах і т.д. Такі порушення в роботі ряду органів і систем організму людини можуть викликати негативні зміни в емоційному стані людини аж до стресових. Під впливом шуму знижується концентрація уваги, порушуються фізіологічні функції, з'являється втома у зв'язку з підвищеними енергетичними витратами і нервово-психічним напруженням, погіршується мовна комутація. Все це знижує працездатність людини і її продуктивність, якість і безпеку праці. Тривала дія інтенсивного шуму [вище 80 дБ (А)] на слух людини приводить до його часткової або повної втрати.

У таблиці 3.2.4.1 вказані граничні рівні звуку залежно від категорії тяжкості і напруженості праці, що є безпечними відносно збереження здоров'я і працездатності.

Категорія напруженості праці	Категорія важкості праці			
	I. Легка	II. Середня	III. Важка	IV. Дуже важка
I. Мало напружений	80	80	75	75
II. Помірно напружений	70	70	65	65
III. Напружений	60	60	-	-
IV. Дуже напружений	50	50	-	-

Таблиця 3.2.4.1 Граничні рівні звуку, дБ, на робочих місцях

Рівень шуму на робочому місці математиків-програмістів і операторів відеоматеріалів не повинен перевищувати 50дБА, а в залах обробки інформації на обчислювальних машинах - 65дБА. Для зниження рівня шуму стіни і стеля приміщень, де встановлені комп'ютери, можуть бути облицьовані звукопоглинальними матеріалами. Рівень вібрації в приміщеннях обчислювальних центрів може бути понижений шляхом встановлення устаткування на спеціальні віброізолятори.

3.2.5 Електромагнітне і іонізуюче випромінювання

Більшість вчених вважають, що як короткочасне, так і тривалий вплив усіх видів випромінювання від екрану монітора не небезпечно для здоров'я персоналу, що обслуговує комп'ютери. Проте вичерпних даних щодо небезпеки дії випромінювання від моніторів на працюючих з комп'ютерами не існує і дослідження в цьому напрямі продовжуються.

Допустимі значення параметрів неіонізуючих електромагнітних випромінювань від монітора комп'ютера представлені в табл. 7.4.

Максимальний рівень рентгенівського випромінювання на робочому місці оператора комп'ютера звичайно не перевищує 10мкбер / ч, а інтенсивність ультрафіолетового і інфрачервоного випромінювань від екрану монітора лежить в межах 10 ... 100МВт / м 2.

Найменування параметра	Допустимі значення
Напруженість електричної складової електромагнітного поля на відстані 50см від поверхні відеомонітора	10В / м
Напруженість магнітної складової електромагнітного поля на відстані 50см від поверхні відеомонітора	0,3 А / м
Напруженість електростатичного поля не повинна перевищувати: • для дорослих користувачів; • для дітей дошкільних установ і що вчаться; • середніх спеціальних і вищих навчальних закладів.	 20кВ / м 15кВ / м

Таблиця 3.2.5.1 Допустимі значення параметрів неіонізуючих електромагнітних випромінювань (відповідно до СанПіН 2.2.2.542-96)

Для зниження дії цих видів випромінювання рекомендується застосовувати монітори із зниженим рівнем випромінювання (MPR-II, TCO-92, TCO-99), встановлювати захисні екрани, а також дотримуватися регламентовані режими праці та відпочинку.

3.2.6 Ергономічні вимоги до робочого місця

Проектування робочих місць, забезпечених відеотерміналами, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення устаткування, що до складу робочого місця і достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документа, клавіатури і т.д.), характеристики робочого крісла, вимоги до поверхні робочого столу, регульованість елементів робочого місця.

Головними елементами робочого місця програміста є стіл і крісло. Основним робочим положенням є положення сидячи.

Робоча поза сидячи викликає мінімальне стомлення програміста. Раціональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору.

Моторне поле - простір робочого місця, в якому можуть здійснюватися рухові дії людини.

Максимальна зона досяжності рук - це частина моторного поля робочого місця, обмеженого дугами, описуваними максимально витягнутими руками при русі їх у плечовому суглобі.

Оптимальна зона - частина моторного поля робочого місця, обмеженого дугами, описуваними передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем.

3.2.7 Розрахунок природного освітлення

Основною задачею світлотехнічних розрахунків при природному освітленні – визначення площі світлових прорізів.

Розрахунок природного освітлення проводиться у два етапи – проектний і перевірочний.

На першому етапі здійснюється попередній розрахунок площі світлових прорізів:

– $К_з$ – коефіцієнт запасу, залежить від концентрації пилу у приміщенні, періодичності їх очищення.

– $К_{буд}$ – коефіцієнт, що враховує затінення вікон протилежними будинками.

На другому етапі розрахунку при обраних світлових прорізах визначають дійсне значення КПО у різних точках приміщення з використанням графічного методу по СНіП II-4-79.

$$100\frac{S_0}{S_п} = \frac{e_{min} \times \eta_в \times К_з \times К_{буд}}{\tau_в \times r_1}, \qquad (3.4)$$

де $\mathbf{S_0}$ – площа світлових прорізів вікон при бічному освітленні;

$\mathbf{S}_п$ – площа підлоги приміщення;

$\boldsymbol{\eta}_в$ – світлова характеристика вікон, залежить від конфігурації і розмірів вікон і приміщень;

$К_з$ – коефіцієнт запасу (30-50%) ($К_з$ = 1, 3-1,5);

$К_{буд}$ – коефіцієнт, що враховує затінення вікон ворогуючими будинками;

$\boldsymbol{\tau}_в$ – загальний коефіцієнт світлопропускання, що враховує коефіцієнт світлопропускання скла і втрати світла в несучих конструкціях, в сонцезахисних пристроях, в захисній сітці, яка встановлюється над ліхтарями:

$$\tau_в = \tau_1 \times \tau_2 \times \tau_3 \times \tau_4 \times \tau_5, \qquad (3.5)$$

де $\boldsymbol{\tau_1}$ – світлопропускаючий матеріал;

$\boldsymbol{\tau_2}$ – конструкції палітурок;

$\boldsymbol{\tau_3}$ – забруднення вікон;

$\boldsymbol{\tau_4}$ – несучі конструкції;

$\boldsymbol{\tau_5}$ – наявність сонцезахисних пристроїв.

Для вікон будівель, не обладнаних сонцезахисними пристроями, τ_o = 0,5.

$\mathbf{r_1}$ – коефіцієнт, що враховує підвищення КПО при бічному освітленні завдяки світлу, відбитому від поверхонь приміщення і підстилаючого шару (земля, трава та ін.), прилеглого до будівлі.

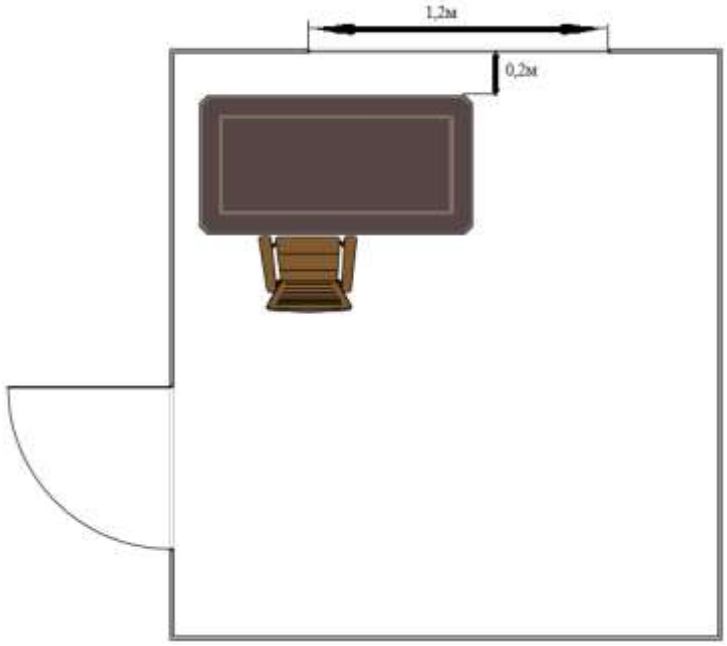

Рис. 3.2.7.1 План приміщення для розрахунку природного освітлення

3.2.8 Параметри мікроклімату

Мікроклімат виробничих приміщень – це комплекс фізичних факторів, що впливають на теплообмін людини і визначають самопочуття, працездатність, здоров'я і продуктивність праці. Підтримка мікроклімату робочого місця в межах гігієнічних норм - найважливіше завдання охорони праці.

Параметри клімату можуть змінюватися в широких межах, в той час як необхідною умовою життєдіяльності людини є підтримка постійної температури тіла завдяки терморегуляції, так це здатності організму регулювати віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату – створення оптимальних умов для теплообміну тіла людини з навколишнім середовищем. Обчислювальна техніка є джерелом суттєвих тепловиділень, що може призвести до підвищення температури і зниженню відносної вологості в приміщенні. В приміщення, де встановлені комп'ютери, повинні дотримуватися певні параметри мікроклімату. В санітарних нормах СН-245-71 встановлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від пори року, характеру трудового процесу і характеру виробничого приміщення.

Об'єм приміщень, в яких розміщені робітники обчислювальних центрів, не повинні бути менше 19,5 $м^3$/людину з урахуванням максимального числа одночасно працюючих в зміну. Норми подачі свіжого повітря в приміщеннях, де розташовані комп'ютери, приведені в Табл. 3.3.

Характеристика приміщення	Об'ємна витрата подаваного в приміщенні свіжого повітря, $м^3$/на одну людину в годину
*Об'єм до 20*$м^3$ *на людину*	Не менше 30
*20-40*$м^3$ *на людину*	Не менше 20
*Більше 40*$м^3$ *на людину*	Природна вентиляція

Табл. 3.2.8.1 Норми подання свіжого повітря в приміщення, де розташовані комп'ютери

Для забезпечення комфортних умов використовуються як організаційні методи (раціональна організація проведення робіт в залежності від пори року та доби,

чергування праці та відпочинку), так і технічні засоби (вентиляція, кондиціонування повітря, опалювальна система).

3.3 Шум і вібрація

3.3.1 Шум

Шумом прийнято вважати звуки, які негативно впливають на організм людини і заважають його роботі та відпочинку. Тому шум часто називають несприятливим звуком. Зазвичай шум створюється під час хаотичного чергування звуків різної частоти та інтенсивності. Звук, як фізичне явище, є коливальним рухом, що поширюється хвилеподібно у пружному середовищі (газоподібному, рідинному чи твердому). Звук, а значить і шум, характеризується:

- швидкістю звуку, м/с;
- частотою, Гц;
- звуковим тиском, Па;
- інтенсивністю, Вт/м2.

Рис. 3.3.1 Класифікація заходів та засобів віброзахисту

Рівень шуму на робочому місці програмістів не повинен перевищувати 50дБА, а в залах обробки інформації на обчислювальних машинах – 65дБА. Для зниження рівня шуму стіни і стеля приміщення, де встановлені комп'ютери, можуть бути облицьовані звукопоглинаючими матеріалами. Рівень вібрації в приміщеннях обчислювальних центрів може бути знижений шляхом установки обладнання на спеціальні віброізолятори.

3.2.2 Розрахунок рівня шуму

Одним з несприятливих факторів виробничого середовища є високий рівень шуму, що створюється друкарськими пристроями, устаткуванням для кондиціонування повітря, вентиляторами систем охолодження в самих ЕОМ.

Для вирішення питань про необхідність і доцільність зниження шуму необхідно знати рівні шуму на робочому місці.

Рівень шуму, що виникає від декількох некогерентних джерел, що працюють одночасно, підраховується на підставі принципу енергетичного підсумовування випромінювань окремих джерел:

$$L_{\Sigma} = 10\lg \sum_{i=1}^{i=n} 10^{0,1L_i}, \quad (3.3)$$

де $\mathbf{L_i}$ – рівень звукового тиску і-го джерела шуму;

n – кількість джерел шуму.

Отримані результати розрахунку порівнюються з допустимим значенням рівня шуму для даного робочого місця. Якщо результати розрахунку вище допустимого значення рівня шуму, то необхідні спеціальні заходи щодо зниження шуму. До них відносяться: облицювання стін і стелі залу звукопоглинальними матеріалами, зниження шуму в джерелі, правильне планування устаткування і раціональна організація робочого місця програміста.

Рівні звукового тиску джерел шуму, що діють на програміста на його робочому місці представлені в Табл. 3.6.

Джерело шуму	Рівень шуму, дБ
Жорсткий диск	40
Вентилятор	45
Монітор	17
Клавіатура	10
Принтер	45
Сканер	42

Табл. 3.2.2.1 Рівні звукового тиску різноманітних джерел

Зазвичай робоче місце програміста оснащене наступним устаткуванням: вінчестер в системному блоці, вентилятор(и) систем охолодження ПК, монітор, клавіатура, принтер і сканер.

Підставивши значення рівня звукового тиску для кожного виду обладнання в формулу, отримаємо:

$$L_{\sum} = 10\lg(10^{4} + 10^{4,5} + 10^{1,7} + 10^{1} + 10^{4,5} + 10^{4,2}) = 49{,}5 \text{ дБ}$$

Отримане значення не перевищує допустимий рівень шуму для робочого місця програміста, рівний 65 дБ (ГОСТ 12.1.003-83). І якщо врахувати, що навряд чи такі периферійні пристрої як сканер і принтер будуть використовуватися одночасно, то ця цифра буде ще нижчою. Крім того при роботі принтера безпосередня присутність програміста необов’язкова, так як принтер забезпечений механізмом автоподачі листів.

3.3 Вібрація

3.3.1 Причини вібрації та характеристика основних вібраційних параметрів

Вібрація – це коливальні процеси, що відбуваються у механічних системах. Найпростішою формою вібрації є гармонійні синусоїдні коливальні рухи.

Основні параметри синусоїдного коливання:

– частота в Герцах (ʃ = 1 кол/с, амплітуда зміщення – А м або см);

– коливальна швидкість – V (м/с);

– прискорення – W (м/с^2).

Для синусоїдних коливань швидкість і прискорення визначають за формулами:

$$V = 2П \times ʃ \times A, \quad (3.4)$$

$$W = (2П \times ʃ)^2 \times A, \quad (3.5)$$

За опорний нульовий рівень коливальної швидкості взято величину 5×10^{-8} м/с.

За нульовий рівень коливального прискорення взято величину 3×10^{-4} м/с^2.

В практиці віброакустичних вимірювань використовують не абсолютні значення параметрів, а відносні величини – віброшвидкості L_v і рівень віброприскорення L_w, які визначаються відносного опорного значення й вимірюються в децибелах (дБ).

Відносні рівні L_v і L_w визначаються за формулами:

$$L_v = 20\lg(\frac{V}{5} \times 10^{-8}), \quad (3.6)$$

$$L_w = (\frac{V}{5} \times 10^{-4}), \quad (3.7)$$

де **V** і **W** – коливальна швидкість і прискорення в точці вимірювання, м/с і м/с^2.

Порогове відчуття вібрації виникає тоді коли прискорення її дорівнює 1% від нормального прискорення сил земного тяжіння.

Хворобливе відчуття виникає, коли прискорення становить 5% від прискорення земного тяжіння, тобто при 0,5 м/с^2.

Коливальну швидкість 10^{-4} м/с людина сприймає як порогову, а при швидкості 1 м/с, виникають хворобливі відчуття.

Величина коливальної енергії поглинутої тілом людини (Q кгм), прямо пропорційна площі контакту, часу дії та інтенсивності подразника

$$Q = I \times S \times T, \qquad (3.8)$$

де **S** – площа контакту, м^2;

T – тривалість дії, с;

I – інтенсивність вібрації, кгм/м^2/с.

Інтенсивність вібрації, а відтак коливальна енергія прямо пропорційна квадрату коливальної швидкості:

$$l = V^2(\tfrac{z}{s}), \qquad (3.9)$$

де **V –** середньоквадратичне значення коливальної швидкості, м/с;

$\frac{\mathbf{z}}{\mathbf{s}}$ – модуль вхідного питомого механічного імпеденса в зоні контакту, кг/с.

Механічний імпеданс визначається як відношення коливальної сили до результуючої коливальної швидкості в точці прикладання цієї сили.

У виробничих умовах майже не зустрічаються прості гармонійні коливання там переважають аперіодичні, квазіперіодичні, імпульсивні або поштовхоподібні вібрації.

Залежно від джерела вібрації і характеру контакту з тілом людини вібрація умовно поділяється на місцеву (локальну), загальну та змішану.

3.4 Ергономічні вимоги до робочого місця

Проектування робочих місць, забезпечених відеотерміналами, відносяться до числа важливіших проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинно відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер праці. В тому числі, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення обладнання, що входить в склад робочого місця і достатній робочий простір, що дозволяє втілювати всі необхідні рухи та переміщення.

Ергономічними аспектами проектування відеотермінальних робочих місць, в тому числі, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різноманітного розміщення документів, відстань від очей користувача до екрану, документу, клавіатури і т.і.), характеристики робочого крісла, вимоги до поверхні робочого стола, урегульованість елементів робочого місця.

Головним елементом робочого місця програміста є стін та крісло. Основним робочим положенням є положення сидячи.

Моторне поле – простір робочого місця, в якому можуть втілювати моторні дії людини.

Оптимальне розміщення предметів праці і документації в зонах досяжності представлено на Рис. 3.2.

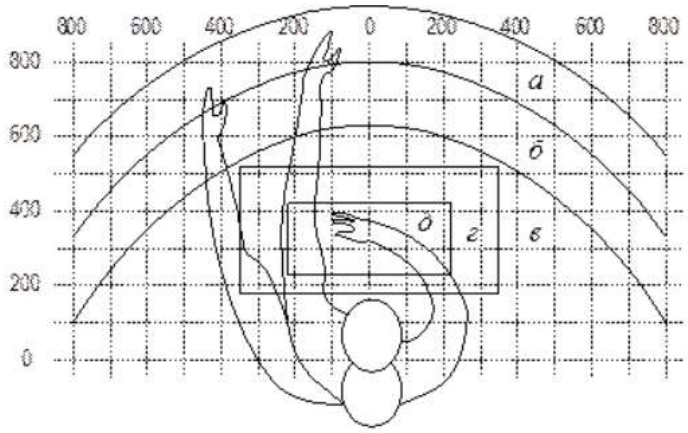

Рис. 3.4.1 – Зони досяжності рук у горизонтальній площині:

а – зона максимальної досяжності;

б – зона досяжності пальців при витягнутій руці;

в – зона легкої досяжності долоні;

г – оптимальний простір для грубої ручної роботи;

д – оптимальний простір для тонкої ручної роботи

Оптимальна зона – частина моторного поля робочого місця, обмеженого дугами, що описують передпліччя при русі в ліктьовому сухожиллі з опорою в точці ліктя і з відносно нерухомим плечем.

Максимальна зона досяжності рук – це частина моторного поля робочого місця, обмежено дугами, що описує максимально витягнутими руками під час руху їх в плечовому сухожиллі.

На Рис. 3.4.2 показано розміщення основних і периферійних складових ПК на робочому столі програміста.

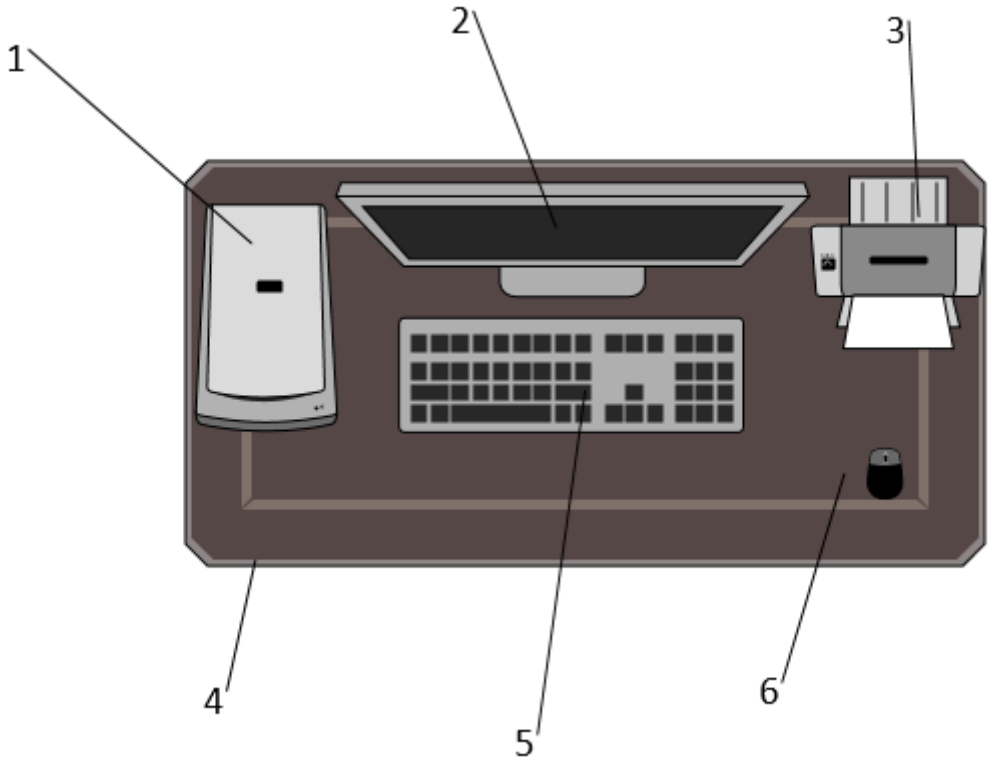

Рис. 3.4.2 Розміщення основних і периферійних складових ПК:

1 – сканер;

2 – монітор;

3 – принтер;

4 – поверхня робочого столу;

5 – клавіатура;

6 – маніпулятор типу «миша»

Для комфортної роботи слід повинен задовольняти наступні умови:

- висота столу повинна бути обрана с урахуванням можливості сидіти вільно, в зручній позі, при необхідності спираючись на підлокітники;
- нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, не був змушений підбирати під себе ноги;

- поверхня столу повинна мати властивості, що виключають появу відблисків в полі зору програміста;
- конструкція столу повинна передбачати наявність висувних ящиків (не менше 3 для зберігання документації, лістингів, канцелярського приладдя);
- висота робочої поверхні рекомендується в межах 680-760 мм. Висота поверхні, на яку встановлюється клавіатура, повинна бути близько 650 мм.

Велике значення надається характеристикам робочого крісла. Так, рекомендована висота сидіння на рівнем поля знаходиться в межах 420 – 550 мм. Необхідно, щоб робочий стілець вільно обертався відносно основи, регулювався по висоті і, крім того, допускав можливість змінювати кут нахилу спинки (добре, якщо і сидіння також), а також встановлювати потрібну відстань від спинки до переднього краю сидіння. Оббивка крісла повинна бути не тільки практичною, стійкою до довготривалих фізичних впливів, але і гігієнічною, так це виконаною з матеріалів, нешкідливих для здоров'я і, що забезпечують зручність і комфорт в роботі.

Ідеальна висота сидіння – коли ступні ніг повністю торкаються підлоги, а кут згину колін при цьому складає приблизно 90˚. Дуже важливо, щоб край сидіння мав м'яку округлену вниз форму. Це дозволяє уникнути натиску на кровоносні судини і не порушувати циркуляцію крові.

Необхідно також передбачати при проектуванні можливість різноманітного розміщення документів: збоку від монітору, між монітором і клавіатурою і т.п. Крім цього, у випадках, коли монітор має низьку якість зображення, наприклад помітні миготіння, відстань від очей до екрану роблять більше (близько 700 мм), ніж відстань від очей до документу (300-450 мм). Взагалі при високій якості зображення на моніторі відстань від очей користувача до екрану, документу і клавіатури може бути однаковим.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з’явитися біль в м’язах, суглобах і сухожиллях. Вимоги до робочої пози користувача монітора наступні:

- голова не повинна бути нахилена більш ніж на 20˚;
- плечі повинні бути розслаблені, лікті – під кутом 80-100˚;
- передпліччя і кисті рук – в горизонтальному положенні.

Причина неправильної пози користувача обумовлена наступними факторами: немає хорошої підставки для документів, клавіатура знаходиться занадто високо, а документи – низько, нікуди покласти руки і кисті, недостатньо простору для ніг.

З метою подолання вказаних недоліків даються загальні рекомендації:

- застосувати пересувну клавіатуру;
- повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури і екрану, а також підставка для рук.

Правильна постава під час роботи максимально розвантажує м’язи і дозволяє працювати довше, менше втомлюючись.

Але навіть абсолютно правильна постава не допоможе, якщо увесь день сидіти в одній позі. Нерухоме положення, навіть абсолютно правильне, призведе до м’язової втоми.

Правильна постава передбачає зміну пози приблизно два рази на годину.

Навість незначна зміна положення тіла кожні півгодини зміщують навантаження на інші м’язи, що дозволяє м’язам перепочити і запастися поживними речовинами.

Під час роботи на персональному комп’ютері дуже важливу роль відіграє дотримання правильного режиму праці та відпочинку. В іншому випадку у персоналу відзначаються напруження зорового апарату з появою скарг на незадоволеність роботою, головний біль, дратівливість, порушення сну, втомленість

і больові відчуття в очах, в попереку, в області шиї та руках. Також можливі нервово-психічні перевантаження:

- розумове перенавантаження;
- перенавантаження аналізаторів;
- монотонність праці;
- емоційні перевантаження.

Вплив цих факторів можна послабити правильним режимом праці та відпочинку.

В Табл. 3.5 представлені відомості про регламентовані перерви, які необхідно робити під час роботи за комп'ютером, в залежності від довготривалості робочої зміни, видів і категорій трудової діяльності з ВДТ (відеодисплейний термінал) і ПЕОМ (відповідно до СанПіН 2.2.2 542-96 «Гігієнічні вимоги до відеодисплейних терміналів, персональних електронно-обчислювальних машин і організації робіт»).

Категорія роботи з ВДТ чи ПЕОМ	Рівень навантаження за робочу зміну при видах роботи з ВДТ			Сумарний час регламентованих перерв, хв	
	група А, кількість знаків	*група Б*, кількість знаків	*група В*, кількість знаків	при 8-годинній зміні	при 12-годинній зміні
I	до 20000	до 15000	до 2,0	30	70
II	до 40000	до 30000	до 4,0	50	90
III	до 60000	до 40000	до 6,0	70	120

Табл. 3.4.1 Час регламентованих перерв при роботі з комп'ютером

Примітка. Час перерв дано при дотриманні зазначених Санітарних правил і норм. У разі невідповідності фактичних умов праці вимогам СанПіН час регламентованих перерв слід збільшити на 30%.

Відповідно до СанПіН 2.2.2 546-96 усі види трудової діяльності, зв'язані з використанням комп'ютера поділяються на три групи:

– *група А:* робота з зчитування інформації з екрану ВДТ або ПЕОМ з попереднім запитом;

– *група Б:* робота по вводу інформації;

– *група В:* творча робота в режимі діалогу з ЕОМ.

Ефективність перерв підвищується при поєднанні з виробничою гімнастикою або організації спеціального приміщення для відпочинку персоналу із зручними м’якими меблями, акваріумом, зеленою зоною і т.п.

3.5 Заходи та засоби протипожежного захисту

Можна виділити 2 типи причин пожеж: електричного і неелектричного характеру. До перших відносяться: іскріння в електричних апаратах, машинах, електростатичні розряди і удари блискавки, струми коротких замикань, погані контакти в місцях з’єднання проводів та інше.

В якості засобів пожежної сигналізації передбачається використовувати автоматичний комбінований сповіщувач, який реагує на виникнення диму, на підвищення температури.

Оскільки в приміщенні серверної знаходяться електроустановки під напругою, то в якості засобів гасіння пожежі можна використовувати тільки вуглекислотні вогнегасники – ОУ-2, ОУ-5, ОУ-8, де 2, 5, 8 – відповідно ємність балонів. На підприємстві повинні бути розроблені заходи, усувають причини пожеж та вибухів. Вони підрозділяються на технічні, експлуатаційні, організаційні та режимні.

До технічних заходів відносять: дотримання протипожежних норм при влаштування опалення, вентиляції, виборі та монтажу електрообладнання. Експлуатаційними заходами називають правильну експлуатацію виробничих машин і устаткування.

До організаційних належать навчання виробничого персоналу протипожежним правилам і видання необхідних інструкцій і плакатів.

Усі струмопровідні частини, розподільні пристрої, пускові апарати повинні монтуватися на негорючих підставах. Вимірювання опору ізоляції електромережі повинно проводитися в приміщенні один раз в три роки. На випадок виникнення пожежі повинні бути розроблені плани евакуації людей.

При виявленні пожежі або його ознак (дим, запах гару і ін.). А також за сигналами оповіщення про пожежу та при евакуації, працівник і службовець повинен:

- повідомити про це в пожежну охорону за телефоном «101» або «112», при цьому необхідно повідомити точну адресу об'єкта, місце виникнення пожежі або виявлення ознак пожежі, ймовірну можливість загрози людям, а також інші відомості, необхідні диспетчеру пожежної охорони;
- крім того, слід назвати себе і номер телефону, з якого робиться повідомлення про пожежу;
- повідомити керівнику об'єкта про пожежу;
- голосом сповістити про пожежу або його ознаках людей, які перебувають поблизу і вжити людей з будівлі;
- по можливості використовуючи первинні засоби пожежогасіння, загасити вогнище пожежі. До гасіння пожежі приступати тільки в тому випадку, якщо немає загрози для життя, і існує можливість покинути небезпечну зону якщо буде потреба.

Яке з перерахованих дій є першочерговим? Це повинна вирішити в кожному конкретному випадку сама людина, що виявила пожежу. Найбільш ефективним є одночасне виконання всіх трьох першочергових заходів. Це можливо, якщо про пожар сповіщено відразу декількох чоловік і вони, розподіливши між собою обов'язки, зможуть вжити всі необхідні екстрені заходи.

Під час отримання сигналу про евакуацію робочі і службовці повинні забезпечити безаварійну зупинку виробництва, швидко без паніки, у відповідності з Планом евакуації покинути приміщення і вийти в безпечне місце.

Висновки

В ході даної дипломної роботи, в першому розділі було розглянуто детальний опис задач та вимог, розв’язання яких планується здійснити у дипломному проекті, вимоги до структури, архітектури та технічних характеристик розроблюваного програмного засобу, додаткові вимоги щодо реалізації його складових частин.. Були розглянуті теоретичні відомості про розробку такого плану програмного продукту, різноманітні підходи та їх переваги та недоліки. Був проведений огляд відомих методів розв’язання поставлених задач, обґрунтування обраних методів розв’язання задач, організації зберігання, пошуку та обробки інформації, опис математичної моделі досліджуваної системи або процесу, розробку та опис алгоритмів. Також була сформулювана актуальність роботи та визначенні задачі. У другому розділі розглянуто:

- опис алгоритму створення програмного засобу;
- опис засобів реалізації;
- порівняльний аналіз реалізованого програмного засобу та аналогів;
- інструкція роботи користувача.

У третьому розділі були розглянуті такі питання:

- визначення оптимальних умов праці техніка-програміста;
- розрахунок освітленості;
- розрахунок рівня шуму;
- заходи та засоби протипожежного захисту.

Список використаних джерел

1. Вигерс К. Разработка требований к программному обеспечению: пер с англ. / Вигерс Карл, Битти Джой — М.: Русская Редакция, 2004. — 314 с.
2. Desktop Operating System Market Share Worldwide [Електронний ресурс]/ StatCounter — Режим доступу: http://gs.statcounter.com/os-market-share/desktop.
3. .NET Framework 4 [Електронний ресурс]: відомості та системні вимоги — Режим доступу: https://www.microsoft.com/ru-ru/download/details.aspx?id=17718.
4. Шилдт Г. Полный справочник по C#: пер. с англ. / Герберт Шилдт: — М. Вильямс, 2004. — 752 с.
5. Sells Ch. Програмування Windows Forms in C# / Sells Chris: — AddisonWesley Professional, 2003.
6. Conway J.H. On numbers and Games. London: Academic Press Inc, 1976
7. Кнут Д. Сюрреальные числа. М.: Бином. Лаборатория знаний, 2014. -112 с.
8. Кириллов А., Клумова И., Сосинский А. Сюрреальные числа // Квант: журнал. – М.: Наука, 1979. -№11. –с. 2-9.
9. Деорнуа П. Комбинаторная теория игр. –М.: МЦНМО, 2017. -40с.
10. Працьовитий М.В. Геометрія класичного двійкового зображення дійсних чисел – Київ: Вид-во НПУ імені М.П.Драгоманова, 2012. -68с.
11. Пономаренко О. В., Лещинський О. Л., Погульський А. М. ТЕОРЕТИЧНІ АСПЕКТИ КВАЗІДВІЙКОВОГО ПРЕДСТАВЛЕННЯ СЮРРЕАЛЬНИХ ЧИСЕЛ.// Scientific achievements of modern society. Abstracts of the 8th International scientific and practical conference. Cognum Publishing House. Liverpool, United Kingdom. 2020. Pp. 651-661. URL: http://sci-conf.com.ua.
12. Пономаренко О. В., Лещинський О. Л., Погульський А. М. ОПЕРАЦІЇ ДОДАВАННЯ І МНОЖЕННЯ ДВІЙКОВОРАЦІОНАЛЬНИХ СЮРРЕАЛЬНИХ ЧИСЕЛ, ЗАДАНИХ СВОЇМ КВАЗІДВІЙКОВИМ ПРЕДСТАВЛЕННЯМ.// Dynamics of the development of world science. Abstracts of the 8th International

scientific and practical conference. Perfect Publishing. Vancouver, Canada. 2020. Pp. 679-689. URL: http://sci-conf.com.ua.

13. Пономаренко О. В., Лещинський О. Л., Погульський А. М. МНОЖЕННЯ І ДІЛЕННЯ СЮРРЕАЛЬНИХ ЧИСЕЛ ЗДАНИХ СВОЇМ КВАЗІДВІЙКОВИМ ПРЕДСТАВЛЕННЯМ// Eurasian scientific congress. Abstracts of the 5th International scientific and practical conference. Barca Academy Publishing. Barcelona, Spain. 2020. Pp. 296-304. URL: http://sciconf.com.ua.
14. Ріхтер, Джеффрі CLR via C #. Програмування на платформі Microsoft .NET Framework 4.0 мовою C # / Джеффрі Ріхтер. - М .: Питер, 2013. - 928 с.

Додаток А – Текст програми

SurrealNumber.cs

```
using System;
using System.Collections.Generic;
using System.Security.Cryptography.X509Certificates;
using System.Text;

namespace SurrealCalculator
{
   public enum Relation
   {
      EQUAL,
      MORE,
      LESS
   }
   public enum MoveDirection
   {
      ZERO,
      RIGTH,
      LEFT
   }
   public class SurrealNode
   {
      private SurrealNode prev;
      public SurrealNode Prev
      {
```

```
    get
    {
        return prev;
    }
    set
    {
        prev = value;
        if (prev != null)
        {
            Flag = prev.Flag;
            Level = prev.Level + 1;
            Scale = prev.Scale;
        }
    }
}
public Relation Rel;
public MoveDirection Direction;
public string BinValue;
public float Value;
public float Scale;
public bool Flag;
public int Level;

public const string Z = "Z";
public const string ONE = "1";
public const string NULL = "0";
```

```
public SurrealNode BuildTreeFromBin(string binRep, int i)
{
   if (i >= binRep.Length)
      return this;
   string str = binRep[i].ToString();
   if (str == Z)
   {
      Prev = null;
      SetAsZERO();
      return BuildTreeFromBin(binRep, i + 1);
   }
   else
   {
      SurrealNode next = new SurrealNode();
      next.Prev = this;
      MoveDirection d;
      if (str == ONE)
      {
         d = MoveDirection.RIGTH;
         Rel = Relation.LESS;
      }
      else
      {
         d = MoveDirection.LEFT;
         Rel = Relation.MORE;
```

```
        }
        next.Move(d);
        return next.BuildTreeFromBin(binRep, i + 1);
    }
}

public void Invers()
{
    Value = Value * -1;
    if(Rel!=Relation.EQUAL)
    {
        Rel = Rel == Relation.MORE ? Relation.LESS : Relation.MORE;
    }
    if (Direction != MoveDirection.ZERO)
    {
        Direction = Direction == MoveDirection.RIGTH ? MoveDirection.LEFT : MoveDirection.RIGTH;
    }
    if (Prev != null)
    {
        Prev.Invers();
    }
}

public List<SurrealNode> GetCut(Relation relation)
{
    List<SurrealNode> result = new List<SurrealNode>();
```

```
    SurrealNode current = Prev;
    while (current != null)
    {
        if (current.Rel == relation)
        {
            result.Add(current);
        }
        current = current.Prev;
    }
    return result;
}

private void SetAsZERO()
{
    Direction = MoveDirection.ZERO;
    Value = 0;
    Scale = 1;
    Flag = false;
    Level = 0;
}

public SurrealNode BuildSeparator(SurrealNode a, SurrealNode b)
{
    if (Prev == null)
    {
        SetAsZERO();
```

```
    }
    SurrealNode result = new SurrealNode();
    result = result.BuildTree(0);
    if (b == null)
        result = FindFirstMore(a);
    else if (a == null)
        result = result.FindFirstLess(b);
    else
        result = result.FindFirst(a, b);
    return result;
}

private SurrealNode FindFirstMore(SurrealNode l)
{
    if (Value <= l.Value)
    {
        SurrealNode n = new SurrealNode();
        Rel = Relation.LESS;
        n.Prev = this;
        n.Move(MoveDirection.RIGTH);
        return n.FindFirstMore(l);
    }
    else
    {
        return this;
    }
```

```
}

private SurrealNode FindFirstLess(SurrealNode r)
{
   if (Value >= r.Value)
   {
      SurrealNode n = new SurrealNode();
      Rel = Relation.MORE;
      n.Prev = this;
      n.Move(MoveDirection.LEFT);
      return n.FindFirstLess(r);
   }
   else
   {
      return this;
   }
}

private SurrealNode FindFirst(SurrealNode l, SurrealNode r)
{

   if (Value > l.Value && Value < r.Value)
      return this;
   SurrealNode n = new SurrealNode();
   if (Value <= l.Value)
```

```
    {
        //Node n = new Node();
        Rel = Relation.LESS;
        n.Prev = this;
        n.Move(MoveDirection.RIGTH);
        n= n.FindFirst(l, r);
    }
    if (Value >= r.Value)
    {
        //Node n = new Node();
        Rel = Relation.MORE;
        n.Prev = this;
        n.Move(MoveDirection.LEFT);
        n= n.FindFirst(l, r);
    }
    return n;
}

public SurrealNode BuildTree(float val)
{
    if (Prev == null)
    {
        SetAsZERO();
    }
    if (val != Value)
    {
```

```
        SurrealNode next = new SurrealNode();
        next.Prev = this;
        MoveDirection d;
        if (val > Value)
        {
            d = MoveDirection.RIGTH;
            Rel = Relation.LESS;
        }
        else
        {
            d = MoveDirection.LEFT;
            Rel = Relation.MORE;
        }
        next.Move(d);
        return next.BuildTree(val);
    }
    else
    {
        Rel = Relation.EQUAL;
        return this;
    }
}

private void Move(MoveDirection direction)
{
    Direction = direction;
```

```
    if ((Prev.Direction != direction && Prev.Direction != MoveDirection.ZERO) && !Flag)
    {
        Flag = true;
    }
    if (Flag)
    {
        Scale = Scale / 2;
    }
    int sign = DirectionToSign(direction);
    Value = Prev.Value + (sign * (1 * Scale));
}

private int DirectionToSign(MoveDirection direction)
{
    if (direction == MoveDirection.LEFT)
    {
        return -1;
    }
    else
    {
        return 1;
    }
}

private static void SumWithSet(List<SurrealNode> set, SurrealNode node, List<SurrealNode> resultSet)
{
```

```
    foreach (SurrealNode n in set)
    {
        SurrealNode r = Sum(n, node);
        resultSet.Add(r);
    }
}

private static SurrealNode FindSimplest(List<SurrealNode> nodes, Relation relation)
{
    if (nodes.Count == 0)
        return null;
    SurrealNode simplest = nodes[0];
    SurrealNode temp;
    for (int i = 1; i < nodes.Count; i++)
    {
        temp = nodes[i];
        if (relation == Relation.MORE)
        {
            if (simplest.Value < temp.Value)
            {
                simplest = temp;
            }
        }
        else if (relation == Relation.LESS)
        {
            if (simplest.Value > temp.Value)
```

```
            {
                simplest = temp;
            }
        }
    }
    return simplest;
}

static Dictionary<float, SurrealNode> d = new Dictionary<float, SurrealNode>();
public static SurrealNode Sum(SurrealNode a, SurrealNode b)
{
    SurrealNode result;
    if (!d.TryGetValue(a.Value + b.Value, out result))
    {

        if (a.Direction == MoveDirection.ZERO)
            return b;
        if (b.Direction == MoveDirection.ZERO)
            return a;
        List<SurrealNode> a_more = a.GetCut(Relation.MORE);
        List<SurrealNode> a_less = a.GetCut(Relation.LESS);
        List<SurrealNode> b_more = b.GetCut(Relation.MORE);
        List<SurrealNode> b_less = b.GetCut(Relation.LESS);
        List<SurrealNode> leftResult = new List<SurrealNode>();
        SumWithSet(a_less, b, leftResult);
        SumWithSet(b_less, a, leftResult);
```

```
            List<SurrealNode> rigthResult = new List<SurrealNode>();
            SumWithSet(a_more, b, rigthResult);
            SumWithSet(b_more, a, rigthResult);
            SurrealNode left = FindSimplest(leftResult, Relation.MORE);
            SurrealNode rigth = FindSimplest(rigthResult, Relation.LESS);
            result = new SurrealNode();
            result = result.BuildSeparator(left, rigth);
            if(!d.ContainsKey(result.Value))
                d.Add(result.Value, result);
        }
        return result;
    }

    private static void Multiply(List<SurrealNode> a_nodes, SurrealNode a, List<SurrealNode> b_nodes,
SurrealNode b, List<SurrealNode> resultSet)
    {
        for (int i = 0; i < a_nodes.Count; i++)
        {
            for (int j = 0; j < b_nodes.Count; j++)
            {
                SurrealNode x = Product(a_nodes[i], b);
                SurrealNode y = Product(a, b_nodes[j]);
                SurrealNode xy = Sum(x, y);
                SurrealNode z = Product(a_nodes[i], b_nodes[j]);
                z.Invers();
                SurrealNode xyz = Sum(xy, z);
                resultSet.Add(xyz);
```

```
        }
    }
}

public static SurrealNode Product(SurrealNode a, SurrealNode b)
{
    SurrealNode result;
    if (a.Direction == MoveDirection.ZERO || b.Direction == MoveDirection.ZERO)
    {
        SurrealNode n = new SurrealNode();
        return n.BuildTree(0);
    }
    List<SurrealNode> a_more = a.GetCut(Relation.MORE);
    List<SurrealNode> a_less = a.GetCut(Relation.LESS);
    List<SurrealNode> b_more = b.GetCut(Relation.MORE);
    List<SurrealNode> b_less = b.GetCut(Relation.LESS);
    List<SurrealNode> leftResult = new List<SurrealNode>();
    Multiply(a_less, a, b_less, b, leftResult);
    Multiply(a_more, a, b_more, b, leftResult);
    List<SurrealNode> rigthResult = new List<SurrealNode>();
    Multiply(a_less, a, b_more, b, rigthResult);
    Multiply(b_less, b, a_more, a, rigthResult);
    SurrealNode left = FindSimplest(leftResult, Relation.MORE);
    SurrealNode rigth = FindSimplest(rigthResult, Relation.LESS);
    result = new SurrealNode();
    result = result.BuildSeparator(left, rigth);
```

```
        return result;
    }

    public override string ToString()
    {
        string result = "";
        SurrealNode current = this;
        while (current != null)
        {
            switch (current.Direction)
            {
                case MoveDirection.ZERO:
                    result = Z + result;
                    break;
                case MoveDirection.RIGTH:
                    result = ONE + result;
                    break;
                case MoveDirection.LEFT:
                    result = NULL + result;
                    break;
            }
            current = current.Prev;
        }
        return result;
    }
}}
```