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: «Програмний модуль оцінки продуктивності роботи апаратного забезпечення робочої станції»

Автор: Сюркель А.В.

Керівник проєкту: Куц К.Б.

Нормконтролер: Кругляк В.М.

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

Сюркель Артем Валерійович

1. Тема роботи: «Програмний модуль оцінки продуктивності роботи апаратного забезпечення робочої станції»

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

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

3. Вихідні дані: Розробити програмний модуль для виведення інформації про ступінь навантаженості та продуктивність роботи апаратного забезпечення робочої станції на прикладі процесора, оперативної пам'яті, відеокарти та жорсткого диску.

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 _____ Сюркель А.В.

(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

Продуктивність - це міра ефективності роботи. При цьому даний показник використовується як для оцінки виконання поставлених завдань персоналом фірми або підприємства, так і для функціонування верстатів, персональних комп'ютерів, їх складових частин та окремого програмного забезпечення. Зазвичай під продуктивністю розуміють кількість продукції або обсяг переробляється за годину, хвилину або секунду. Обернена до неї величина - трудомісткість - відображає час, який витрачається на випуск або ж аналіз даних.

Ключовим питанням на порядку денному будь-якого підприємства є зростання продуктивності праці, тобто скорочення витрат часу на виготовлення продукції і збільшення обсягу без додаткових витрат на наймання нових робітників. Тому стратегія і засновані на ній цілі і завдання повинні враховувати основні резерви її підвищення і фактори, що стимулюють персонал працювати краще в якісному і кількісному аспекті. Без цього ніякі конкурентні переваги не зможуть зробити підприємство лідером в галузі.

Основа ведення будь-якого бізнесу - це раціональне і ефективне використання наявних ресурсів, в тому числі і праці. Цілком логічно, що менеджмент прагне збільшити обсяг продукції, що випускається без додаткових витрат на наймання працівників. Експерти виділяють кілька факторів, які дозволяють поліпшити продуктивність: Управлінський стиль (головне завдання керівника - мотивувати персонал, створити організаційну культуру, в якій цінується активність і працьовитість). Інвестиції в технічні інновації (покупка нового устаткування, що відповідає запитам часу, дозволяє значно скоротити тимчасові витрати кожного працівника).

1 Загальна частина

1.1 Постановка задачі

Індекс продуктивності (Windows Experience Index, WEI) - це оцінка самої системи Windows основних характеристик комп'ютера, на якому встановлена ОС.

Основні характеристики, це:

1) процесор;
2) оперативна пам'ять;
3) графіка;
4) графіка для ігор;
5) дискова підсистема.

Загальна оцінка ПК показує найменшу продуктивність обладнання в загальному, з огляду на можливості окремих елементів. Проводиться аналіз швидкості роботи центрального процесора (ЦП), оперативної пам'яті (ОЗУ), вінчестера і графічної карти з урахуванням потреб 3D графіки і анімації робочого столу. Подивитися дану інформацію можна як за допомогою сторонніх програмних рішень, так і за допомогою стандартних можливостей.

Технічні вимоги до продукту наступні:

1) програмний продукт повинен функціонувати на персональних комп‘ютерах із стандартним набором компонент;

2) забезпечувати високу швидкість обробки даних та відклик користувачеві у реальному часі;

3) забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;

4) передбачати мінімальні витрати на впровадження програмного продукту.

Мінімальні системні вимоги до апаратного забезпечення та платформа:

1) апаратна платформа: x86;
2) програмна платформи: Windows 7.
3) відеокарта з 1 Гб відеопам'яті;
4) процесор з частотою 2,0 ГГц і 2 ядрами;
5) 2 Гб оперативної пам'яті;
6) 2 Гб місця на жорсткому диску.

1.2 Теоретичні відомості

Продуктивність пристрою - величина дії пристрою, тобто відношення кількості виробленої роботи (випущеного продукту) до часу їх виконання (випуску), обсяг продукції (роботи), виробленої в одиницю часу даним устаткуванням відповідно до його конструктивними особливостями, технічною характеристикою та виробничої кваліфікацією робітників.

Обчислювальна потужність комп'ютера (продуктивність комп'ютера) - це кількісна характеристика швидкості виконання певних операцій на комп'ютері. Найчастіше обчислювальна потужність вимірюється у флопс (кількість операцій з плаваючою комою в секунду), а також похідними від неї. На даний момент прийнято зараховувати до суперкомп'ютерів системи з обчислювальною потужністю більше 10 терафлопс (10 * 1012 чи десять трильйонів флопс; для порівняння - середньостатистичний сучасний настільний комп'ютер має продуктивність близько 0,1 терафлопса). Одна з найбільш потужних на тесті HPL комп'ютерних систем - китайський Sunway TaihuLight - має продуктивність, що перевищує кілька десятків петафлопсів.

Існує кілька складнощів при визначенні обчислювальної потужності суперкомп'ютера. По-перше, слід мати на увазі, що продуктивність системи може сильно залежати від типу виконуваної завдання. Зокрема, негативно позначається

на обчислювальної потужності необхідність частого обміну даних між складовими комп'ютерної системи, а також часте звертання до пам'яті. У зв'язку з цим виділяють пікову обчислювальну потужність - гіпотетично максимально можливу кількість операцій над числами з плаваючою комою в секунду, яке здатний зробити даний суперкомп'ютер.

Важливу роль відіграє також розрядність значень, що обробляються програмою (зазвичай мається на увазі формат чисел з плаваючою комою). Так, наприклад, у графічних процесорів NVIDIA Tesla перших двох поколінь максимальна продуктивність в режимі одинарної точності (32 біт) становить близько 1 терафлопса, однак при проведенні обчислень з подвійною точністю (64 біт) вона в 10 разів нижче (так, в мікросхемах серії GF200 в 10 разів менше блоків з підтримкою обчислень з подвійною точністю).

Оцінка реальної обчислювальної потужності виробляється шляхом проходження спеціальних тестів (бенчмарков) - набору програм, спеціально призначених для проведення обчислень і вимірювання часу їх виконання. Зазвичай оцінюється швидкість рішення системою великої системи лінійних алгебраїчних рівнянь, що обумовлюється, в першу чергу, хорошою масштабністю цього завдання.

Найбільш популярним тестом продуктивності є Linpack benchmark. Зокрема, HPL (високопараллельних реалізація Linpack із застосуванням MPI) використовується при складанні списку TOP500 суперкомп'ютерів у світі.

Іншими популярними програмами для проведення тестування є NAMD (рішення задач молекулярної динаміки), HPCC (HPC Challenge Benchmark), NAS Parallel Benchmarks.

Багато виробників оснащують нові комп'ютери програмами, які не потрібні користувачам. Це можуть бути пробні і обмежені за часом версії програм, які надаються розробниками в надії, що користувачі знайдуть їх корисними і придбають повні або нові версії. Ці програми можуть уповільнювати роботу комп'ютера, оскільки витрачається пам'ять, місце на диску і ресурси процесора.

Рекомендуємо видалити всі програми, які ви не плануєте використовувати, в тому числі ПЗ, встановлене виробником або навіть вами самими, яке вам більше не потрібно, особливо службові програми для адміністрування і настройки обладнання та програмного забезпечення комп'ютера. Службові програми, такі як антивірусні сканери, засоби очищення диска і програми резервного копіювання, часто запускаються автоматично при запуску системи і непомітно для вас працюють у фоновому режимі. Багато хто навіть не знають, що такі програми запущені.

Навіть на старому комп'ютері можуть виявитися встановлені виробником програми, які ви ніколи не помічали або просто про них забули. Ніколи не пізно видалити все зайве і звільнити ресурси системи. Можливо, якусь програму ви планували використовувати, але цього не сталося. Якщо видалити її, комп'ютер зможе працювати швидше.

Обмеження програм, що запускаються при завантаженні

Багато програм запускаються автоматично при завантаженні Windows. Розробники часто налаштовують програми для роботи у фоновому режимі, непомітно для користувача, щоб вони відкривалися відразу при натисканні значка. Це зручно для часто використовуваних програм, але якщо програма запускається рідко або зовсім не використовується, то при такому підході марно витрачається пам'ять і сповільнюється завантаження Windows.

Як дізнатися, які програми запускаються автоматично при завантаженні системи? Іноді це очевидно, оскільки програма додає значок в область сповіщень на панелі завдань. Перевірте в області повідомлень програми, які не потрібно запускати автоматично. Наведіть покажчик на значок, щоб дізнатися назву програми. Щоб показати всі значки, натисніть кнопку Відображати приховані значки.

Деякі програми, автоматично запускаються при завантаженні, можуть не відображатися в області сповіщень. Безкоштовна програма AutoRuns для Windows, яку можна завантажити з веб-сайту Майкрософт, показує всі програми і

процеси, які запускаються при завантаженні Windows. Щоб заборонити автоматичний запуск програми при завантаженні Windows, відкрийте програму AutoRuns for Windows і зніміть прапорець поруч із назвою програми, яка не повинна запускатися. Програма AutoRuns для Windows призначена для досвідчених користувачів.

Дефрагментація жорсткого диска. В результаті фрагментації збільшується число операцій з жорстким диском, що може уповільнити роботу комп'ютера. Програма дефрагментації диска впорядковує фрагментовані дані, підвищуючи ефективність роботи жорсткого диска. Програма дефрагментації працює за розкладом, але можна запустити дефрагментацію жорсткого диска вручну.

Очищення жорсткого диска. Непотрібні файли займають місце на жорсткому диску і можуть уповільнити роботу комп'ютера. Програма очищення диска видаляє тимчасові файли, очищує кошик і видаляє різноманітні системні файли і інші непотрібні елементи.

Обмеження числа одночасно працюючих програм. Іноді зміна правил роботи з комп'ютером може значно підвищити продуктивність. Якщо у вас весь час відкрито вісім програм, з десяток вікон браузера і при цьому йде активний обмін повідомленнями, - не дивуйтеся, що ваш комп'ютер працює повільно. Одночасно відкриті повідомлення електронної пошти також витрачають пам'ять.

Якщо комп'ютер працює повільно, подумайте, чи дійсно потрібно тримати відкритими всі програми і вікна. Наприклад, є й інші способи нагадувати собі про те, що потрібно відповісти на листи.

Переконайтеся, що працює тільки одна антивірусна програма. Одночасне функціонування декількох антивірусних програм може сповільнити роботу комп'ютера. Центр підтримки виводить повідомлення про роботу кількох антивірусних програм і дозволяє усунути проблему.

Відключення візуальних ефектів. Якщо Windows працює повільно, можна прискорити роботу, відключивши деякі візуальні ефекти. Потрібно підібрати оптимальне співвідношення ефективності і ефектності. Якщо комп'ютер досить

швидкий, то немає необхідності жертвувати візуальними ефектами, але якщо продуктивності ледь вистачає для запуску Windows 7, то краще відмовитися від надмірностей.

Можна відключати візуальні ефекти самостійно, окремо, або скористатися підказкою Windows. Ви можете контролювати 20 візуальних ефектів: ефект прозорості, відкриття і закриття меню, відображення тіней і інше.

Перезапускайте комп'ютер не рідше разу на тиждень, особливо якщо він інтенсивно використовується. Перезапуск дозволяє очистити пам'ять і завершити помилкові процеси і служби, які почали роботу.

Під час перезапуску закриваються всі програми, що працюють на комп'ютері (не тільки ті, які відображаються на панелі завдань, але також десятки служб, які могли бути запущені різними програмами і не були зупинені). Перезапуск може усунути незрозумілі проблеми з продуктивністю, коли складно встановити їх причину.

Якщо відкрито так багато програм, повідомлень електронної пошти та веб-сайтів, що перезапуск представляється проблемою, то це серйозний привід все-таки перезапустити комп'ютер. Чим більше відкрито програм і чим довше вони працюють, тим вище шанси, що комп'ютер буде працювати все повільніше і в кінці кінців утворюється брак пам'яті.

Якщо комп'ютер під керуванням Windows 7 працює надто повільно, зазвичай це обумовлено браком оперативної пам'яті. Кращим способом прискорити роботу стане розширення пам'яті.

Windows 7 може працювати на комп'ютері з 1 ГБ оперативної пам'яті, але з 2 ГБ вона буде працювати краще. Для оптимальної продуктивності рекомендуємо розширити пам'ять до 3 ГБ і більше.

Ще один варіант розширення пам'яті - технологія Windows ReadyBoost. Ця функція дозволяє використовувати знімні носії, в тому числі USB флеш-пам'яті, для прискорення роботи комп'ютера. Вставити флеш-пам'ять в USB-порт значно

простіше, ніж відкривати корпус комп'ютера і вставляти модулі пам'яті в системну плату.

Перевірка вірусів і шпигунського ПЗ. Якщо комп'ютер працює повільно, то можливо, що він заражений вірусами або шпигунським ПО. Ця проблема зустрічається рідше інших, але про неї слід пам'ятати. Щоб не турбуватися, перевірте комп'ютер за допомогою антишпигунських і антивірусних програм.

Сильне зниження продуктивності комп'ютера - поширений симптом вірусу. До інших ознак слід зазначити поява несподіваних спливаючих повідомлень, самостійний запуск програм і звук постійно діючого жорсткого диска.

Шпигунської називається програма, яка зазвичай встановлюється без вашого відома і відстежує ваші дії в Інтернеті. Перевірити наявність шпигунських програм забезпечення можна за допомогою Windows або інших антишпигунських програм.

Кращий спосіб боротьби з вірусами - профілактика. Завжди запускайте антивірусну програму і підтримуйте її актуальність. На жаль, зараження комп'ютера можливо навіть при дотриманні цих запобіжних заходів.

Перевірка швидкості роботи комп'ютера. Якщо після виконання наведених рекомендацій комп'ютер все одно працює занадто повільно, може знадобитися придбати новий комп'ютер або модернізувати обладнання, наприклад встановити новий жорсткий диск або швидший відеоадаптер. Швидкість комп'ютера не потрібно оцінювати "на око". Windows дозволяє виміряти і оцінити швидкість роботи комп'ютера за допомогою індексу продуктивності Windows.

Індекс продуктивності Windows оцінює п'ять найважливіших компонентів комп'ютера і дає числову оцінку кожного компонента, а також загальну оцінку. Загальною оцінкою вважається мінімальна з оцінок за окремими компонентами. У даній версії загальна оцінка може мати значення від 1,0 до 7,9. Якщо комп'ютер отримав оцінку менше 2 або 3, то в залежності від планованих завдань може знадобитися придбати новий комп'ютер.

1.3 Обґрунтування вибору мови програмування

Програма реалізована в інтегрованому середовищі розробки Microsoft Visual Studio на мові програмування C#.

Microsoft Visual Studio допомагає індивідуальним програмістам і невеликим групам, що створюють будь-які види програмного забезпечення, прискорити розробку додатків і створення призначених для користувача інтерфейсів з принципово новим рівнем зручності, підвищити ефективність колективної роботи.

Visual Studio допомагає писати код швидше, підтримуючи безліч засобів і можливостей, які підвищують продуктивність праці.

Технологія Intellisense є різновидом авто завершення: як тільки ви вводите ім'я класу або об'єкту і ставите крапку, показується список доступних членів даного класу або об'єкту. Це прискорює кодування, оскільки зменшується кількість тексту, що набирає на клавіатурі.

Візуальні конструктори Visual Studio дозволяють створювати потужні і привабливі застосування, засновані на Windows Presentation Foundation, — графічній підсистемі. C # - це подієво-керована, повністю об’єктно-орієнтована мова візуального програмування, в якій програми створюються за допомогою інтегрованого середовища розробки (Integrated Development Environment, IDE). У цьому середовищі програміст може писати, запускати, тестувати і налагоджувати програми, написані на C #. Процес оперативного створення програмних додатків за допомогою IDE називається швидкої розробкою додатків (Rapid Application Development, RAD).

C # призначена для створення переносимого коду. Всі змінні автоматично ініціалізуються середовищем і мають типову захищеність, що дозволяє уникнути невизначених ситуацій у разі відсутності ініціалізації, зміни в об’єкті або спроби виконати недопустиме перетворення типів.

У C # уніфікована система типів: кожен тип розглядається як об’єкт. Об’єкти зібрані в простір імен (namespaces), який дозволяє програмно звертатися

до чого-небудь. C # дозволяє використовувати типізовані, розширювані метадані, які можуть бути прикріплені до об'єкта. Архітектурою проекту можуть визначатись локальні атрибути, які будуть пов'язані з будь-якими елементами мови - класами, інтерфейсами і т.д.

Перевагами використання мови C# є:

1) підтримки корпорацією Майкрософт. На відміну від Java, якої не пішов на користь перехід у власність Oracle, C# добре розвивається завдяки зусиллям Microsoft;

2) останнім часом багато вдосконалюється. Так як C # був створений пізніше, ніж Java та іншими мовами, то потрібно дуже багато доопрацювати. Також це стосується популяризації і безкоштовності - було обіцяно відкрити вихідний код, а інструменти (Visual Studio, Xamarin) стали безкоштовними для приватних осіб і невеликих компаній;

3) багато конструкцій, які створені для полегшення написання і розуміння коду (особливо якщо це код іншого програміста) і не грають ролі при компіляції;

4) середній поріг входження. Синтаксис схожий на C, C ++ або Java полегшує перехід для інших програмістів. Для новачків це також один з найперспективніших мов для вивчення;

5) додано функціональне програмування (F #);

6) велика спільнота програмістів;

7) багато вакансій на посаду C # програміста в будь-якому регіоні.

Незначними недоліками є:

1) орієнтованість, в основному, тільки на .NET (на Windows платформу);

2) безкоштовність тільки для невеликих компанії, учнів і програмістів-одинаків. Для великих команд покупка ліцензій обійдеться недешево;

3) зберегли оператор go to.

2 Спеціальна частина

2.1 Опис алгоритму створення програмного засобу

Для розробки програмного засобу потрібно пройти низку етапів, таких як:

1) замовлення;

2) поставка;

3) розробка;

4) експлуатація;

5) супровід.

Блок-схема загальної роботи програмного засобу, виконання основних функцій, зображена на рисунку 2.1.

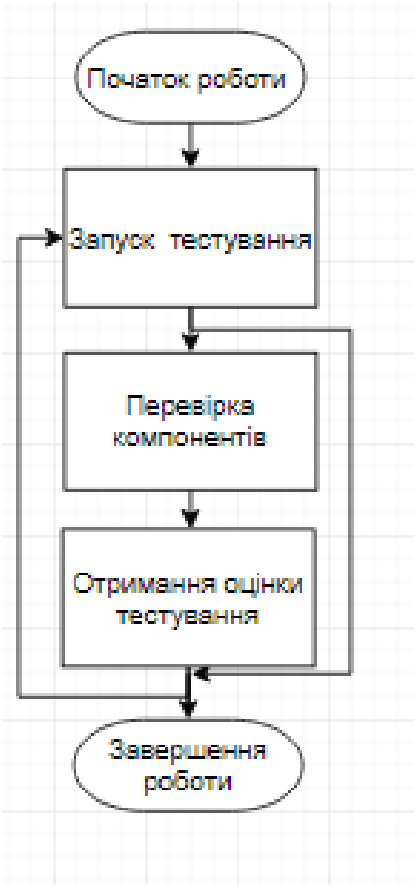

Рисунок 2.1 – Блок-схема загального алгоритму роботи програми

=

Етап проектування ПЗ передбачає створення наступних діаграм:

1) діаграма класів;

2) діаграма послідовності;

3) діаграма станів.

Діаграма класів представляє статичну структуру ПЗ, відображає класи, типи даних, їх зміст та відношення.

Рисунок 2.2 –Діаграма класів даного програмного засобу

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень.

Іншими словами, діаграма послідовностей відображає часові особливості передачі і прийому повідомлень об'єктами.

На діаграмі послідовності зображаються тільки ті об'єкти, які безпосередньо беруть участь у взаємодії.

Логіка виконання дій для кожного із процесів, зображених на діаграмах послідовності, описана раніше.

Діаграма послідовності виглядає наступним чином:

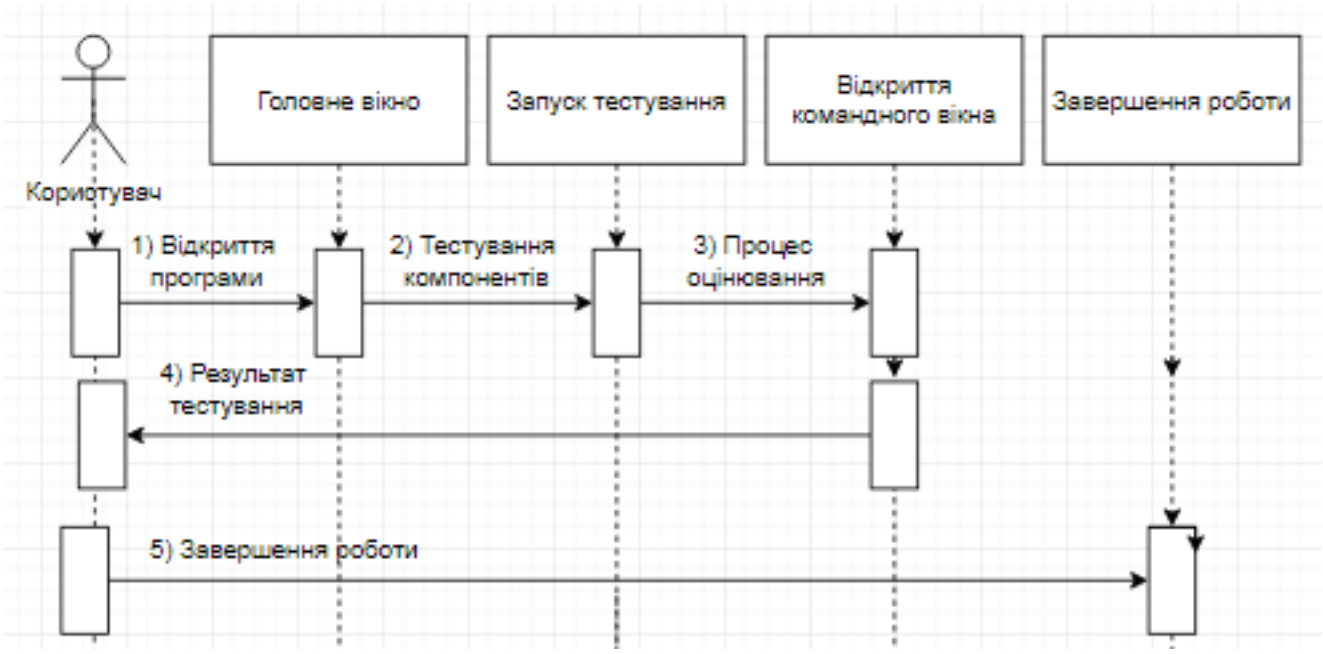

Рисунок 2.3 – Діаграма послідовності користувача програмного засобу

Після діаграми послідовності доречно створити діаграму станів програми.

Діаграма станів користувача показує стани програмного забезпечення в якому може перебувати проектоване програмне забезпечення та переходи між ними. Діаграма станів, зображена на рисунку 2.4 має три основні функціональні стани «Запуск тестування», «Процес тестування» та «Отримання результатів».

Рисунок 2.4 - Діаграма станів користувача

Діаграма станів також показує, що після процесу тестування можливий перехід до стану «Завершення роботи», а після стану «Отримання результату» повернення до запуску програми.

2.2 Опис засобів реалізації

StringBuilder надає змінну рядок символів. Цей клас не успадковується. Цей клас об'єкт, подібний рядку, значення якого є змінною послідовністю символів.

List <T> являє строго типізований список об'єктів, доступних за індексом. Підтримує методи для пошуку за списком, виконання сортування і інших операцій зі списками.

Клас List <T> є універсальним еквівалентом класу ArrayList. Він реалізує IList <T> універсальний інтерфейс за допомогою масиву, розмір якого динамічно збільшується в міру необхідності.

Елементи можна додавати в List <T> за допомогою методів Add або AddRange.

Клас List <T> використовує як компаратор перевірки на рівність, так і компаратор упорядкування.

Методи, такі як Contains, IndexOf, LastIndexOfi Remove, використовують компаратор перевірки на рівність для елементів списку. Компаратор перевірки на рівність за замовчуванням для типу T визначається наступним чином. Якщо тип T реалізує IEquatable <T> універсальний інтерфейс, то функція порівняння на рівність є методом Equals (T) цього інтерфейсу. в іншому випадку компаратор перевірки на рівність за замовчуванням Object.Equals (Object).

Методи, такі як BinarySearch і Sort, використовують компаратор упорядкування для елементів списку. Компаратор за замовчуванням для типу T визначається наступним чином. Якщо тип T реалізує IComparable <T> універсальний інтерфейс, то компаратором за замовчуванням є метод CompareTo (T) цього інтерфейсу. в іншому випадку, якщо тип T реалізує неуніверсальність

IComparable інтерфейс, то компаратором за замовчуванням є метод CompareTo (Object) цього інтерфейсу. Якщо тип T реалізує ні інтерфейс, ні компаратор за замовчуванням, ні компаратор, ні делегат порівняння повинні бути надані явно.

Сортування List <T> Не гарантується. Необхідно впорядкувати List <T> перед виконанням операцій (наприклад, BinarySearch), для яких потрібно сортування List <T>.

Доступ до елементів в цій колекції можна отримати за допомогою цілочисельного індексу. Індекси в цій колекції відраховуються від нуля.

Тільки .NET Framework: Для дуже великих List <T> об'єктів можна збільшити максимальну ємність до 2 000 000 000 елементів в 64-розрядної системі, встановивши атрибут enabled елемента конфігурації <gcAllowVeryLargeObjects> для true в середовищі виконання.

List <T> приймає null як допустиме значення для посилальних типів і допускає дублювання елементів.

BackgroundWorker виконує операцію в окремому потоці.

Клас BackgroundWorker дозволяє виконувати операцію в окремому виділеному потоці. Тривалі операції, такі як завантаження і транзакції бази даних, можуть привести до того, що призначений для користувача інтерфейс буде виглядати так, ніби він перестає відповідати на запити під час роботи. Якщо вам потрібен призначений для користувача інтерфейс для реагування і ви зіткнулися з довгими затримками, пов'язаними з такими операціями, клас BackgroundWorker надає зручне рішення.

Щоб виконати трудомістку операцію в фоновому режимі, створіть BackgroundWorker і прослухайте події, які повідомляють про хід виконання операції і подають сигнал про завершення операції. Можна створити BackgroundWorker програмним способом або перетягнути його на форму з вкладки компоненти панелі елементів. Якщо ви створите BackgroundWorker в конструктор Windows Forms, він з'явиться в області компонентів, а його властивості будуть відображатися в вікно властивостей.

MarshalAsAttribute вказує спосіб маршалинга даних між керованим і некерованим кодом.

Цей атрибут можна застосувати до параметрів, полях або повертається значенням.

Цей атрибут є необов'язковим, тому що кожен тип даних має поведінка маршалинга за замовчуванням. Цей атрибут необхідний тільки в тому випадку, якщо даний тип може бути упакований в кілька типів. Наприклад, можна маршаліровать рядок в некерований код як LPStr, LPWStr, LPTStrілі BStr. За замовчуванням середу CLR маршалірует строковий параметр як BStr в методи COM. Можна застосувати атрибут MarshalAsAttribute до окремого полю або параметру, щоб при цьому конкретна рядок була упакована як LPStr, а не BStr. Програма Tlbexp. exe (засіб експорту бібліотек типів) передає параметри маршалірованія в середу CLR.

XmlReader надає засіб читання, що забезпечує швидкий прямий доступ (без кешування) до даних XML.

XmlReader надає доступ до XML-даними в документі або потоці тільки для читання вперед. Цей клас відповідає стандартам W3C мова XML (XML) 1,0 (Четвертий випуск) і просторів імен в рекомендаціях по XML 1,0 (третій випуск).

XmlReader методи дозволяють переміщатися по XML-даними і читати вміст сайту. Властивості класу відображати значення поточного вузла, де розташований модуль читання. Значення властивості ReadState вказує поточний стан модуля читання XML. Наприклад, властивість встановлюється в ReadState.Initial методом XmlReader.Read і ReadState.Closedся методом XmlReader.Close. XmlReader також забезпечує перевірку узгодженості даних і перевірку DTD або схеми.

XmlReader використовує модель вилучення для отримання даних

Щоб вказати набір компонентів, які необхідно включити в модулі читання XML, передайте об'єкт System.Xml.XmlReaderSettings в метод Create. Ви можете використовувати один об'єкт System.Xml.XmlReaderSettings, щоб створити кілька модулів читання з однаковими функціональними можливостями або змінити

об'єкт System.Xml.XmlReaderSettings, щоб створити новий засіб читання з іншим набором компонентів. Можна також легко додати компоненти в існуючий модуль читання.

Для візуалізації елементів використовуються такі елементи, як Panel, Label, Button.

Таблиця 2.1 – Елементи візуалізації програми

Елемент	**Опис**
Panel	Використовується, щоб згрупувати колекції елементів управління.
Label	Являє стандартну мітку Windows.
Button	Представляє елемент керування "кнопка Windows".

2.3 Порівняльний аналіз реалізованого програмного засобу та програм-аналогів

Продуктивність програмних засобів залежить від їх компонентів. Для тестування продуктивності різних компонентів використовують наступні програми:

Деякими із них є:

1) «3DMark Basic Edition»;
2) «AnVir Task Manager»;
3) «Speccy».

«3DMark Basic Edition» — безкоштовна версія програми для перевірки рівня продуктивності апаратної частини комп’ютера (в першу чергу — його відеокарти). Включає в себе тести продуктивності, що імітують реальну гру, синтетичні тести можливостей відеокарти, тести центрального процесора і пам’яті.

По завершенні тестування системи 3DMark Basic Edition виводить необхідну інформацію про частоти, температури CPU, GPU і інші важливі інформаційні дані системи на сайті програми. Також є можливість порівняти результати тестування свого комп'ютера з даними інших користувачів 3DMark. Приклад роботи показаний на рисунку 2.5.

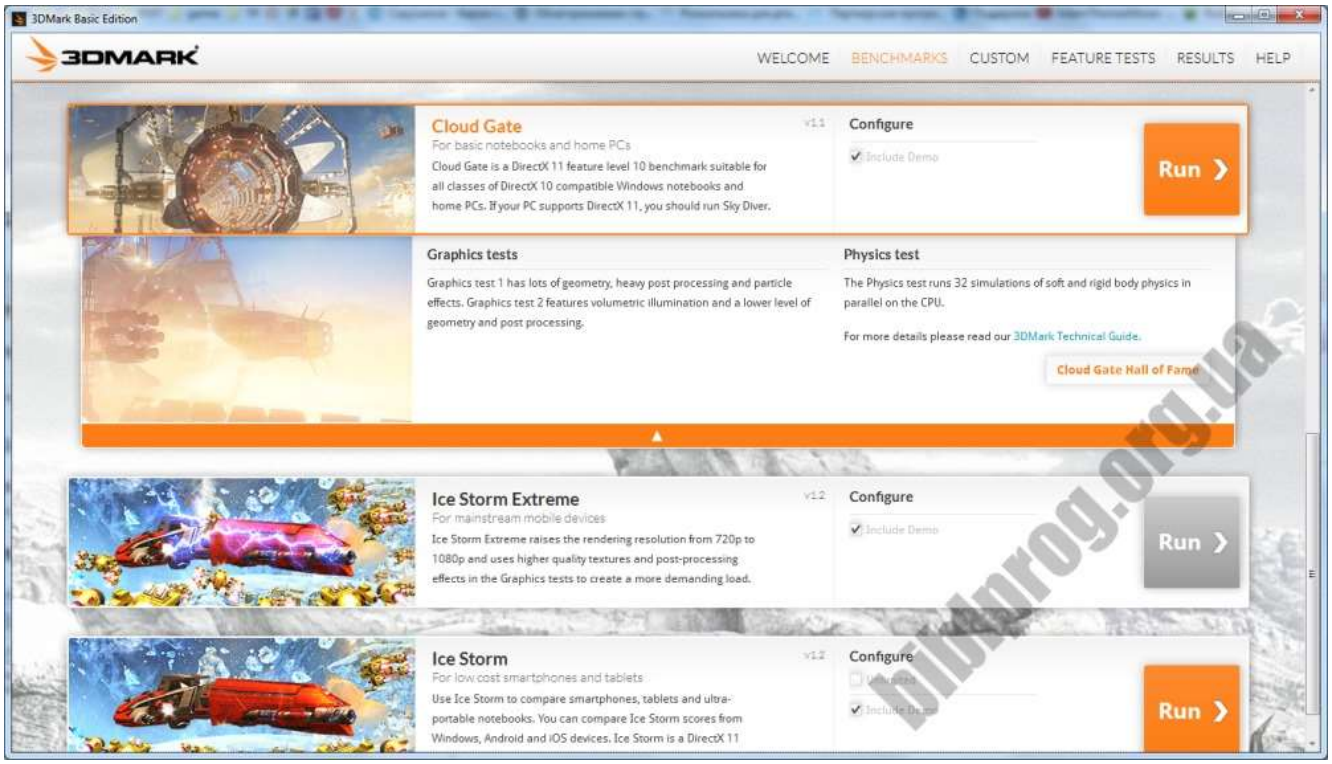

Рисунок 2.5 – ПЗ «3DMark Basic Edition»

«AnVir Task Manager» — безкоштовна системна утиліта, що дозволяє контролювати все, що запущено на комп'ютері, а також пропонує інші зручні інструменти для налаштування комп'ютера.

Унікальна властивість AnVir Task Manager полягає в тому, що він надає детальну інформацію про роботу системи і, а також має велику кількість інструментів для управління Windows. Для того, щоб замінити всі функції AnVir Task Manager, доведеться встановити близько 10 різних програм.

Основні характеристики anvir task manager:

1) управління автозавантаженням, запущеними процесами, сервісами і драйверами;

2) заміна диспетчера завдань;

3) виявлення і видалення вірусів та шпигунських програм;

4) тонке налаштування xp, vista і windows 7, включно зі встановленням прихованих налаштувань;

5) прискорення завантаження windows і роботи комп’ютера. Приклад роботи зображений на рисунку 2.6.

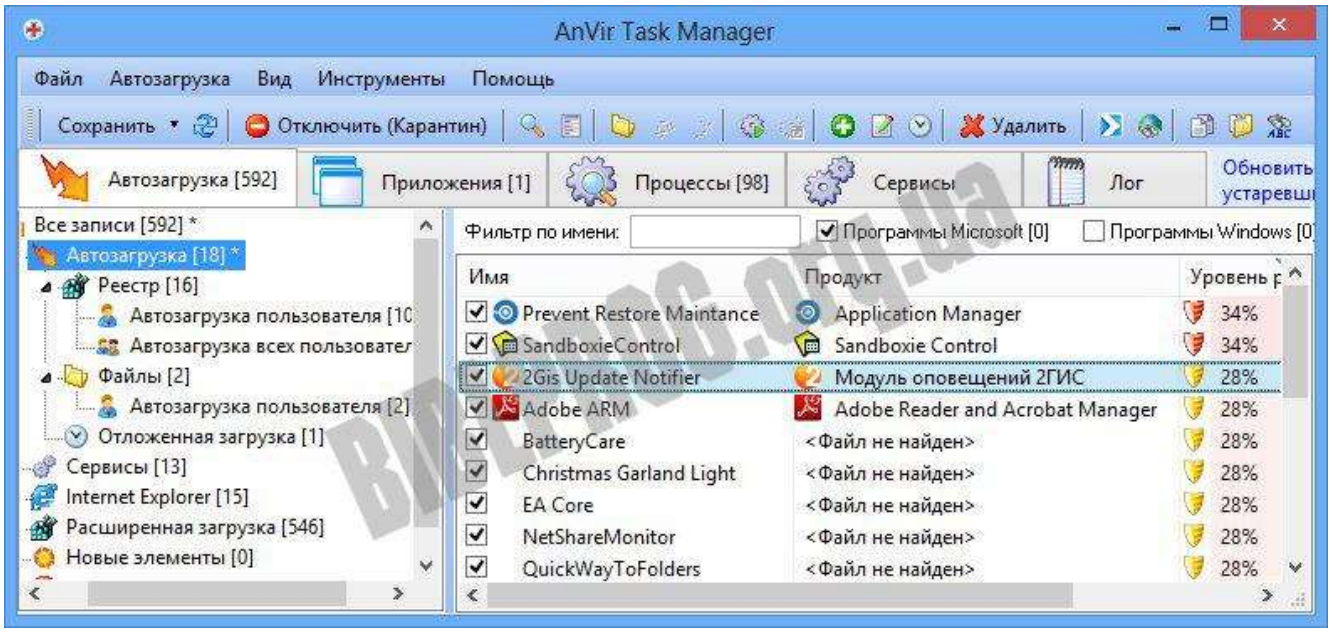

Рисунок 2.6 – ПЗ «AnVir Task Manager»

«Speccy» — безкоштовна програма, для отримання докладної інформації про апаратне забезпечення комп’ютера. Під час запуску Speccy сканує апаратну частину комп’ютера і відображає інформацію про операційну систему і характеристики встановленого «заліза».

За допомогою Speccy користувач може дізнатися всі дані про процесор, материнську плату, оперативну пам’ять, графічну карту, жорсткі та оптичні диски, аудіокарту і т.д. Крім цього, за наявності спеціальних датчиків, ця програма показує поточну температура модулів, які підтримують дану опцію. Приклад роботи на рисунку 2.7.

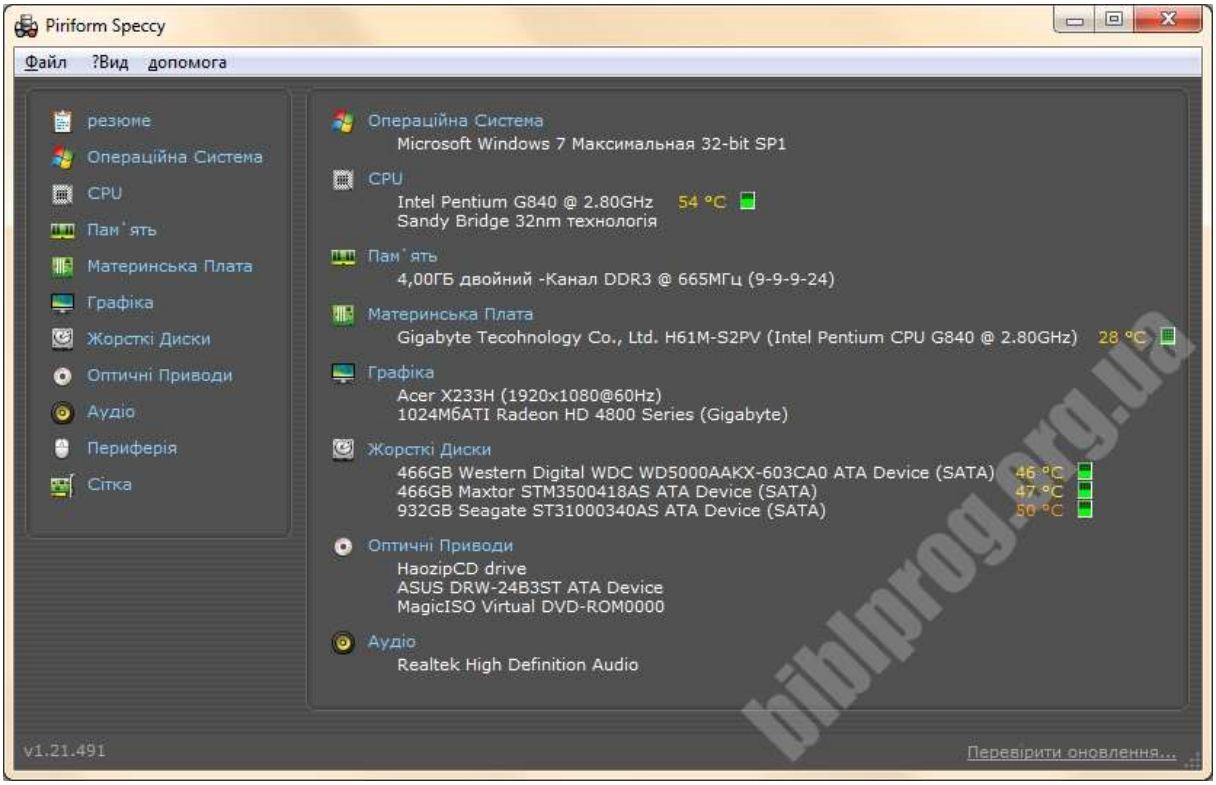

Рисунок 2.7 – ПЗ «Speccy»

Створена програма «Win Bench» є зручним багатофункціональним програмним засобом, який містить основні функції будь-якого засобу оцінки продуктивності. Програма дозволяє виконувати тестування компонентів. Поле програми зображене на рисунку 2.8.

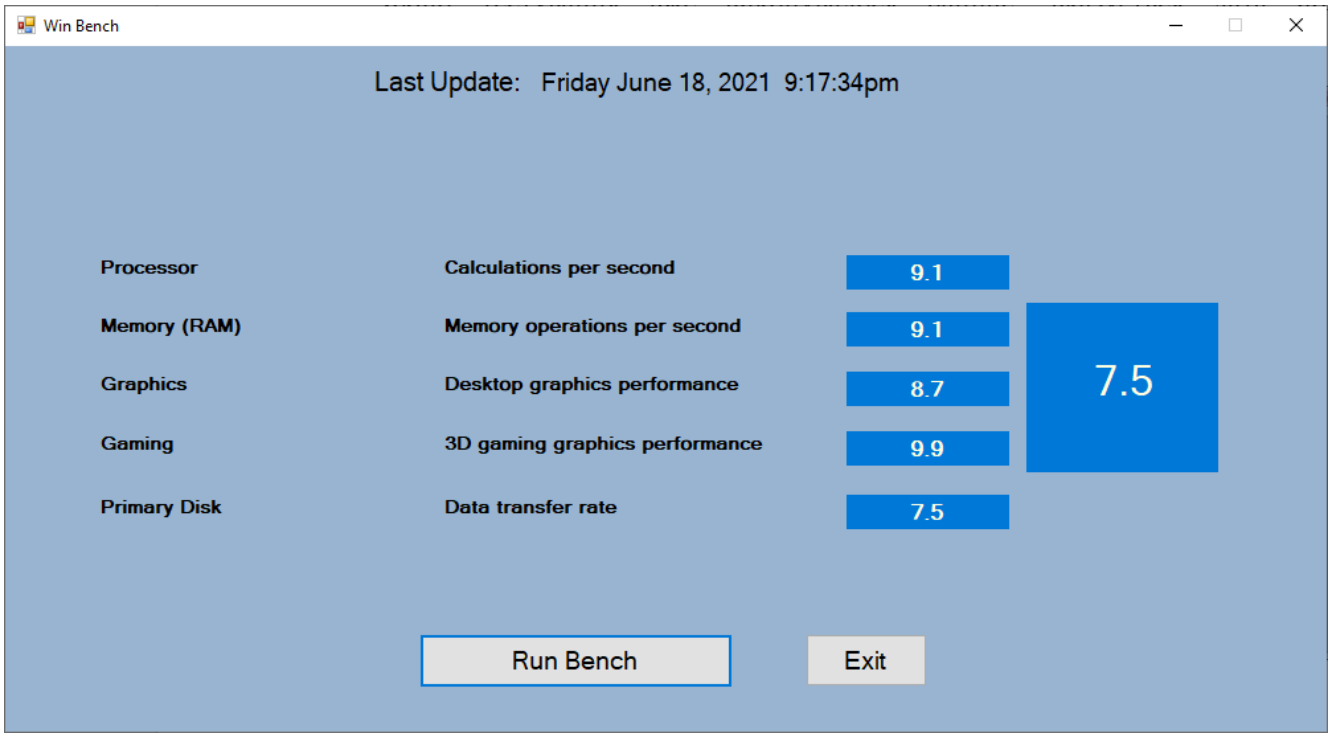

Рисунок 2.8 – Створений редактор «Win Bench»

Детальне функціональне порівняння ігор у таблиці 2.2.

Таблиця 2.2 – Порівняння ПЗ

	«3DMark Basic Edition»	**«AnVir Task Manager»**	**«Speccy»**	**«Win Bench»**
Інтерфейс	Зручний	Зручний	Незручний	Зручний
Швидкість тестування	Висока	Середня	Низька	Висока
Зручність керування	Зручно	Неручно	Неручно	Зручно
Якість зображення	Висока	Середня	Середня	Висока
Розмір програми	Компактна	Некомпактна	Компактна	Компактна

Після детального порівняння створеного програмного засобу із існуючими аналогами можна зробити висновок, що він має багато переваг. Створена програма дозволяє користувачеві проводити оцінку продуктивності комп’ютера. Програмний засіб є зручним, зрозумілим у використанні із приємним інтерфейсом та багатофункціональним. Перевагами створеного програмного засобу є його швидкість, доступність, компактність, зручність та функціональність.

2.4 Інструкція роботи користувача

Для відкриття програми потрібно клацнути двічі на ярлик програми (рисунок 2.9).

Рисунок 2.9 – Ярлик програми

Після запуску програми з’явиться головне вікно із описом основних компонентів, зображене на рисунку 2.10.

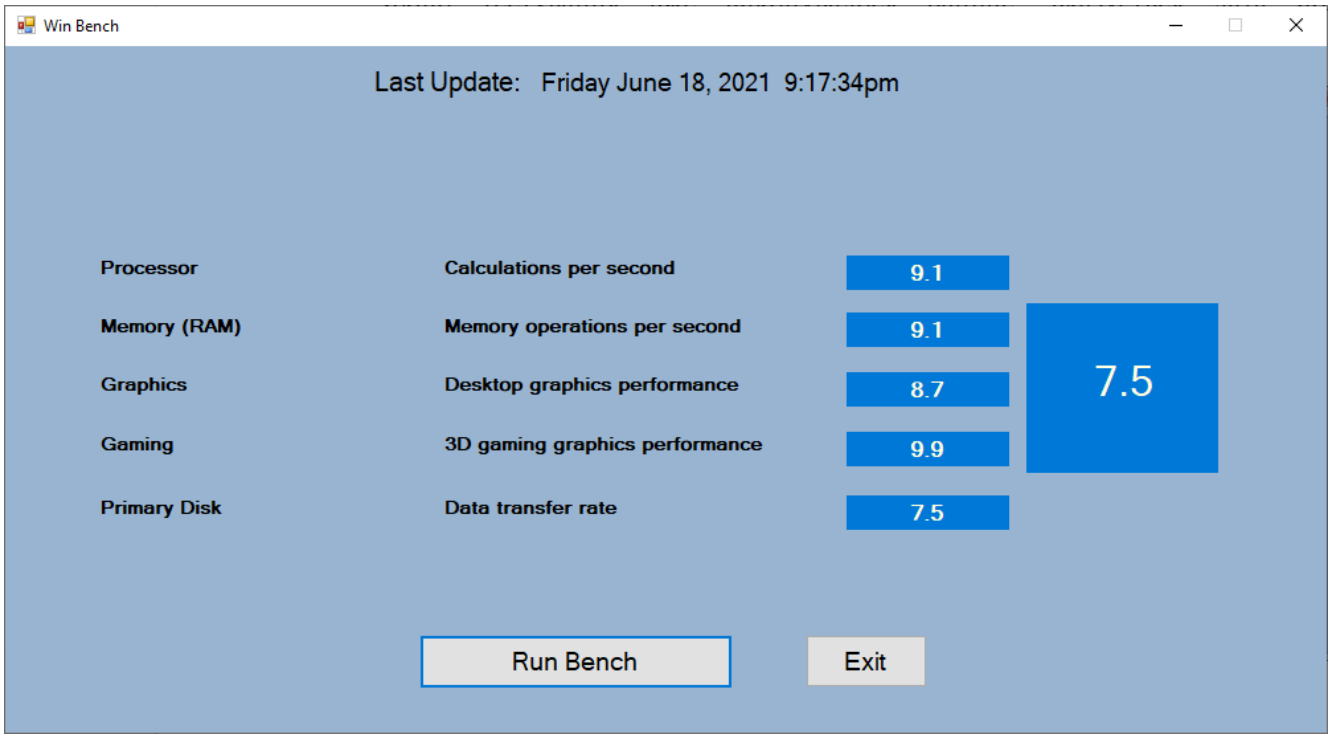

Рисунок 2.10 – Головне вікно

Для запуску тестування потрібно натиснути кнопку «Run Bench». (рисунок 2.11).

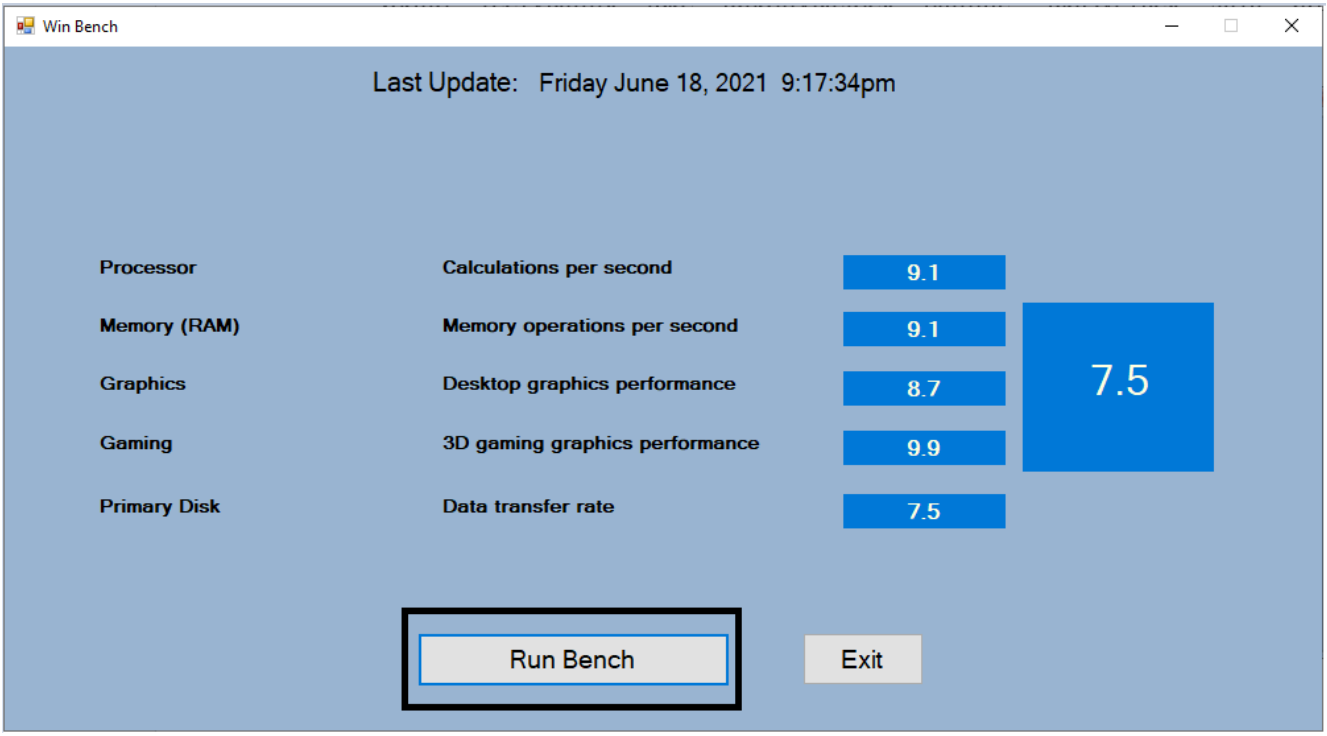

Рисунок 2.11 – Запуск тестування

Під час тестування з’являється командне вікно та операційна система переходить на спрощенний стиль (рисунок 2.12).

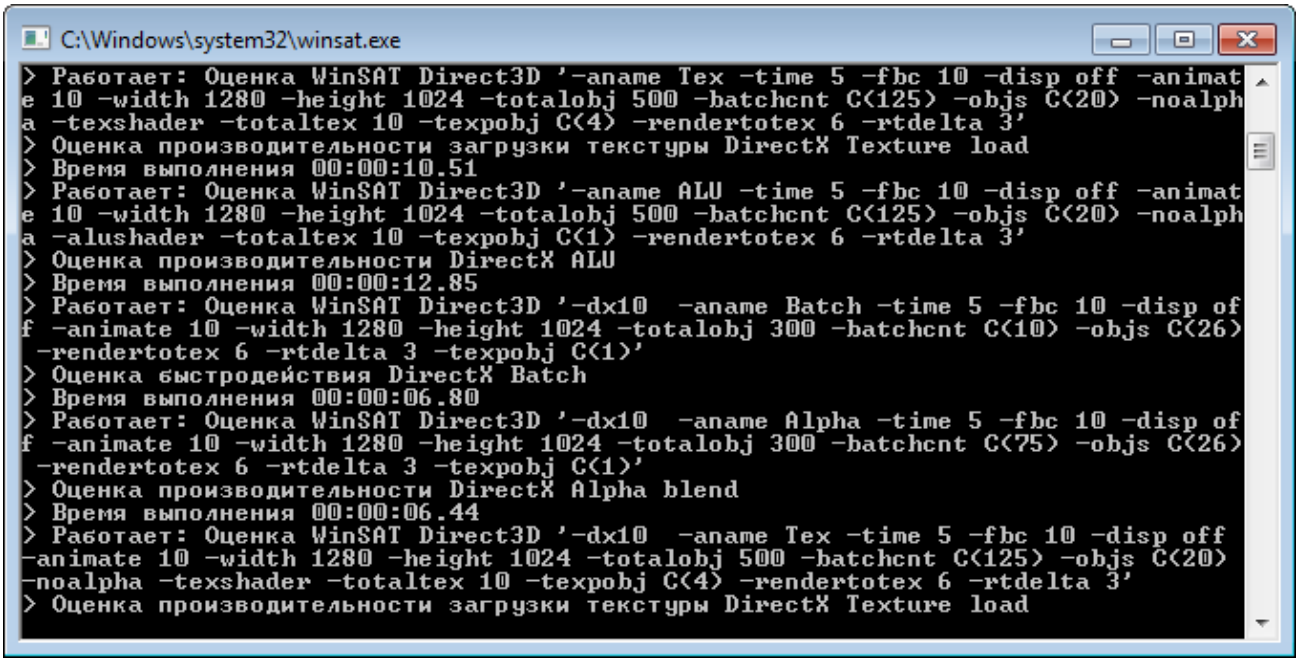

Рисунок 2.12 – Командне вікно

Після завершення тестування у головному вікно з’являється загальна оцінка продуктивності, а також оцінки кожного з компонентів (рисунок 2.13).

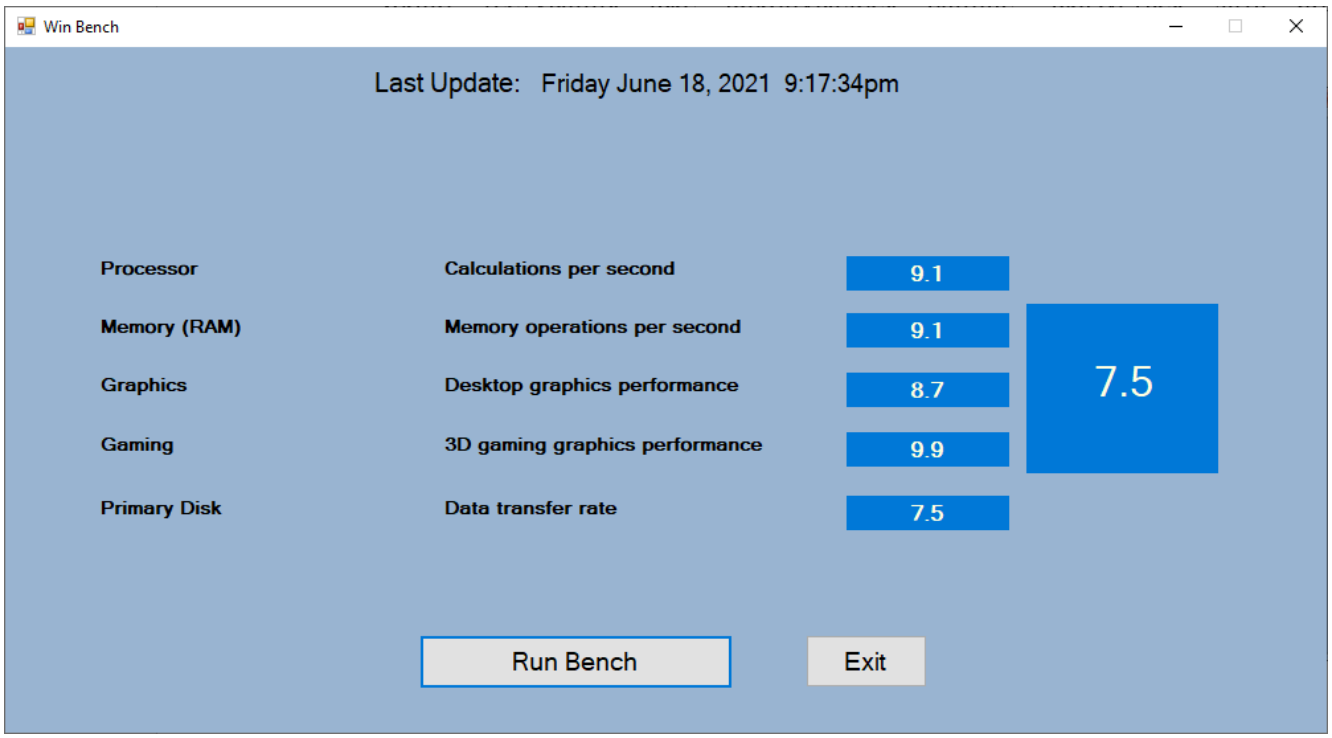

Рисунок 2.13 – Оцінка тестування

Якщо тестування вже виконувалося раніше, вказується дата його проведення (рисунок 2.14).

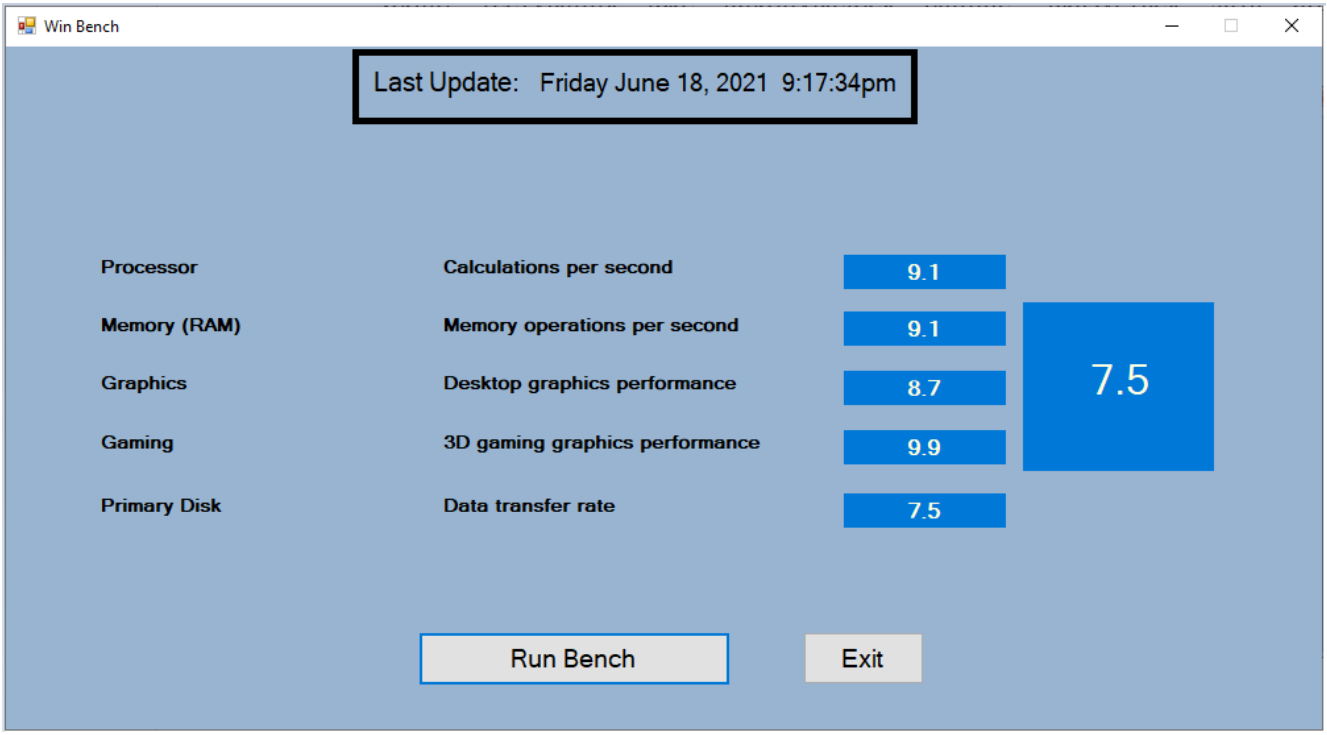

Рисунок 2.14 – Дата останнього тестування

Для того, щоб вийти із програми можна натиснути на червоний хрестик або вибравши кнопку «Exit» (рисунок 2.15).

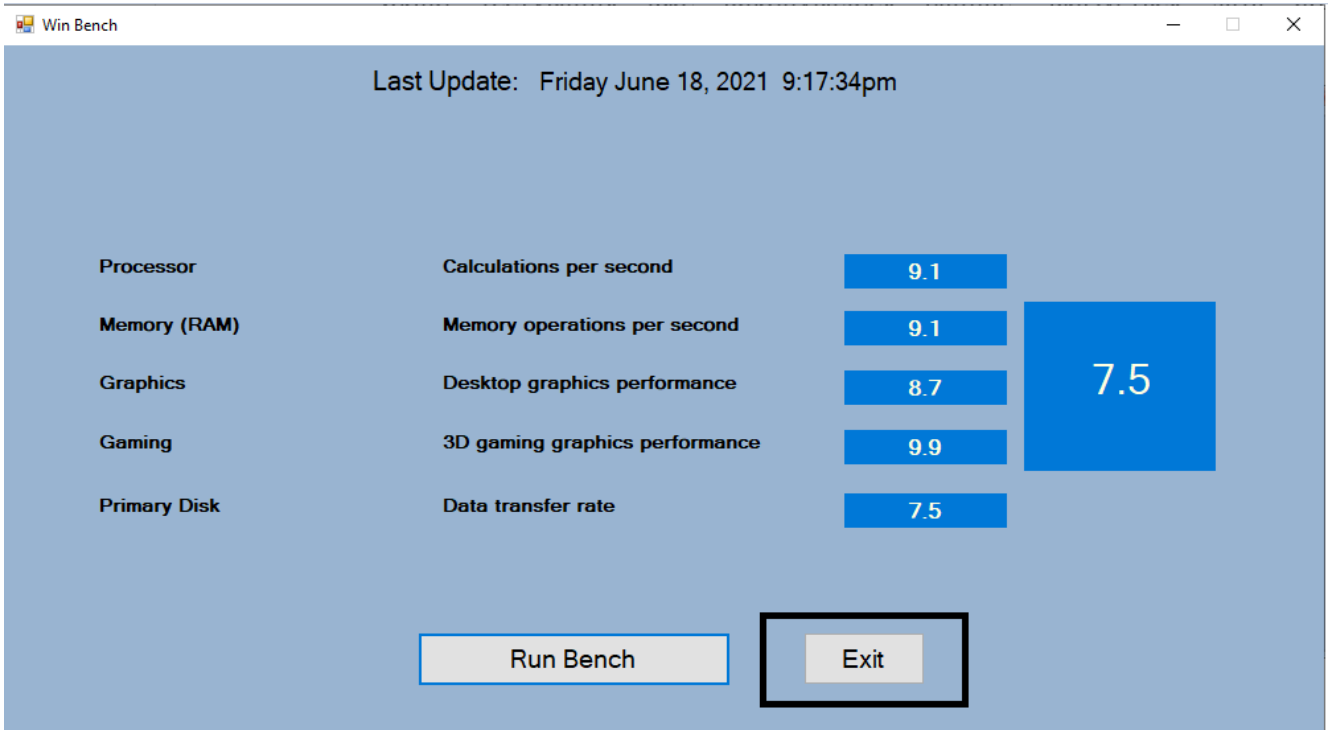

Рисунок 2.15 – Завершення роботи

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 оцінки продуктивності апаратного забезпечення робочої станції. 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 функцій кодування.

4. Коректність функціонування декодування.

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
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 тестування	Тестування виконане	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

З розвитком науково-технічного прогресу важливу роль грає можливість безпечного виконання людьми своїх трудових обов'язків. У зв'язку з цим була створена і розвивається наука про охорону праці і життєдіяльності людини.

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.

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.

Даний розділ дипломного проекту присвячений розгляду наступних питань:

1) визначення оптимальних умов праці техніка-програміста;
2) розрахунок освітленості;
3) розрахунок 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

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.

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.

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.

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 (табл. 3.1).

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.

Таблиця 3.1 - Параметри мікроклімату для приміщень, де встановлені комп'ютери

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

Таблиця 3.2 - Норми подачі свіжого повітря в приміщення, де розташовані 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 20 ... 40м ^{3} на особу Більш 40м ^{3} на особу	Не менше 30 Не менше 20 Природна вентиляція

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).

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. Тривала дія інтенсивного шуму [вище 80 дБ (А)] на слух людини приводить до його часткової або повної втрати.

У табл. 3.3 вказані граничні рівні звуку залежно від категорії тяжкості і напруженості праці, що є безпечними відносно збереження здоров'я і працездатності.

Таблиця 3.3 - Граничні рівні звуку, дБ, на робочих місцях.

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	-	-

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.

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.

Оптимальне розміщення предметів праці і документації в зонах досяжності:

1) дисплей розміщається в зоні а (у центрі);
2) системний блок розміщається в передбаченій ніші столу;
3) клавіатура – у зоні г/д;
4) «миша» – в зоні в справа;
5) сканер в зоні а/б (зліва);
6) принтер знаходиться в зоні а (праворуч);
7) документація: необхідна при роботі – в зоні легкої досяжності долоні – в, а у висувних ящиках столу – література, невикористовувана постійно.

Для комфортної роботи стіл повинен задовольняти наступним умовам:

1) висота столу повинна бути вибрана з урахуванням можливості сидіти вільно, в зручній позі, при необхідності спираючись на підлокітники;

2) нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, не був змушений підбирати ноги;

3) поверхня столу повинна мати властивості, що виключають появу відблисків у поле зору програміста;

4) конструкція столу повинна передбачати наявність висувних ящиків (не менше 3 для зберігання документації, лістингів, канцелярських приналежностей);

5) висота робочої поверхні рекомендується в межах 680-760мм. Висота поверхні, на яку встановлюється клавіатура, повинна бути близько 650мм.

Велике значення надається характеристикам робочого крісла. Так, рекомендована висота сидіння над рівнем підлоги перебуває в межах 420-550мм. Поверхня сидіння м'яка, передній край закруглений, а кут нахилу спинки - регульований.

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

1) голова не повинна бути нахилена більш ніж на 20^0;
2) плечі повинні бути розслаблені;
3) лікті - під кутом 80 ◦ ... 100^0;
4) передпліччя і кисті рук - в горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними чинниками: немає хорошої підставки для документів, клавіатура знаходиться дуже високо, а документи - низько, нікуди покласти руки і кисті, недостатній простір для ніг.

3.3 Розрахунок освітленості і рівня шуму

Розрахунок освітленості робочого місця зводиться до вибору системи освітлення, визначенню необхідного числа світильників, їхнього типу і розміщення. Виходячи з цього, розрахуємо параметри штучного освітлення.

Зазвичай штучне освітлення виконується за допомогою електричних джерел світла двох видів: ламп накалювання і люмінесцентних ламп. Використовуватимемо люмінесцентні лампи, які порівняно з лампами розжарювання мають ряд істотних переваг:

1) за спектральним складом світла вони близькі до денного, природного світла;

2) володіють більш високим ККД (у 1,5-2 рази вище, ніж ККД ламп розжарювання);

3) мають підвищену світловіддачею (в 3-4 рази вище, ніж у ламп розжарювання);

4) більш тривалий термін служби.

Розрахунок освітлення проводиться для кімнати площею 15м 2, ширина якої 5м, висота - 3 м. Для визначення кількості світильників визначється світловий потік, де

F - розраховується світловий потік, Лм;

E - нормована мінімальна освітленість, Лк (визначається за таблицею). Роботу програміста, відповідно до цієї таблиці, можна віднести до розряду точних робіт, отже, мінімальна освітленість E = 300лк;

S - площа освітлюваного приміщення (у нашому випадку S = 15м 2);

Z - відношення середньої освітленості до мінімальної (звичайно приймається рівним 1,1 ... 1,2, нехай Z = 1,1);

K - коефіцієнт запасу, враховує зменшення світлового потоку лампи в результаті забруднення світильників у процесі експлуатації (його значення залежить від типу приміщення й характеру проведених у ньому робіт і в нашому випадку K = 1,5);

n - коефіцієнт використання, (виражається відношенням світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп і обчислюється в частках одиниці; залежить від характеристик світильника, розмірів приміщення, фарбування стін і стелі, які характеризуються коефіцієнтами відображення від стін (Р С) і стелі (Р П)), значення коефіцієнтів Р С і Р П були зазначені вище: Р С = 40%, Р П = 60%. Значення n визначимо по таблиці коефіцієнтів використання різних світильників. Для цього обчислимо індекс приміщення:

S - площа приміщення, S = 15 м 2;

h - розрахункова висота підвісу, h = 2.92 м;

A - ширина приміщення, A = 3 м;

B - довжина приміщення, B = 5 м.

Знаючи індекс приміщення I, за таблицею 7 [23] знаходимо n = 0,22

Для освітлення вибираємо люмінесцентні лампи типу ЛБ40-1, світловий потік яких F = 4320 Лк.

Розрахуємо необхідну кількість ламп:

N - обумовлений число ламп;

F - світловий потік, F = 33750 Лм;

F л - світловий потік лампи, F л = 4320 Лм.

При виборі освітлювальних приладів використовуємо світильники типу ОД. Кожен світильник комплектується двома лампами.

Розрахунок рівня шуму.

Одним з несприятливих факторів виробничого середовища в ІОЦ є високий рівень шуму, створюваний друкованими пристроями, обладнанням 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 оператора.

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

L_i - рівень звукового тиску i-го джерела шуму;

n - кількість джерел шуму.

Отримані результати розрахунку порівнюється 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 обладнання і раціональна організація робочого місця оператора.

Рівні звукового тиску джерел шуму, що діють на оператора на його робочому місці представлені в табл. 3.4.

Таблиця 3.4 - Рівні звукового тиску різних джерел.

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

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

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

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

Отримане значення не перевищує допустимий рівень шуму для робочого місця оператора, 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 ще нижчою. Крім того при роботі принтера безпосередню присутність оператора необов'язково, тому що принтер обладнаний механізмом автоподачі аркушів.

3.4 Заходи та засоби протипожежного захисту

Під пожежною безпекою розуміють такий стан промислового або цивільного об’єкта, за якого з регламентованою ймовірністю виключається можливість виникнення і розвитку пожеж та впливу на людей небезпечних чинників пожежі, а також забезпечується захист матеріальних цінностей та довкілля.

Пожежна безпека об’єкта – доволі складне і багатоаспектне завдання, тому для його вирішення потрібно підходити комплексно. Комплекс заходів та засобів щодо пожежної безпеки складається із відповідних систем, зокрема:

1) системи запобігання пожеж, що містить підсистеми запобігання утворенню горючого середовища та виникненню в горючому середовищі джерела запалювання;

2) системи протипожежного захисту, що, своєю чергою, містить такі підсистеми: підсистема обмеження розвитку пожежі; підсистема забезпечення безпечної евакуації людей та майна; підсистема створення умов для успішного гасіння пожежі;

3) системи організаційно-технічних заходів, що передбачає організаційні, технічні, режимні та експлуатаційні заходи.

Організаційні заходи пожежної безпеки передбачають організацію пожежної охорони на об’єкті, проведення навчань з питань пожежної безпеки (інструктажі та пожежно-технічні мінімуми), застосування наочних засобів протипожежної пропаганди та агітації, організацію ДПД та ПТК, проведення перевірок, оглядів стану пожежної безпеки приміщень, будівель, об’єкта загалом та ін.

До технічних заходів належать суворе дотримання правил і норм, визначених чинними нормативними документами при реконструкції приміщень, будівель та об’єктів, технічному переоснащенні виробництва, експлуатації чи можливому переобладнанні електромереж, опалення, вентиляції, освітлення тощо.

Заходи режимного характеру передбачають заборону куріння та застосування відкритого вогню у недозволених місцях, недопущення появи сторонніх осіб у вибухонебезпечних приміщеннях чи об’єктах, регламентацію пожежної безпеки при проведенні вогневих робіт тощо.

Експлуатаційні заходи передбачають своєчасне проведення профілактичних оглядів, випробувань, ремонтів технологічного та допоміжного устаткування, а також інженерного господарства (електромереж, електроустановок, опалення, вентиляції).

Система запобігання пожежі – це комплекс заходів і технічних засобів, які запобігають виникненню пожежі.

Керівники підприємств повинні визначити обов'язки посадових осіб (у тому числі заступників керівника) з забезпечення пожежної безпеки, призначити відповідальних за пожежну безпеку окремих будівель, споруд, приміщень, дільниць, технологічного та інженерного обладнання, а також за зберігання та експлуатацію технічних засобів протипожежного захисту.

Обов'язки осіб, відповідальних за забезпечення пожежної безпеки, утримання та експлуатації засобів протипожежного захисту слід відобразити у відповідних документах.

Керівник підприємства зобов'язаний вживати (в межах наданих йому повноважень) відповідних заходів реагування на факти порушень чи невиконання іншими працівниками підприємства встановленого протипожежного режиму, вимог правил пожежної безпеки та нормативних актів, що діють у цій сфері.

Керівники підприємств повинні:

1) організувати розроблення комплексних заходів для забезпечення пожежної безпеки, впроваджувати на підприємстві досягнення науки і техніки, позитивний досвід;

2) відповідно до нормативних актів з пожежної безпеки розробляти і затверджувати положення, інструкції та інші нормативні акти,

що діють у межах підприємства, здійснювати постійний контроль за їх додержанням;

3) забезпечувати додержання протипожежних вимог стандартів, норм, правил, а також виконання вимог приписів і постанов органів держпожнагляду;

4) організовувати навчання працівників правилам пожежної безпеки та пропаганду заходів для їх забезпечення;

5) в разі відсутності в нормативних актах вимог, потрібних для гарантування пожежної безпеки - вживати відповідних заходів, узгоджуючи їх з органами держпожнагляду;

6) тримати у справному стані засоби протипожежного захисту і зв'язку, пожежну техніку, обладнання та інвентар, не допускати їх використання не за призначенням;

7) створювати в разі потреби відповідно до встановленого порядку підрозділи пожежної охорони та потрібну для їх функціонування матеріально-технічну базу;

8) подавати на вимогу Державної пожежної охорони відомості та документи про стан пожежної безпеки підприємства (об'єкта) і продукції, яку підприємство виробляє;

9) вживати заходів з впровадження автоматичних засобів виявлення і гасіння пожеж та використання з цією метою виробничої автоматики;

10) своєчасно інформувати пожежну охорону про несправність пожежної техніки, систем протипожежного захисту, водопостачання, а також завчасно інформувати про закриття доріг і проїздів на своїй території;

11) проводити службове розслідування випадків пожеж.

До первинних засобів пожежогасіння належать:

1) вогнегасники;

2) пожежні крани-комплекти, ручні насоси

3) лопати, ломи, сокири, гаки, пили, багри;

4) ящики з піском, бочки з водою;

5) азбестові полотнища, повстяні мати та ін.

Первинні засоби пожежогасіння розміщують на пожежних щитах, які встановлюють на території об'єкта з розрахунку один щит на 5000 $м^2$. Вони мають бути пофарбовані у червоний колір, а пожежний інструмент у чорний.

Серед первинних засобів пожежогасіння найважливішу роль відіграють вогнегасники різних типів: водяні, водо-пінні, порошкові, вуглекислотні, газові.

Залежно від способу транспортування вони бувають: переносні (до 20 кг) та пересувні (до 450 кг).

Залежно від об'єму вогнегасники бувають малолітражні (до 5л), ручні (до 10 л), пересувні (понад 10л).

Вогнегасники маркують буквами, що означає їх вид та цифрами, що визначають їх об'єм.

Найбільш перспективними є порошкові вогнегасники, які застосовують для гасіння лужних металів, ЛЗР і ТР, електрообладнання, що горить під напругою до 1000В, твердих та газоподібних речовин.

Найбільш розповсюдженими є:

1) ОП-1, ОП-2, ОП-9, ОП-10 — переносні;

2) ОПА-50, ОПА-100 — пересувні.

Вони відрізняються між собою лише складом порошку та пристроєм для його подачі.

Вуглекислотні вогнегасники застосовуються для гасіння загорянь на машинах, автомобілях і для невеликих об'ємів нафтопродуктів, а також електроустановок під напругою до 1000В.

У корпусі вогнегасника міститься вуглекислий газ у рідкому стані під високим тиском 6мПа (ручні) і 15 мПА (переносні). У горловині балону змонтований спеціальний пусковий пристрій із сифонною трубкою, який

приводиться у дію за допомогою вентильного або пістолетного пристрою. Виходячи з балону назовні, зріджений двооксид вуглецю перетворюється на снігоподібну масу за температури - 80°С.

Вибір типу вогнегасника визначається розмірами загоряння і можливих осередків пожеж.

Утворенню горючого середовища запобігають застосуванням герметичного виробничого устаткування, максимально можливою заміною в технологічних процесах горючих речовин та матеріалів негорючими, обмеженням кількості пожежо-, вибухонебезпечних речовин та матеріалів під час використання та зберігання, а також правильним їх розміщенням, ізоляцією горючого та вибухонебезпечного середовища, організацією контролю за складом повітря в приміщенні та контролю за станом середовища в апаратах, застосуванням робочої та аварійної вентиляції, відведенням горючого середовища в спеціальні пристрої та безпечні місця, застосуванням в установках з горючими речовинами пристроїв від пошкодження та аварій, використанням інгібувальних (хімічно активні компоненти, що сприяють припиненню пожеж) та флегматизаційних (інертні компоненти, що роблять середовище негорючим) речовин.

Виникненню в горючому середовищі джерела запалювання запобігають використанням устаткування та пристроїв, при роботі яких не виникає джерел запалювання, використанням електроустаткування, що відповідає за досягнення класу пожежо- та вибухонебезпеки приміщеннями та зонами груп і категорій вибухонебезпечної суміші, виконанням вимог щодо сумісного зберігання речовин та матеріалів, використанням устаткування, що задовольняє вимоги електростатичної іскробезпеки, улаштуванням блискавкозахисту, організацією автоматичного контролю параметрів, що визначають джерела запалювання, використанням швидкодіючих засобів захисного вимкнення, заземленням устаткування, видовжених металоконструкцій, використанням при роботі з ЛЗР

інструментів, що не допускають іскроутворення, ліквідацією умов для самоспалахування речовин і матеріалів, усуненням контакту з повітрям піроформих речовин, підтриманням температури нагрівання поверхні устаткування, пристроїв, речовин та матеріалів, які можуть контактувати з горючим середовищем нижче гранично допустимої (80 %) температури займання.

У даному розділі дипломної роботи були викладені вимоги до робочого місця техніка-програміста. Створені умови повинні забезпечувати комфортну роботу. На підставі вивченої літератури з даної проблеми, були зазначені оптимальні розміри робочого столу і крісла, робочої поверхні, а також проведено вибір системи і розрахунок оптимального освітлення виробничого приміщення, а також розрахунок рівня шуму на робочому місці. Дотримання умов, що визначають оптимальну організацію робочого місця інженера - програміста, дозволить зберегти гарну працездатність протягом усього робочого дня, підвищить як в кількісному, так і в якісному відносинах продуктивність праці програміста, що в свою чергу сприятиме якнайшвидшій розробці і налагодженню програмного продукту.

Висновки

Метою створення дипломного проекту є розробка програми тестування продуктивності «Win Bench». Вікно програми має основні компоненти, тестування яких виконується в програмі, а саме: процесор, пам’ять, графіка, ігрові можливості, основний розділ. В процесі тестування програмний засіб відкриває командне вікно. Також в пам’яті засобу зберігається дата та результати останнього тестування.

Мовою програмування, яка використовується для створення програмного засобу, є C#. Програмний код реалізований у середовищі розробки Microsoft Visual Studio.

При виконанні дипломного проекту було виконано наступне:

1) створено діаграми класів, послідовності, станів;
2) створено загальну блок-схему роботи програмного засобу;
3) проведено порівняння аналогів з даним програмним засобом;
4) розроблено інтерфейс редактора;
5) розроблено структурний елемент інтерфейсу;
6) розроблено функціональність редактора;
7) створено зрозумілу інструкцію користувача;
8) розроблено та реалізовано логіку програми.

У разі впровадження, подальшого використання та попиту можливо удосконалювати програмний засіб..

Програмний засіб матиме попит серед аналогічних завдяки багатьом перевагам, таким як компактність, наявність основних функцій оцінювання продуктивності, швидкодія. Завдяки чому програмний засіб є актуальним в наш час.

Перелік використаних джерел

1. Bloom C. Solving the problems of context modeling // California Institute of Technology, 1996.
2. Schmidhuber J. Sequential neural text compression // IEEE Transactions on Neural Networks. 1996. V. 7 (1)., pp. 142 - 146.
3. Гогіташвілі Г.Г., Лапін В.М. Основи охорони праці: Навч. посіб. – 4-е вид., випр.. і доп. – К.: Знання, 2008. - 302с.
4. Гігієна праці та виробнича санітарія: (Навч.-метод. посібник) / І.М. Трахтенберг, М.М. Коршун, О.В. Чебанова; За ред. І.М. Трахтенберга. - К.; 1997. - 464 с.
5. Безпека праці: ергономічні та естетичні основи: Навчальний посібник / С. Апостолюк, В.С. Джигирей. А.В . Апостолюк та ін. - К.: Знання, 2006. -215 с.
6. Желібо Є П., Заверуха Н. М., Зацарний В, В. Безпека життедіяльності / За ред. Є П. Желібо. - К.: Каравела, 2010. - 328 с.
7. Апостолюк С., Джигирей В.С., Апостолюк А.В. та ін. / Безпека праці: ергономічні та естетичні основи: Навчальний посібник - К.: Знання, 2006. – 215 с.
8. Траїтенберг І.М., Коршун М.М., Чебанова О.В. / Гігієна праці та виробнича санітарія: (Навч.-метод. посібник); За ред. І.М. Трахтенберга. - К.; 1997. - 464 с
9. Гандзюк М.П., Желібо Є.П., Халімовський М.О. Основи Охорони праці: Підруч. для студ. вищих навч. закл. За ред. Гандзюка М.П. - К.: Каравела; Львів: Новий Світ-2000, 2003. - 408 с.
10. Годин, В.В. Информационное обеспечение управленческой деятельности: підручник / Годин В.В., Гордеев И.К. – М.: Мастерство: Высшая школа, 2001.
11. Управління проектами : навч. посіб. / О. В. Ульянченко [та ін.] ; за ред. д-ра екон. наук, проф. О. В. Ульянченка та канд. екон. наук П. Ф. Цигікала ; Харк. нац. аграр. ун-т ім. В. В. Докучаєва. - Х. : ХНАУ ім. В. В. Докучаєва, 2010. - 522 с.

12. Соммервил И. Инженерия программного обеспечения / И.Соммервилл. – М.: Вильямс, 2002. – 540с.

13. Гущина, И.Э. Управленческий учет: основы теории и практики: уч. посібник/ И.Э. Гущина, Н.М. Балакирева. – М.: КНОРУС, 2004.

14. Титоренко Г. А. / Інформаційні технології управління: Навчальний посібник для вузів – М.: ЮНИТИ, 2007.

15. С++. Основи програмування. Теорія та практика: підручник / [О.Г. Трофименко, Ю.В. Прокоп, І.Г. Швайко, Л.М. Буката та ін.]; за ред. О.Г. Трофименко. – Одеса: Фенікс, 2010. – 544 с.

16. Інтегроване середовище розробки Visual Studio: [Електронний ресурс] –Режим доступу: https://msdn.microsoft.com/ruru/library/dn762121.aspx

17. Мова програмування C# Огляд мови C#: [Електронний ресурс] – Режим доступу:https://ci-sharp.ru/ukr/Teaching/mova-programmirovaniya-s-obzor-yazika-s.html

18. Нові можливості .NET Framework: [Електронний ресурс] – Режим доступу: https://msdn.microsoft.com/ruru/library/ms171868%28v=vs.110%29.aspx

19. Узагальнення досвіду використання АРМ: [Електронний ресурс] – Режим доступу: http://biogr.znate.ru/docs/index-1467.html

20. Степова С. В., Когут А. Л. Доцільність застосування інформаційних технологій в ресторанному бізнесі. [Електронний ресурс]. – Режим доступу: http://www.rusnauka.com/3_ANR_2014/ Informatica/3_153623.doc.htm.

21. Узагальнення досвіду використання АРМ: [Електронний ресурс] – Режим доступу: http://biogr.znate.ru/docs/index-1467.html

22. WPF – Windows Presentation Foundation: [Електронний ресурс] – Режим доступу: http://professorweb.ru/my/WPF/base_WPF/level1/info_WPF.php

Додаток А – Код модуля MainWindow

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Xml;
using System.IO;
using System.Reflection;
using System.Diagnostics;
using System.Threading;
using System.Runtime.InteropServices;
using hcwgenericclasses;
using editform;


namespace winbench
{
   public partial class Form1 : Form
   {
      StringBuilder sb = new StringBuilder();
      List<XMLItem> Xitem = new List<XMLItem>();
      List<string> WsFiles = new List<string>();
      List<string> LockedFiles = new List<string>();
      List<string> AssessmentFileNames = new List<string>();
      ListInfoStack HeaderStack = new ListInfoStack();
      private BackgroundWorker bw = new BackgroundWorker();
      string CmdLine = "formal";
      const int PAD = 6;
      bool CancelProcessDir = false;
      bool EnabletscbSelectedIndexChangedHandler = true;
      private Int32 ProcessID;
      private IntPtr ProcessHandle;
      private Color SavedBackColor;
      private Color SavedTextColor;

      private static string DataStorePath = "c:\\Windows\\Performance\\WinSAT\\DataStore";
      private static string WinSATFileName = GlobalRes.ExePathFor64BitApplication;

      private static bool is64BitProcess = (IntPtr.Size == 8);
      private static bool is64BitOperatingSystem = is64BitProcess || InternalCheckIsWow64();


      DragDropTools DDT = new DragDropTools();
      ExceptionHandlerTools EHT = new ExceptionHandlerTools();
      DialogTools DT = new DialogTools();
      FileTools FT = new FileTools();
      AboutBoxTools ABT = new AboutBoxTools();

      public Form1()
      {
         InitializeComponent();
      }
```

```
[DllImport("kernel32.dll", SetLastError = true, CallingConvention = CallingConvention.Winapi)]
[return: MarshalAs(UnmanagedType.Bool)]
private static extern bool IsWow64Process(
    [In] IntPtr hProcess,
    [Out] out bool wow64Process
 );
private static bool InternalCheckIsWow64()
{
   if ((Environment.OSVersion.Version.Major == 5 && Environment.OSVersion.Version.Minor >= 1) ||
      Environment.OSVersion.Version.Major >= 6)
   {
      using (Process p = Process.GetCurrentProcess())
      {
         bool retVal;
         if (!IsWow64Process(p.Handle, out retVal))
         {
            return false;
         }
         return retVal;
      }
   }
   else
   {
      return false;
   }
}

public class Win64Interop
{
   [DllImport("Kernel32.Dll", EntryPoint = "Wow64EnableWow64FsRedirection")]
   public static extern bool EnableWow64FSRedirection(bool enable);
}



private void Form1_Load(object sender, EventArgs e)
{
   PopulateDataFileList();
   PopulateAssessmentFileNames();
   if (AssessmentFileNames.Count > 0)
   {
      ParseXml(AssessmentFileNames[AssessmentFileNames.Count - 1]);
   }


   bw.DoWork += new DoWorkEventHandler(bwDoWork);
   bw.RunWorkerCompleted += new RunWorkerCompletedEventHandler(bwCompleted);
   SavedBackColor = this.lbDisk.BackColor;
   SavedTextColor = this.lbDisk.ForeColor;

}

private void ParseXml(string filename)
{


   GetXmlFromFile(filename);
   string systemscore = GetDataValue("WinSPR", "SystemScore");
   lblLastUpdate.Text = GetAttributeValue("SystemEnvironment", "ExecDateTOD", 1);
   lbDisk.Text = GetDataValue("WinSPR", "DiskScore");
```

```
    if (lbDisk.Text == systemscore)
    {
        lbDisk.BackColor = tbBaseScore.BackColor;
        lbDisk.ForeColor = tbBaseScore.ForeColor;
    }
    else
    {
        lbDisk.BackColor = SavedBackColor;
        lbDisk.ForeColor = SavedTextColor;
    }
    lbGaming.Text = GetDataValue("WinSPR", "GamingScore");
    if (lbGaming.Text == systemscore)
    {
        lbGaming.BackColor = tbBaseScore.BackColor;
        lbGaming.ForeColor = tbBaseScore.ForeColor;
    }
    else
    {
        lbGaming.BackColor = SavedBackColor;
        lbGaming.ForeColor = SavedTextColor;
    }
    lbGraphics.Text = GetDataValue("WinSPR", "GraphicsScore");
    if (lbGraphics.Text == systemscore)
    {
        lbGraphics.BackColor = tbBaseScore.BackColor;
        lbGraphics.ForeColor = tbBaseScore.ForeColor;
    }
    else
    {
        lbGraphics.BackColor = SavedBackColor;
        lbGraphics.ForeColor = SavedTextColor;
    }
    lbRam.Text = GetDataValue("WinSPR", "MemoryScore");
    if (lbRam.Text == systemscore)
    {
        lbRam.BackColor = tbBaseScore.BackColor;
        lbRam.ForeColor = tbBaseScore.ForeColor;
    }
    else
    {
        lbRam.BackColor = SavedBackColor;
        lbRam.ForeColor = SavedTextColor;
    }
    lbProcessor.Text = GetDataValue("WinSPR", "CpuScore");
    if (lbProcessor.Text == systemscore)
    {
        lbProcessor.BackColor = tbBaseScore.BackColor;
        lbProcessor.ForeColor = tbBaseScore.ForeColor;
    }
    else
    {
        lbProcessor.BackColor = SavedBackColor;
        lbProcessor.ForeColor = SavedTextColor;
    }
    tbBaseScore.Text = GetDataValue("WinSPR", "SystemScore");
    this.btnRunBench.Select();

}
private void ExitBtn(object sender, EventArgs e)
{
    this.Close();
```

```
}
private void openToolStripMenuItem_Click(object sender, EventArgs e)
{

}

private void GetXmlFromFile(string filename)
{
   XmlReaderSettings settings = new XmlReaderSettings();
   settings.DtdProcessing = DtdProcessing.Parse;
   XmlReader reader = XmlReader.Create(filename, settings);
   Xitem.Clear();
   int count = 0;
   int x = 0;
   bool AddToXi = false;

   while (reader.Read())
   {
      XMLItem dataitem = new XMLItem();
      AddToXi = false;
      switch (reader.NodeType)
      {
         case XmlNodeType.Element:
            dataitem.name = reader.Name.ToString();
            if (!HeaderStack.IsEmpty())
            {
               dataitem.header = HeaderStack.GetLastItem();
               HeaderStack.Push(dataitem.name);
            }
            else
            {
               dataitem.header = "";
               HeaderStack.Push(dataitem.name);
            }

            dataitem.data = reader.Value;
            AddToXi = true;
            break;
         case XmlNodeType.Text:
            Xitem[Xitem.Count - 1].data = reader.Value.ToString();
            break;
         case XmlNodeType.CDATA:
            Xitem[Xitem.Count - 1].data = reader.Value.ToString();
            break;
         case XmlNodeType.ProcessingInstruction:
            dataitem.name = reader.Name;
            dataitem.data = reader.Value;
            dataitem.header = "PROCESSING INSTRUCTION";
            AddToXi = true;
            break;
         case XmlNodeType.Comment:
            dataitem.name = reader.Name;
            dataitem.data = reader.Value;
            dataitem.header = "COMMENT";
            AddToXi = true;
            break;
         case XmlNodeType.XmlDeclaration:
            dataitem.header = "XML DECLARATION";
            dataitem.name = reader.Name;
            dataitem.data = reader.Value;
            AddToXi = true;
```

```
                break;
            case XmlNodeType.Document:
                break;
            case XmlNodeType.DocumentType:
                dataitem.name = "<!DOCTYPE " + reader.Name + " " + reader.Value;
                dataitem.header = "";
                dataitem.data = reader.Value.ToString();
                AddToXi = true;
                break;
            case XmlNodeType.EntityReference:
                dataitem.name = "ENTITY REFERENCE";
                dataitem.data = reader.Name.ToString();
                dataitem.header = "";
                AddToXi = true;
                break;
            case XmlNodeType.EndElement:
                HeaderStack.Pop();
                break;
        }
        if (reader.HasAttributes)
        {
            for (x = 0; x < reader.AttributeCount; x++)
            {
                reader.MoveToAttribute(x);
                XMLAttribute xa = new XMLAttribute();
                xa.aname = reader.Name;
                xa.avalue = reader.Value;
                dataitem.attributes.Add(xa);
            }
        }
        if (AddToXi)
        {
            Xitem.Add(dataitem);
            count++;
        }

    }
}

private bool ItemContainsData(int itemnumber)
{
    if (itemnumber < 0 || itemnumber > Xitem.Count - 1)
    {
        return false;
    }
    if (Xitem[itemnumber].data.Length > 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}
private string FormatXmlForDisplay()
{

    sb.Clear();
    sb.Append("File Name: " + tscbAssessmentFiles.SelectedItem.ToString() + "\r\n\n");
    int itemcount = 1;
```

```
                int attributecount = 1;
                foreach (XMLItem i in Xitem)
                {

                    sb.Append("Item     : " + itemcount.ToString() + "\r\n");
                    sb.Append("Header   : " + i.header.PadRight(PAD) + "\r\n");
                    sb.Append("Name     : " + i.name.PadRight(PAD) + "\r\n");
                    attributecount = 1;
                    foreach (XMLAttribute s in i.attributes)
                    {
                        sb.Append("Attr (" + attributecount.ToString() + ") : " + s.aname.PadRight(PAD) + " " +
s.avalue.PadRight(PAD) + "\r\n");
                        attributecount++;
                    }
                    sb.Append("Data     : " + i.data.PadRight(PAD) + "\r\n\n");
                    itemcount++;
                }
                return sb.ToString();

            }

            private string GetAttributeValue(string header, string name, int number)
            {
                string result = "";
                int x = 0;
                for (x = 0; x < Xitem.Count; x++)
                {
                    if (Xitem[x].header == header && Xitem[x].name == name)
                    {
                        if (Xitem[x].attributes.Count >= number)
                        {
                            result = Xitem[x].attributes[number - 1].avalue;
                            break;
                        }

                    }
                }
                return result;
            }
            private string GetDataValue(string header, string name)
            {
                string result = "";
                int x = 0;
                for (x = 0; x < Xitem.Count; x++)
                {
                    if (Xitem[x].header == header && Xitem[x].name == name)
                    {
                        result = Xitem[x].data;
                    }
                }
                return result;
            }

            private void PopulateDataFileList()
            {
                if (!Directory.Exists(DataStorePath))
                {
                    DT.NotifyDialog(this, "WEXEP.EXE \r\n No benchmark files found. \r\n Click the RunBenchmark
Button.", 4000);
                    btnRunBench.Select();
                    return;
```

```
            }
            WsFiles.Clear();
            LockedFiles.Clear();
            WsFiles.Add(DataStorePath);
            FT.ProcessDirectories(ref WsFiles, ref LockedFiles, false, ref CancelProcessDir);

        }

        private void PopulateAssessmentFileNames()
        {
            AssessmentFileNames.Clear();
            foreach (string s in WsFiles)
            {
                if (s.Contains("Formal.Assessment"))
                {
                    AssessmentFileNames.Add(s);
                }
            }
            AssessmentFileNames.Sort();
            tscbAssessmentFiles.Items.Clear();
            foreach (string s in AssessmentFileNames)
            {
                tscbAssessmentFiles.Items.Add(s);
            }
            EnabletscbSelectedIndexChangedHandler = false;
            tscbAssessmentFiles.SelectedIndex = tscbAssessmentFiles.Items.Count - 1;
            EnabletscbSelectedIndexChangedHandler = true;

        }



        private void btnGetBenchMark_Click(object sender, EventArgs e)
        {

            bool result = true;
            Process n = new Process();
            Int32 ID = 0;

            n.StartInfo.FileName = WinSATFileName;

            n.StartInfo.Arguments = CmdLine;
            n.StartInfo.WindowStyle = ProcessWindowStyle.Minimized;
            Exception StartEx = new Exception("(Application) - Unable to Start WinSAT.exe, althought path is
valid.");
            try
            {
                if (is64BitOperatingSystem)
                {
                    Win64Interop.EnableWow64FSRedirection(false);
                    n.Start();
                    ID = n.Id;
                    Win64Interop.EnableWow64FSRedirection(true);
                }
                else
                {
                    n.Start();
                    ID = n.Id;
                    ProcessHandle = n.Handle;
                }
```

```
    }
    catch (Exception ex)
    {
        EHT.GeneralExceptionHandler("Error Rerunning Benchmark", n.StartInfo.FileName, false, ex);
        result = false;
    }
    finally
    {
        if (result)
        {
            DT.NotifyDialog(this, "Starting Benchmark...");
            lblLastUpdate.Text = "";
            lblLastUpdate.Text = "Update is Running";
            ProcessID = n.Id;
            this.btnClose.Select();
        }
    }

    btnRunBench.Enabled = false;
    bw.RunWorkerAsync();
    return;
}
private void bwDoWork(object sender, DoWorkEventArgs e)
{
    BackgroundWorker worker = sender as BackgroundWorker;
    FileWatcher FSW = new FileWatcher(DataStorePath, "*.*");
    FSW.Start();
    while (!FSW.Completed)
    {

    }

}
private void bwCompleted(object sender, RunWorkerCompletedEventArgs e)
{
    Thread.Sleep(3000);
    PopulateDataFileList();
    PopulateAssessmentFileNames();
    if (AssessmentFileNames.Count > 0)
    {
        ParseXml(AssessmentFileNames[AssessmentFileNames.Count - 1]);
    }
    btnRunBench.Enabled = true;
}
private void tscbAssessmentFiles_SelectedIndexChanged(object sender, EventArgs e)
{
    if (!EnabletscbSelectedIndexChangedHandler)
    {
        return;
    }
    int x = tscbAssessmentFiles.SelectedIndex;
    string s = tscbAssessmentFiles.SelectedItem.ToString();
    ParseXml(s);
}
private void reloadAssessmentsToolStripMenuItem_Click(object sender, EventArgs e)
{
    DT.NotifyDialog(this, "Rescanning DataStore Folder");
    PopulateDataFileList();
    PopulateAssessmentFileNames();
    if (AssessmentFileNames.Count > 0)
    {
```

```
                ParseXml(AssessmentFileNames[AssessmentFileNames.Count - 1]);
            }
        }

        private void lblLastUpdate_Click(object sender, EventArgs e)
        {

        }

        private void panelBSHighlight_Paint(object sender, PaintEventArgs e)
        {

        }

        private void panelRatingDes_Paint(object sender, PaintEventArgs e)
        {

        }
    }
}
```