Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

БАКАЛАВРСЬКА ДИПЛОМНА РОБОТА

на тему:

Клавіатурний тренажер для вивчення слів англійської мови

ПОЯСНЮВАЛЬНА ЗАПИСКА

Виконав студент 2 курсу, групи 1КІ-20мсз

спеціальності 123 Комп'ютерна інженерія

Крупський А. О.

Керівник к.т.н., доц. каф. ОТ

Снігур А. В.

" " 2022 р.

Рецензент д.т.н., проф.,зав. каф. ЗІ

Лужецький В. А.

" 16 " 06 2022 р.

Допущено до захисту

д.т.н., проф. Азаров О.Д.

" 17 " 06 2022 р.

ВНТУ 2022

ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітній рівень — бакалавр

Спеціальність — 123 Комп'ютерна інженерія

ЗАТВЕРДЖУЮ

Завідувач кафедри обчислювальної техніки

О.Д. Азаров

"08" 02 2022 р.

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ

студенту **Крупського Артура Олексійовича**

1 Тема роботи «Клавіатурний тренажер для вивчення слів англійської мови» керівник роботи Снігур А.В. к.т.н., доцент, затверджено наказом вищого навчального закладу від 24 березня 2022 року № 66

2 Строк подання студентом роботи 17.06.2022р.

3 Вихідні дані до роботи: технічний опис програмного застосунку, мова програмування C#, віконний додаток, середовище розробки Microsoft Visual Studio.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): вступ, аналіз сучасних методів розробки,варіантний вибір засобів

для розробки програмного забезпечення, програмна реалізація, тестування розробленого програмного забезпечення.

5 Графічний матеріал — діаграма класів.

6 Консультанти розділів роботи приведені в таблиці 1.

Таблиця 1— Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Снігур А.В. к.т.н., доцент	03.03.22	04.03.22

7 Дата видачі завдання 09.02.22 .

8 Календарний план виконання МКР приведений в таблиці 2.

Таблиця 2 — Календарний план

№ з/п	Назва етапів МКР	Строк виконання	Підпис
1	Постановка задачі	10.03.22	вик.
2	Пошук матеріалів по технологіям розробки Android IoT застосунків	10.03.22— 25.03.22	вик.
3	Структурне проектування установки	26.03.22— 28.03.22	вик.
4	Обгрунтування та вибір засобів реалізації системи	29.03.22— 04.04.22	вик.
5	Розробка Android застосунку	05.04.22— 29.04.22	вик.
6	Підготовка матеріалів тестування та розробка інструкції користувача	30.04.22— 27.05.22	вик.
7	Оформлення пояснювальної записки та ілюстративного матеріалу	28.05.22— 06.06.22	вик.
8	Перевірка якості виконання бакалаврської роботи та усунення недоліків	07.06.22	вик.

Студент Крупський А. О.

Керівник к.т.н., доц. Снігур А.В.

АНОТАЦІЯ

Роботу виконав Крупський Артур Олексійович

Пояснювальна записка містить 90 сторінок, 2 таблиці, 18 рисунків, 5 лістингів та 8 посилань.

Дану бакалаврську дипломну роботу присвячено створенню додатку для вивчення англійської мови

В роботі був виконаний зрозумілий інтерфейс та швидкий переклад слів для засвоєння інформації.

В результаті виконання роботи було зроблено програмне забезпечення для перекладу та вивчення слів англійської мови.

Перевірка працездатності додатку протестована шляхом практичного тестування.

Ключові слова: клавіатурний тренажер, вивчення іноземних мов, англійська мова

ABSTRACT

This Bachelor's thesis is devoted to creating an application for Learning English

The work has a clear interface and fast translation of words for assimilation of information.

As a result of the work, software was created for translating and learning English words.

The app's health check is tested by practical testing.

Keywords: keyboard simulator, learning foreign languages, English

ЗМІСТ

ВСТУП

У міру того як розширюється інформатизація сучасного суспільства при переході до суспільства майбутнього, зростає значення прикладної (обчислювальної, комп'ютерної, інженерної) лінгвістики, науки, що знаходиться на стику глибоко людської, гуманітарної науки лінгвістики (мовознавства), що вивчає закони розвитку та користування могутнім засобом мовою, — і комп'ютерного знання, за допомогою якого машині передасться все більша частина інтелектуальної праці людини.

Переклад — процес та результат створення на основі вихідного тексту однією мовою рівноцінного йому відношення тексту іншою мовою.

При цьому комунікативна рівноцінність розуміється, як якість тексту перекладу, що дозволить йому виступити:

— у процес спілкування носіїв різноманітних мов;

— як повна заміна вихідного тексту (оригіналу);

— у сферах дій мов перекладів.

Інтерфейс користувача представляє собою віконний додаток розроблений за допомогою модульної платформи з відкритим доступом в середовищі Visual Studio використовуючи об'єктно–орієнтовану мову програмування C++. Можливості розробки на ООП C++ передусім використовуються для організації роботи самої програми, збереження інформації для подальшого використання та універсальність доступу на різних системах [1].

Спочатку розроблюється дизайн інтерфейсу, як будуть виглядати вікна, інтерфейс, повідомлення. Вони повинні враховувати ціль додатку та бути зручними та зрозумілими. Потрібні вікна для головного інтерфейсу з розпорядком, подіями, їх описом, часом. Для використання програми користувачеві потрібну ввімкнути додаток та почати створювати події вказуючи їх час та день проведення, для того щоб програма зручно вивела розпорядок і могла повідомити користувача про наближення події.

Розробка, яка полягає у необхідності вирішення можливості організації та оптимізації часу, які дозволять отримати додаток, на основі якого буде можливість коректно організовувати та оптимізовувати час та робочий процес є **актуальною задачею.**

Об’єкт дослідження — процес створення програмного забезпечення для тестування знань користувачів на знання англійської мови.

Предмет дослідження — методи та засоби для розробки клавіатурного тринажеру.

Метою роботи є розробка програмного забезпечення для тестування знань користувачів на знання англійської мови використовуючи середовище Visual Studio 2019 та мову програмування C++.

Задачі дослідження бакалаврської роботи:

— проаналізувати методи та технології розробки додатків ООП;

— проаналізувати існуючі аналоги;

— проаналізувати та обрати засоби реалізації системи;

— виконати моделювання;

— виконати розробку віконного додатку за допомогою мови програмування C++;

— створити систему що буде аналізувати введені данні та повідомляти користувача про недоліки;

— протестувати розроблений додаток.

Методи дослідження бакалаврської роботи є використання принципів об’єктно-орієнтованого програмування мовою C++, NET Framework для реалізації поставленної мети.

Апробація результатів бакалаврської роботи: зроблено доповідь на L науково–технічній конференції підрозділів ВНТУ.

Практичне значення отриманих результатів полягає в можливості використовування розробленої системи для тестування знань користувачів на знання англійської мови

1 АНАЛІТИЧНИЙ ОГЛЯД ІСНУЮЧИХ ПІДХОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ-ТРЕНАЖЕРІВ ДЛЯ РОЗРОБЛЕНОГО ПРОДУКТУ

Головною ідеєю роботи є створення додатку для тренажеру для вивчення англійської мови.

1.1 Загальні методи побудови та функціонування клавіатурних тренажерів

Дуже багато людей сьогодні хочуть вивчити англійську мову швидко, системно, якісно. Особливо це актуально для тих, хто хоче виїхати для життя або роботи в англомовну країну, або для тих, хто пов'язує своє життя з англійською мовою, набуваючи професії перекладача або викладача англійської. Для кожного буде зручно у вивченні мови використовувати тренажер з англійської мови.
Основою програми тренажеру є те, що людина одночасно може тренувати і читання, і слухання, і письмо. Розвивати усі ці навички. Вправ, вкладених у це - безліч. Наприклад, диктант. Коли диктор читає текст, учень повинен написати його, не бачачи перекладу, а лише слухаючи текст. Також можна спростити завдання, відкривши вікно з перекладом тексту. Вправа перекладом передбачає те, що учень бачить текст, який потрібно перекласти на англійську мову. Учень перекладає, вибираючи потрібні слова з пропонованих карток та вибудовуючи їх у правильній послідовності [1].

Широкої популярності досягли онлайн тренажери [2]. Онлайн тренажер зручний тим, що користувач може використати його у будь-який час, уроки не надто довгі. При цьому користь від уроків – колосальна. Вчитися за допомогою тренажера приємно та не не сильно виснажує. В основі таких програм лежить клієн-серверна архітектура.

Більшість онлайн додатків дозволяють користувачам спілкуватися за допомогою інтернету. У цих умовах для забезпечення комунікації безлічі користувачів застосовуються клієнт-серверні технології, що забезпечують такі переваги:

— мінімізація навантаження на клієнта;

— відсутність необхідності встановлювати пряме з'єднання між клієнтами;

— централізація діяльності з обслуговування клієнтів на одному місці.

Для прикладу, розглянемо архітектуру онлайн тренажера Lim English, що використовую архітектуру «клієнт-сервер».

Для повного опису загальної структури програми вдамося до опису платформи Microsoft .Net [3-5], так як ця платформа надає гнучкі засоби розробки розподілених додатків. У тому числі CLR (Common Language Runtime), єдине середовище виконання, заснований на виконанні коду, реалізованого різними мовами програмування. При цьому використовується перетворення такого коду в платформонезалежну проміжну мову — MSIL (Microsoft Intermediate Language), а компіляція під конкретну платформу відбувається на вимогу, в момент виконання.

В .Net серверна частина цієї програми може бути реалізована у вигляді віддаленого об'єкта, який використовується безліччю клієнтів. Цей об'єкт забезпечуватиме розділення та синхронізацію даних між клієнтами [1].

.Net Remoting — технологія віддаленої взаємодії, яка дозволяє звертатися до об'єктів, розташованих у різних доменах додатків і навіть різних машинах. Об'єкти клієнту передаються за значенням або посиланням. Для передачі об'єкта за значенням створюється повна копія, яка потім відправляється клієнту. При зверненні до віддаленого об'єкта за посиланням на стороні клієнта створюється об'єкт-проксі, використовуючи який клієнт отримує доступ до потрібного об'єкта прозорим способом, минаючи деталі взаємодії з віддаленим додатком.

Web-сервіси (XML Web Services) також можуть розглядатися як віддалені об'єкти [6]. У широкому сенсі, web-сервіс — це клас, розташований на web-сервері і використовує для надання доступу до своїх методів HTTP як протокол передачі даних і XML - як їх формат. Клієнт може звертатися до веб-сервісу за допомогою URL і взаємодіяти з ним через проксі-об'єкт. Проксі містить ті самі

методи, що й оригінальний клас, але його реалізація просто перенаправляє виклики методів оригінальному об'єкту.

Для динамічного створення проксі-об'єктів у .Net використовуються метадані. Зокрема, веб-служба може надати клієнту свій опис мовою WSDL (Web Services Description Language). Будь-який клієнт, який має можливість розуміти і спрямовувати SOAP-запити відповідно до цього опису, може викликати методи web-сервісу та взаємодіяти з ним за допомогою SOAP. Як альтернатива динамічному створенню проксі можна статично згенерувати вихідні файли, що описують віддалений web-сервіс, і включити їх до складу клієнтської програми (рис. 1.1).

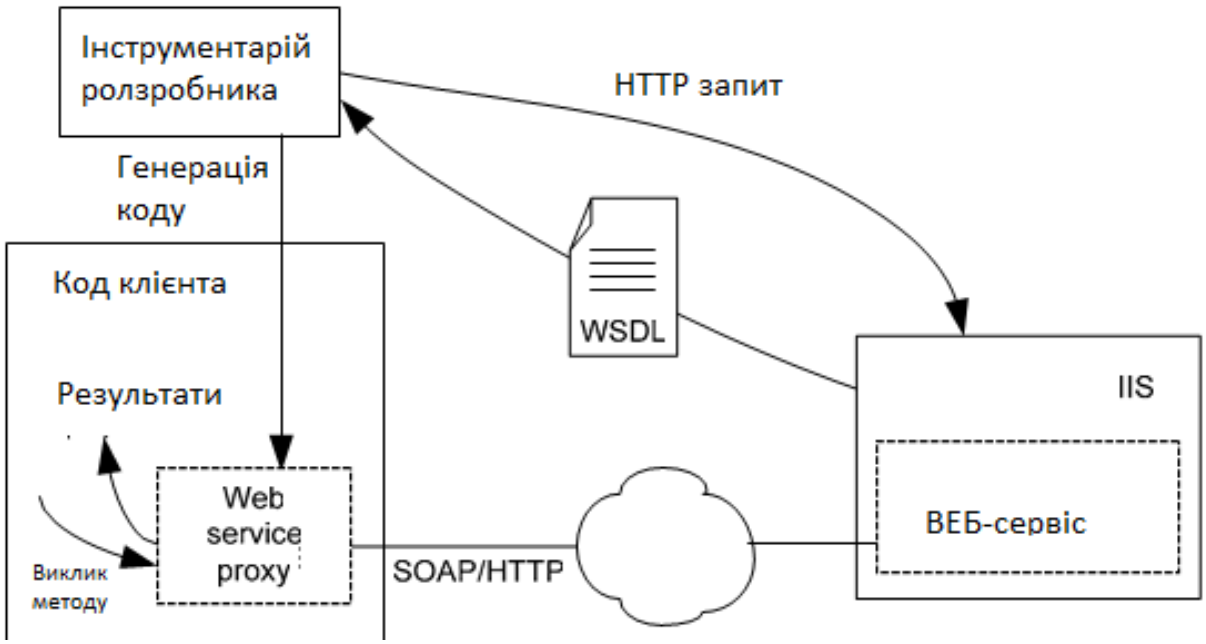

Рисунок 1.1 — Статична генерація проксі-класу web-сервісу

Така конфігурація підходить для взаємодії з клієнтом, що знаходиться за межмережевим екраном (див. рис. 1.2), а також дозволяє використовувати широку інфраструктуру .Net Framework.

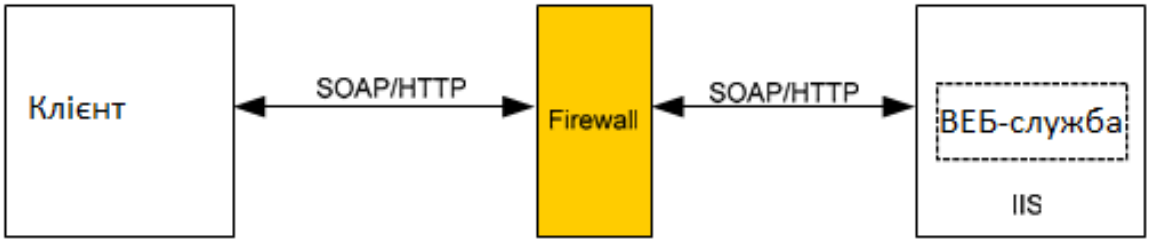

Рисунок 1.2 — Приклад виклику web-сервісу через firewall

Реалізація серверної частини як web-служба дозволяє не дбати про специфіку реалізації клієнтської частини, операційної системи користувача. Єдиними вимогами до клієнта з погляду взаємодії з web-службою є реалізація протоколу HTTP у середовищі клієнта та його здатність «розуміти» XML. Ще одним плюсом використання web-служби можна назвати підхід до клієнт-серверної взаємодії як віддаленого виклику методів класу. Це дозволяє абстрагуватися від деталей реалізації мережної частини та зосередитись на функціональній частині.

Клієнтську частину реалізуємо як Windows програми на платформі .Net. Це дає можливість скористатися потужним інструментарієм розробника, бібліотеками .Net, підтримкою генерації проксі-класу web-служби, збиранням сміття та іншими зручностями Microsoft. Хоча, як уже згадувалося, для взаємодії з web-службою немає особливих обмежень на мову та платформу реалізації клієнтської частини, крім розуміння HTTP та XML [7].

Клієнт та сервер можуть обмінюватись різними типами повідомлень. До їх складу входять:

— повідомлення про підключення та відключення користувача;

— повідомлення про зміну статусу користувача;

— текстові повідомлення, що містять текстове введення користувача;

— повідомлення чату;

— повідомлення, що містять дані про графічне введення – графічні команди.

Зауважимо, що повідомлення про підключення-вимкнення користувача та зміну його статусу використовуються виключно для інформування членів кімнати. Події, що викликають розсилку таких повідомлень, є результатом виклику методів сервера клієнтом або внутрішньою логікою сервера.

1.2 Аналіз типових сучасних програм-аналогів

Треба сказати, що тренажери по вивченню англійської мови мають свої переваги та недоліки. Однак, вони в постійному доопрацюванні,. бази безупинно поповнюються новими словами, а помилки програм успішно усуваються розробниками. Отже, давайте розглянемо та проаналізуємо тренажери для вивчення англійської мови.

1.2.1 Duolingo

Duolingo – це простий безкоштовний сервіс для вивчення іноземних мов в ігровій формі [8]. З його допомогою можна вивчити основну лексику на такі теми, як Сім'я, Будинок і так далі, а також навчитися будувати нескладні граматичні конструкції. У сервісі є вправи на аудіювання та вимову, але він розрахований насамперед на освоєння основ граматики та лексики.

Інтерфейс програми представлений на рис. 1.3.

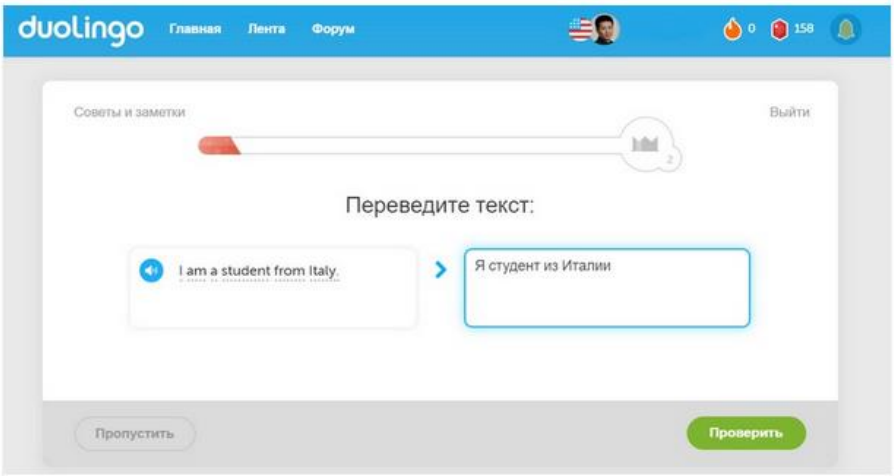

Рисунок 1.3 — Інтерфейс Duolingo

Плюси системи:

— безкоштовність, доступ до всіх тренувань, матеріалів і завдань абсолютно безкоштовний, враховуючи, що інші програми такого штибу рідко бувають безкоштовними, це незаперечна гідність;

— аудіювання, по-перше, добре, що воно є, по-друге, чудово, що аудіозапис можна слухати як у звичайному темпі, так і в уповільненому, що є неоціненним для початківців або для тих, кому важче сприймати мову на слух;

— вимова, повторюйте фрази, які почуєте, а програма проаналізує, чи всі ви правильно робите [6];

— навчальні матеріали, кожен урок супроводжується теорією, чи то рід іменників або умовний спосіб;

— постійне повторення лексики у різних уроках, слова повторюватимуться протягом різних рівнів, що збільшує шанс на їх запам'ятовування.

Мінуси:

— не всі пропозиції мають сенс тому мало застосовні практично, ось у якій життєвій ситуації може стати в нагоді фраза "Твій ведмідь п'є пиво"? вони навіть такі футболки випустили;

— уроки не систематизовано, навіть незважаючи на наявність довідкових матеріалів, чіткої картини у вас у голові не з'явиться;

— немає практики продуктивних навичок: листи та говоріння, перше є лише як переклад коротких речень, а друге – по суті тренування вимови;

— занадто легко для високих рівнів, нової лексики не так багато, тому на рівнях вище Intermediate варто спробувати щось інше.

1.2.2 Rosetta Stone

Програма Rosetta Stone була заснована в 1992 році, і на сьогоднішній день є одним із найстаріших інструментів для вивчення мови за допомогою комп'ютера (CALL - Computer Assisted Language Learning). Компанія пропонує традиційніший підхід до навчання, який називається «Динамічне занурення».

Rosetta Stone (рис. 1.4) відразу ж занурює користувача у мовне середовище, допомагаючи поєднати усне мовлення та зображення. Опис звучить тією мовою, яка вивчається. Проте переклад рідною мовою не передбачено. Спочатку це здається складним, але такий підхід найбільш близький до реального контексту, оскільки спонукає мислити логічно та знаходити швидкі рішення. Таким чином, користувач буде читати, чути, говорити і писати тільки мовою, яку вивчає.

Рисунок 1.4 — Інтерфейс Rosetta Stone

Переваги Rosetta Stone:

— різноманітні завдання, - не нудно вчитися;

— нативне вивчення мови - ви не просто запам'ятовуєте слово, а закріплюєте його з певним чином;

— можна підлаштувати програму під себе: якщо ви не хочете займатися говорінням або граматикою, можете просто відмовитися від них;

— зручний інтерфейс;

— для вивчення доступні більше 30 іноземних мов: окрім найпопулярніших є хінді, китайська, філіппінська та інші).

Недоліки:

— не пояснюються жодні правила: за відсутності мінімальної бази знань може бути складно;

— інтерфейс англійською мовою (хоча якщо врахувати, що мета авторів - занурити вас у мовне середовище, - це не такий вже й мінус);

— лише кілька уроків у безкоштовному доступі.

1.2.3 Lim English

Lim English [3] – це онлайн-порграма для вивчення англійської мови за допомогою смартфона чи комп'ютера. Платформа добре підхоить для бажаючих підвищити вже існуючий рівень володіння мовою і тих, хто починає з нуля. Процес навчання заснований на унікальній авторській методиці, складовими якої є самостійні онлайн-заняття.

У вільному доступі для учнів є захоплюючі відео-уроки, вправи різної спрямованості, правила та тренажери. Звуковий матеріал озвучений професійними дикторами, які є носіями мови.

Переваги, характеристики та корисні функції [3]:

— приємні голоси дикторів з правильною вимовою;

— цікаві тексти, завдання;

— грамотно структуровані уроки;

— дуже цікаво уроки;

— недорогий додаток;

— підходить для початківців;

— багато матеріалу, постійно додається новий;

— стовідсотковий метод для запам'ятовування слів та фраз. та ще й писати грамотно починаєш дуже швидко.

Інтерфейс програми зображено на рис. 1.5.

Рисунок 1.5 — Інтерфейс Lim English

Недоліки та обмеження:

— програма для телефону не дає всіх функцій;

— треба багато друкувати;

— немає мобільного додатку;

— якнайбільше різноманітних вправ.

1.3 Особливості використання середовища Visual Studio та мови програмування C++ для створення тренажеру з англійської мови.

C++ — це мова програмування середнього рівня, розроблена Б’ярном Страуструпом, починаючи з 1979 року в Bell Labs. C++ працює на різних

платформах, таких як Windows, Mac OS і різні версії UNIX. У цьому підручнику C++ використовується простий і практичний підхід до опису концепцій C++ для початківців до досвідчених інженерів-програмістів.

C++ є обов'язковий для студентів і працюючих професіоналів для того, щоб стати чудовим інженером-програмістом. Ось деякі з ключових переваг вивчення C++:

— дуже близький до апаратного забезпечення, тому, що можна працювати на низькому рівні, що дає великий контроль щодо управління пам'яттю, кращої продуктивності та, нарешті, надійної розробки програмного забезпечення;

— програмування на C++ дає чітке розуміння об'єктно-орієнтованого програмування, зрозумілу реалізацію поліморфізму низького рівня;

— одна з усіх мов програмування, яку люблять мільйони розробників програмного забезпечення. хороший програміст на C++ніколи не будете сидіти без роботи, і, що більш важливо, буде отримувати високу платню за свою роботу;

— C++ є найбільш широко використовуваною мовою програмування в прикладному та системному програмуванні, таким чином, можна вибрати сферу інтересів розробки програмного забезпечення [8];

— дає зрозуміти різницю між компілятором, компонувальником і завантажувачем, різними типами даних, класами зберігання, типами змінних, їх областями тощо.

C++ присутній майже в кожній області розробки програмного забезпечення. Ось декілька з них:

— розробка прикладного програмного забезпечення, використовується при розробці майже всіх основних операційних систем, таких як Windows, Mac OSX та Linux, крім операційних систем, основна частина багатьох браузерів, таких як Mozilla Firefox і Chrome, була написана на C++, C++ також використовувався при розробці найпопулярнішої системи баз даних під назвою MySQL;

— розробка мов програмування, широко використовується при розробці нових мов програмування, таких як C#, Java, JavaScript, Perl, UNIX C Shell, PHP і Python, Verilog тощо;

— програмування обчислень, на C++ є найкращим для використання вченими через швидку швидкість та обчислювальну ефективність;

— розробка ігор, надзвичайно швидкий, що дозволяє програмістам виконувати процедурне програмування для інтенсивних процесорних функцій і забезпечує кращий контроль над апаратним забезпеченням, через що він широко використовується при розробці ігрових двигунів;

— вбудована система, активно використовується для розробки медичних та інженерних програм, таких як програмне забезпечення для апаратів МРТ, високоякісних CAD/CAM систем тощо.

Цей список можна продовжувати. Є різні області, де розробники програмного забезпечення, які із задоволенням використовують C++ для створення чудового програмного забезпечення.

Microsoft Visual Studio — це інтегроване середовище розробки (IDE) від Microsoft (рис. 1.4). Його використовують для:

— розробки комп'ютерної програми;

— веб-сайту;

— веб-додатку;

— веб-сервісу;

— мобільного додатку.

Visual Studio використовує платформи розробки програмного забезпечення Microsoft, такі як Windows API, WinForms, WPF та ін.

Visual Studio містить редактор коду, який підтримує IntelliSense, а також методи рефакторингу коду. Інтегрований відладчик працює і як налагоджувач на рівні коду, і як налагоджувач на рівні машини. Інші вбудовані інструменти включають профайлер коду, конструктор для створення додатків із графічним інтерфейсом користувача, веб конструктор сайтів, конструктор класів і конструктор схем баз даних. Дане середовище підтримує плагіни, які розширюють функціональність майже на кожному рівні, включаючи додавання підтримки систем контролю джерел (наприклад, Subversion і Git) та додавання

нових інструментів розробки та тестування, таких як редактори та графічні конструктори для мов, що стосуються домену, або наборів інструментів для інших сторін розробки програмного забезпечення.

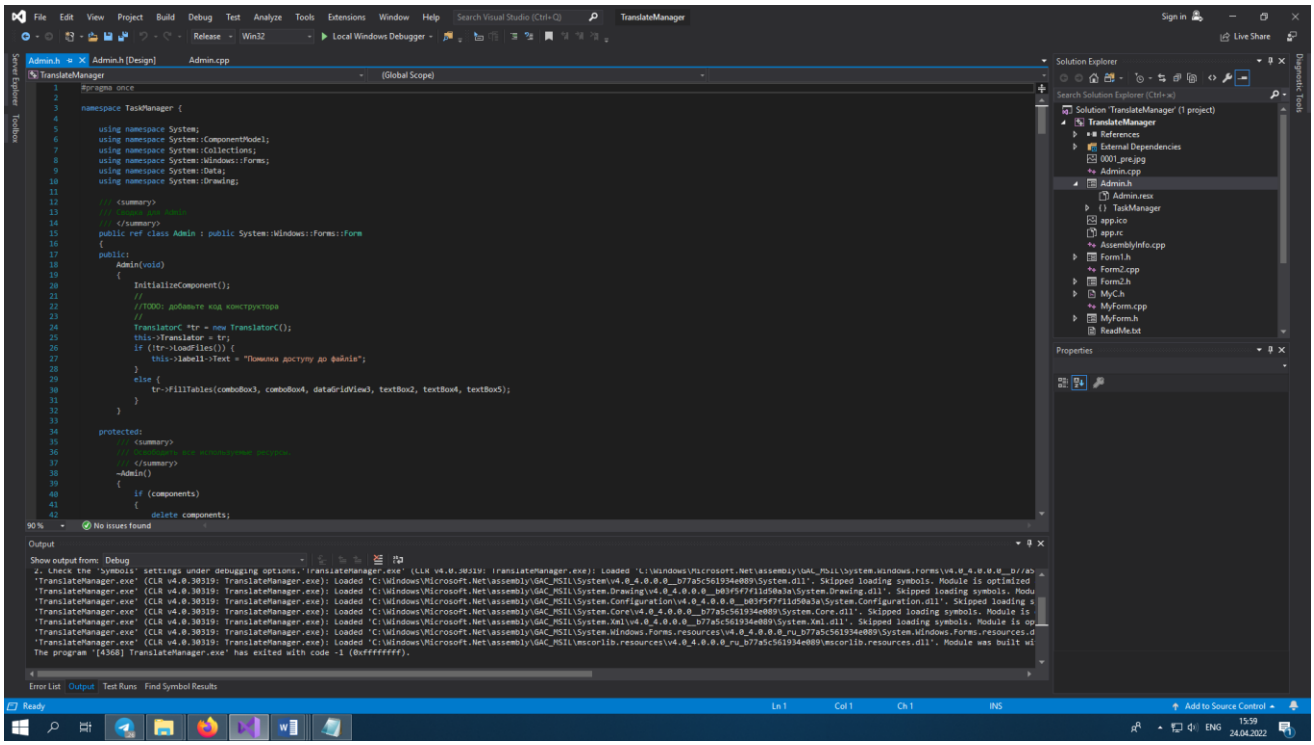

Рисунок 1.4 — Інтерфейс Microsoft Visual Studio

Visual Studio не підтримує жодну мову програмування, рішення чи інструмент по суті; замість цього він дозволяє підключати функціональні можливості, закодовані як VSPackage. Після встановлення ця функція доступна як Сервіс. IDE надає три сервіси:

— SVsSolution (надає можливість перераховувати проекти та рішення);

— SVsUIShell (забезпечує функціональність вікон та інтерфейсу користувача);

— SVsShell (займається реєстрацією VSPackages).

VS підтримує запуск кількох екземплярів середовища (кожен із власним набором VSPackages). Екземпляри використовують різні кули реєстру (див. визначення MSDN терміна «куст реєстру» у значенні, що використовується тут)

для зберігання свого стану конфігурації та розрізняються за їхнім AppId (ідентифікатором програми). Примірники запускаються спеціальним для AppId .exe, який вибирає AppId, встановлює кореневий вулик і запускає IDE. VSPackages, зареєстровані для одного AppId, інтегруються з іншими VSPackages для цього AppId. Різні версії продуктів Visual Studio створюються за допомогою різних ідентифікаторів додатків. Продукти видання Visual Studio Express встановлюються з власними ідентифікаторами AppId, але продукти Standard, Professional і Team Suite мають однаковий AppId. Отже, видання Express можна встановлювати поряд з іншими випусками, на відміну від інших випусків, які оновлюють ту саму інсталяцію. Професійне видання включає додатковий набір VSPackages у стандартній версії, а командний набір включає додатковий набір VSPackages в обох інших випусках. Систему AppId використовує оболонка Visual Studio.

2 РОЗРОБКА СТРУКТУРИ ТА АЛГОРИТМІВ РОБОТИ ПРОГРАМИ

2.1. Розробка структури проекту

Проект має наступну структуру (див рис 2.1.)

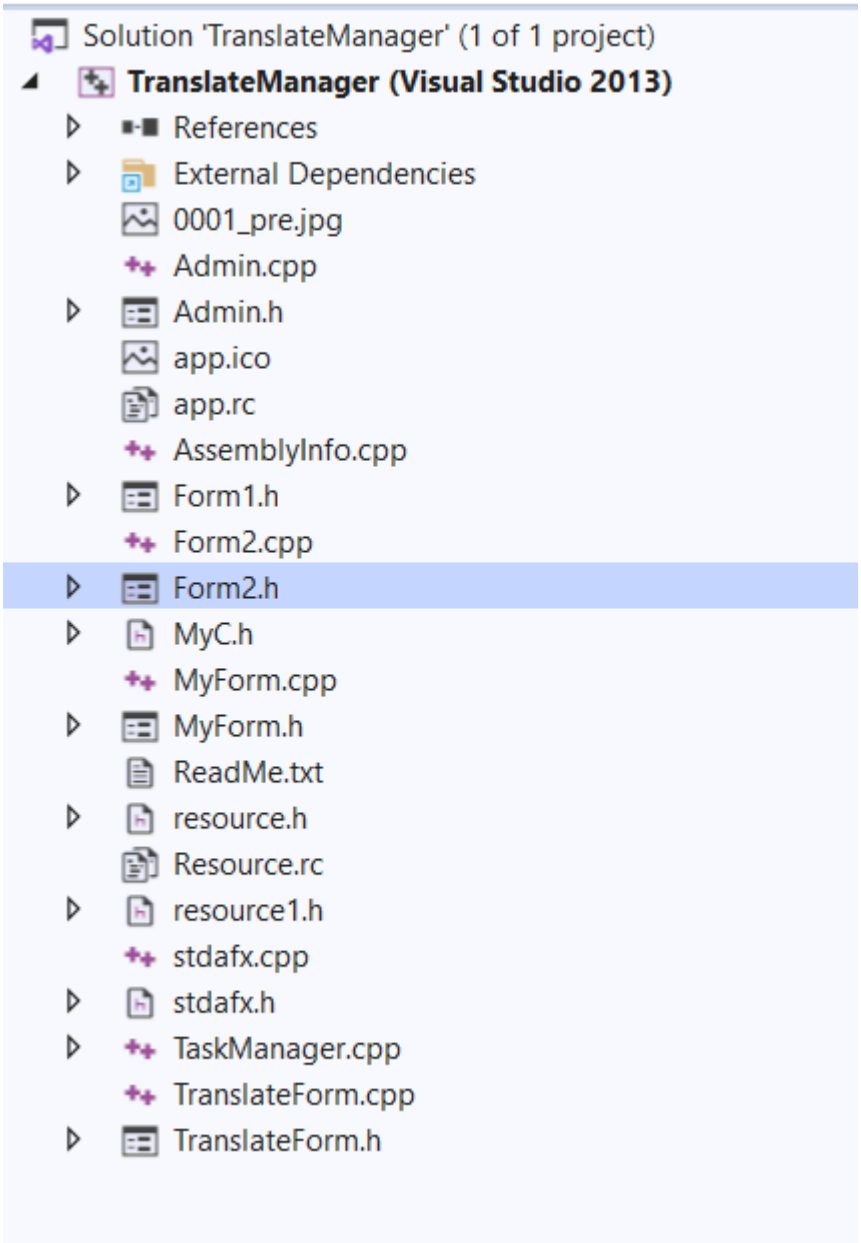

Рисунок 2.1 — Структура проекту

Як видно з рисунку 2.1. проект має 5 форм та 7 файлів cpp. Розглянемо їх детальніше.

Файл AssemblyInfo.cpp містить інформацію про збірку, назву програми та більшість атрибутів, які можна переглянути в диспетчері задач.

Було виявлено одну відмінність для цього файлу: він пов’язаний зі значеннями, які повідомляються під час викликів Assembly.GetReferencedAssemblies.

Тож для файлів vcxproj схоже, що VS_VERSIONINFO здатний оновлювати вміст, який ви знайдете на вкладці властивостей файлу, але AssemblyInfo.cpp здатний надати версію GetReferencedAssemblies. У C# ці дві області звітності, здається, уніфіковані. Можливо, є спосіб спрямувати AssemblyInfo.cpp на розповсюдження деталі файлу якимось чином, але те, що я збираюся зробити, це дублювати інформацію про збірку в обох місцях на етапі попередньої збірки.

Файл TaskManager.cpp містить функцію main, тобто основну точку входу в програму. Це звичайний згенерований код, який встановлює візуальний стиль за умовчанням, та запускає головну форму програми. Головна форма програми в даномувипадку називається Form1.

Інші файли містять підключення файлів бібліотек до проекту.

2.2 Розробка структури віконного додатку

Інтерфейс користувача (UI) використовує елементи для створення форми проекту і забезпечує узгодженість продукту, що робить його зручним для користувача і простим у навігації без особливих зусиль.

UI елементи зазвичай поділяються чотири категорії:

— елементи керування введенням (Input Controls) – дозволяють користувачам передавати інформацію до програми;

— компоненти навігації (Navigation Components) – допомагають користувачам переміщатися програмою чи веб-сайтом, загальні навігаційні компоненти включають панелі вкладок та головне меню програми;

— інформаційні компоненти (Informational Components) – містять інформацію, наприклад написи чи підказки;

— контейнери (Containers) – об’єднують інші елементи інтерфейсу разом.

В даному проекті для розробки користувацького інтерфейсу було використано Windows Forms.

Windows Forms — це платформа користувача інтерфейсу для створення класичних додатків Windows. Вона забезпечує один з найефективніших способів створення класичних додатків за допомогою візуального конструктора в Visual Studio.

Було спроектовано 5 форм:

— головна форма;

— форма адміністратора;

— форма статистики;

— форма перекладу;

— інформаційна форма.

2.3 Розробка для завантаження даних із файлу

Для виведення даних для тестування знань користувачів на знання англійської мови використовуються 5 файлів для зберігання такого роду інформації:

— у файлі із назвою «eng» містяться словник англійських слів;

— файл із назвою «links» містяться налаштування часу на відповідь по кожному слову;

— у файлі із назвою «p» міститься пароль адміністратора;

— у файлі із назвою «param» містяться параметри системи;

— файл із назвою «ukr» містять словник переклату слів.

Загальна робота методу завантаження файлів у програму показана на блок-схемі на рисунку 2.2.

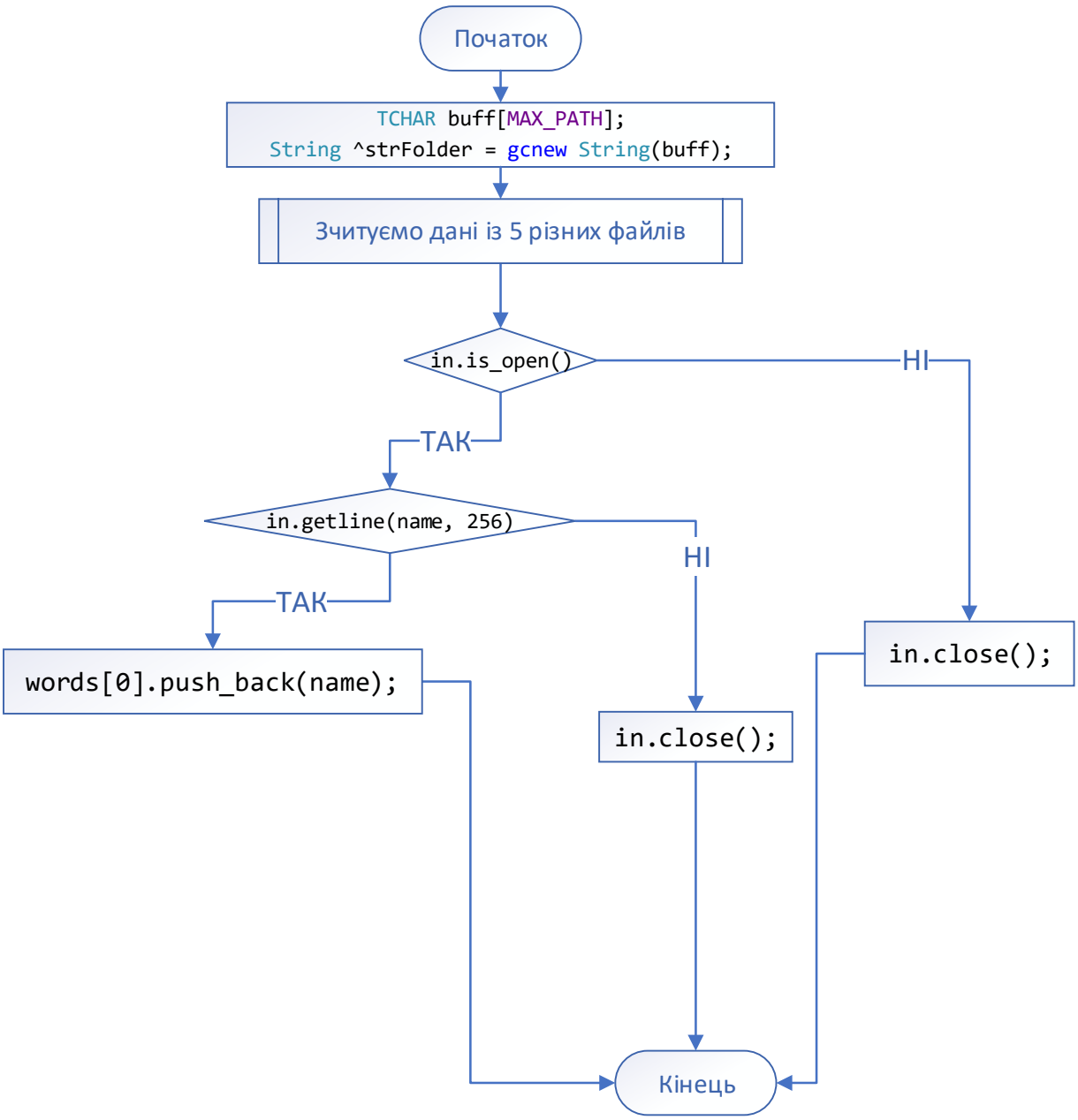

Рисунок 2.2 — Алгоритм завантаження інформації із файлу

2.4 Розробка тестування знань користувача програми

Для того, щоб розпочати тестування необхідно вибрати мову, знання якої необхідно перевірити. Після цього, програма зразуж включивши таймер виводить запитання у вигляді слова. Користувачу програми необхідно правильно написати його переклад і зробити це за вказаний час. Про те скліки часу залишилось, програма буде повідомляти. Якщо ж час вичерпано, то програма виведе наступне слово, аж поки не буде закінчена перевірка всіх слів у словнику.

Алгоритм роботи наведено на рисунку 2.3

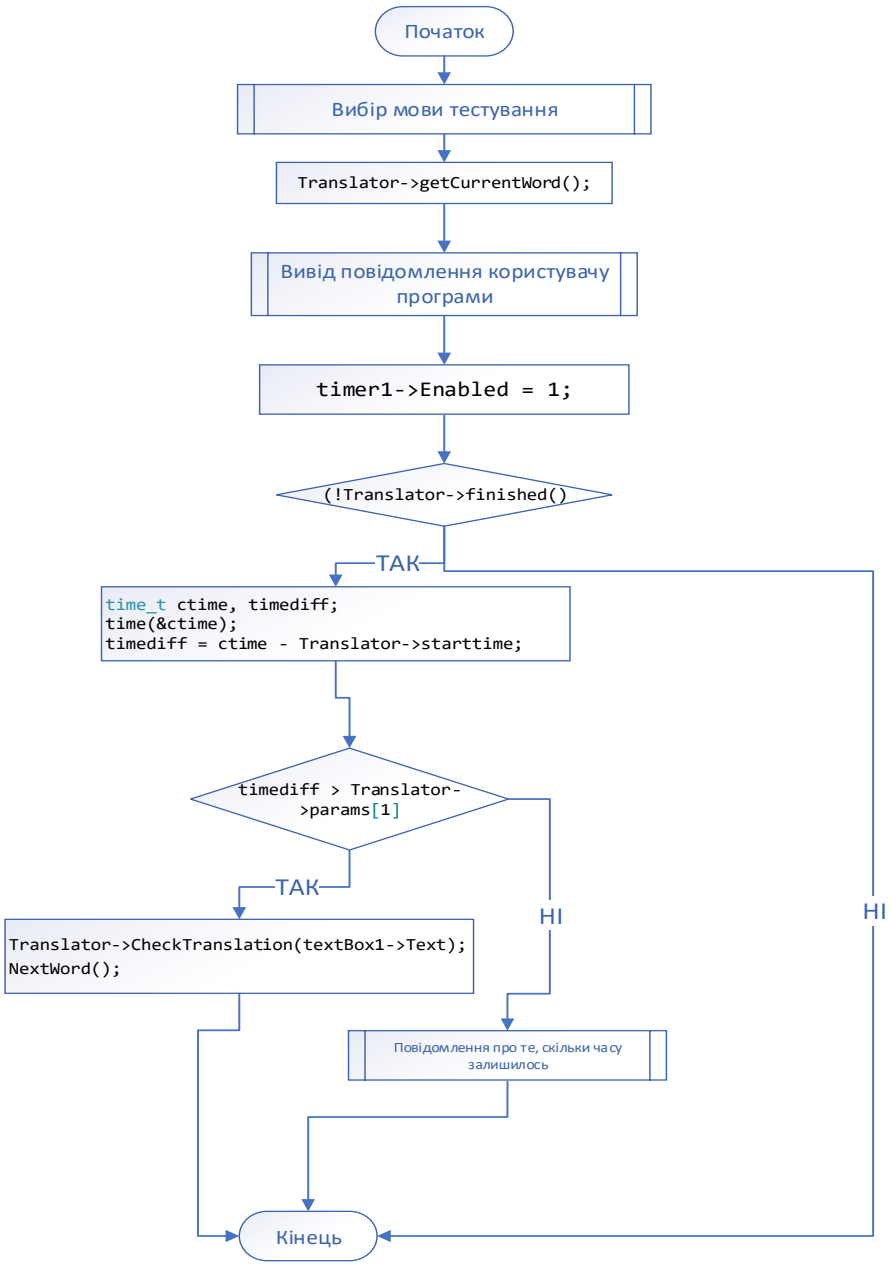

Рисунок 2.3 — Алгоритм роботи тестування знань

2.5 Розробка для збереження даних у файл

Для збереження даних тестування знань користувачів на знання англійської мови використовуються 5 файлів для зберігання інформації. Інформація зберігається після того, як користувач натисне відповідну кнопку в програмі.

Загальна робота методу збереження файлів у програму показана на блок-схемі на рисунку 2.4.

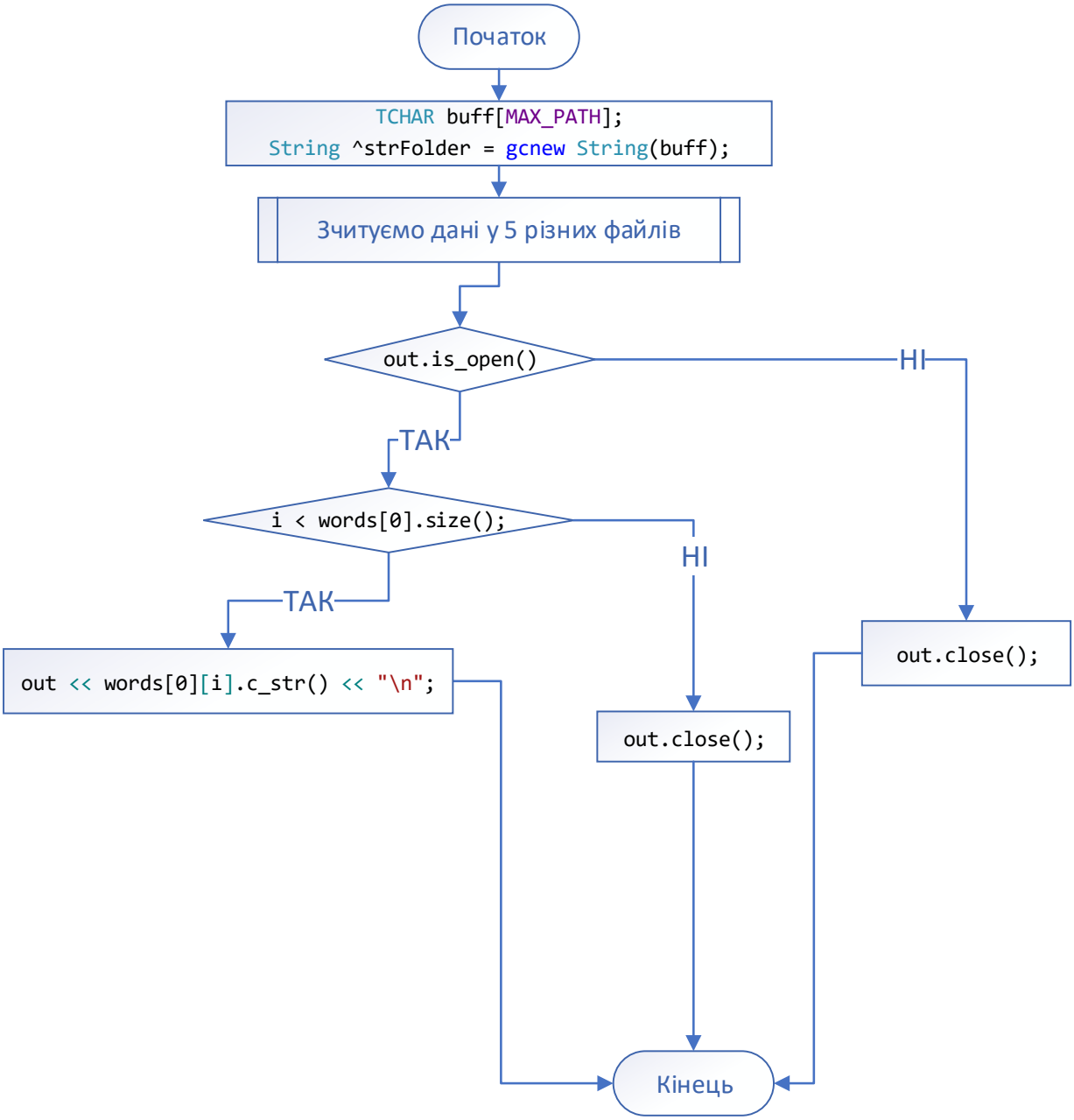

Рисунок 2.4 — Алгоритм збереження дахиї у файл

2.6 Історія мов програмування

Давайте почнемо із самого початку. На самому початку комп'ютери не мали навіть клавіатури! Тобто все було дуже погано - у них не було ні клавіатури, ні екрана, були перфокарти. Відповідно, чи то штирки туди

засовували, чи там світлом світили. Якщо є дірочка (або, навпаки, ні) — це означало нулик або одиничку. І програми на той час писали за допомогою машинних кодів — кожна операція в комп'ютері (складання, віднімання, якісь складніші операції) мала якийсь код машинний. Люди самі по табличці вибирали цей код, усілякі адреси в пам'яті, все це вибивали руками і засовували в зчитувач — і все це вважалося. Звичайно, робота програміста була, напевно, тоді не дуже цікавою — робити дірочки — і з розвитком науки і техніки, звичайно, почали вигадувати всякі, більш «цікаві» штуки. Наприклад, асемблер (Assembler), який вже трохи полегшував життя.

Ну, як він полегшував життя? Замість запам'ятовування того, що там якийсь «чарівний» код у команди, використовувалися всякі слова, схожі на «людську» англійську мову — якісь add або mov — ну і потім перераховувалися регістри або області пам'яті, змінні, з якими потрібні ці операції робити. Але ясна річ, що це загалом теж вимагало досить великої напруги розуму, щоб пам'ятати, в якому регістрі у нас що лежить, де якісь змінні і що взагалі відбувається. Чому так сталося? Тому що комп'ютери були «дурні» і нічого розумнішого зрозуміти не могли. Взагалі-то і зібрати з асемблера машинний код теж вимагає часу, пам'яті (на ті часи було звичайно мало її).

Поступово ставало зрозуміло, що розробляти такі великі складні програми дуже складно. Продуктивність програміста в цих командах була гранично низькою — тобто він писав кілька рядків на день (осмислених), і кожен рядок нічого особливо не робив — якісь прості арифметичні дії. І людям хотілося зробити мови набагато більш схожими на людську мову, англійською зокрема, щоб писати програми було легше та зручніше. І пішло-поїхало!

Однією з перших мов став Fortran. До речі, його також вибивали на перфокартах — були спеціальні перфокарти для вибивання програм на Fortran'і. Але якщо ви зараз візьмете цей Fortran — на мою думку, він навіть десь між 50-60 роками. з'явився і спробуєте на ньому щось написати, то вам буде дуже неприємно, я вам гарантую! Сучасний Fortran ще живий, але вже досить сильно відрізняється від того, що було раніше.

Інші мови — зараз напишу одну штуку, про яку напевно ви якщо й чули, то тільки на будь-яких заходах, де розповідають про історію програмування - це COBOL. Це була мова для написання бізнес-додатків. Що таке бізнес-програми? Якісь транзакції в банках, ще щось, усе це писали на Коболі. У нас, ясна річ, не дуже популярний. Я думаю, програміста на Коболі, в Москві, ви знайдете з великими труднощами. А де-небудь не в Москві - з ще більшою працею. Але, що дивно, ще 10 років тому більше половини всього коду, написаного людством, було написано на Коболі. І досі значна частина всяких банківських транзакцій йде за допомогою програм, написаних на ньому (COBOL), і досі люди на ньому щось пишуть.

Є ще «кумедна» мова, він називався Алгол (68-та версія, що характеризує рік його створення). Це алгоритмічна мова. Загалом вони щось там уміли, але нам зараз не дуже цікаво, що вони вміють.

Алгол вигадали в Європі, а Фортраном користувалися здебільшого в Штатах — великих відмінностей немає. Яка тенденція є помітною? Спочатку все було складно і щоб писати потрібно було мало не інженером, електротехніком, розуміти, де які контакти замикаються і ще щось для програмування. Потім теж треба було сидіти з листочками і рахувати пам'ять, стежити за нею. І поступово все ставало простіше, простіше, простіше і далі ще простіше для програміста — якнайменше думати людині, якнайбільше робити автоматично. Приблизно в кінці цього періоду (лектор вказує на Алгол і Кобол) починають з'являтися мови, які в якомусь сенсі «дожили» до наших днів.

BASIC. Можливо, досі деякі люди на ньому щось пишуть, принаймні я бачив, що в деяких закладах навчають на QBasic'і таке синє віконце, де написано «1989». Загалом, «на повну силу» живе! Він вигадувався як мова для непрограмістів. На той час програмістом була така дуже спеціалізована професія. А тут тобі кажуть: «От у нас є класна мова Basic, і будь-яка розумна людина візьме та напише програму на ній – легко». Знову ж таки Бейсік і сучасний Бейсік — це величезна різниця. Будь-які ці рядки з нумерацією через 10, всякі GOTO та інший жах — вони до сучасного Бейсика вже не мають жодного стосунку і навіть до Бейсика 89-го року вони мають мало відношення.

Ще одна кумедна історія — мова Паскаль, широко відома у колах ВНЗ, в основному в Україні і в країнах колишнього Радянського Союзу. Він використовувався і продовжує використовуватися напрочуд як навчальну мову. У всьому світі він менш поширений, але теж живе і живе. Є така людина Вірт - ось він вчений, теоретик. Він брав участь в обговоренні Алгола, йому не сподобалося те, що вийшло, і він вигадав свою мову - Pascal. А потім фірма Borland (і до цього багато інших фірм Apple займалася, зокрема) взяли і все зіпсували. Він мав гарну теорію, стрункy — «все буде добре», — а вони взяли і напхали туди того, що людям потрібне для роботи. Ну і вийшло не так гарно, як він хотів.

І, нарешті, C. Сі вигадали інженери. Якщо Паскаль вигадав вчений, то Сі придумали Керніган і Рітчі, вони працювали інженерами в Bell. Як це відбулося? Тоді цими мовами (лектор вказує на Fortran, COBOL, Algol) нічого системного написати не можна. Що таке "системне"? Наприклад, операційну систему, драйвера якісь, ще щось. Ці мови призначалися для математичних розрахунків, для бізнес-розрахунків, для такого. А решту писали на Асемблері. Були якісь мови, вони зараз померли, тобто мова Сі з'явилася не відразу від Ассемблера, а через якісь проміжні речі.

Керніган і Рітчі любили грати в іграшку Asteroids — літає космічний корабель, і тут є астероїди, він у них стріляє, і вони розвалюються на частини. Вони мали сервер, на якому вони грали, але там було багато народу, і іграшка гальмувала. І вони виявили у себе десь у кабінеті, що вони мають якийсь комп'ютер, яким ніхто не користується. Але була проблема — він іншої архітектури був, а гра була написана на Ассемблері.

Вони її переписали, звичайно, навіть впилили якісь фічі, щоб грати на ньому. Але це навело їх на думку, що переписувати під нову архітектуру щоразу це не дуже розумно. І вони вирішили написати таку мову високого рівня, яка підходитиме для системного програмування, тобто, в якій можна буде керувати пам'яттю, в якій можна буде розуміти дещо лежить і як звертатися до цих шматочків пам'яті. І так виникла мова Сі, який вплинув на все подальше. Вони

всі (лектор вказує на Алгол, Фортран та інші згадані мови) вплинули, але ось Сі - прямо так ...

Відповідно, це була основна мова в Unix — операційній системі, яка на той час була ще популярнішою, ніж зараз. І приблизно до 80-х років ситуація була якась ось така (лектор вказує на Basic, C та інші згадані мови). Припустимо, що все це у нас вже потихеньку померло (лектор стирає згадки про Асемблера, Фортрана і Алгол) ... І в 80-ті роки комп'ютери стали менше, розумнішими, дешевшими, і людям захотілося будь-яких дивно, щоб жити стало ще краще, жити стало ще веселіше.

У нас на той момент C++ застосовувався майже для всього, що потрібно було писати не для Інтернету, не для обробки тексту, а для додатків, для операційних систем, для іграшок — загалом, для чого завгодно. Але C++ — це моторошна мова насправді. Тому що, по-перше, він успадковував через зворотну сумісність усі проблеми Сі. Там можна було, як і раніше, убитися мільйоном різних способів, тих самих, що були в Сі (природно, додалися і нові способи в C++). При цьому, якщо писати все добре і правильно, як було задумано авторами C++, то, звичайно, старими сишными способами вже убитися було не можна, і як їх стало менше. Проте, він мав дуже дивну об'єктну модель своєрідну. Розбиття програми на модулі, на шматочки якісь взагалі прийшло з Сі (якщо include ви вмієте писати на Сі або на C++ — фактично це було задумано як просто вставити текст бібліотеки у вашу програму, в результаті, коли ви пишите купу інклюдів у вас все — якщо «примітивно», як це було на самому початку — у вас все вставляється в один файлик і потім все це дуже довго компілюється, тому що по кілька разів ходить. версії стали ще кращими.

Загалом недоліків у C++ дуже багато. Кваліфікація у програміста повинна була бути висока, щоб писати на C++, і коштували такі програмісти дорого. А комп'ютери у нас все швидше і швидше вважають, стають дешевшими, люди купують собі нові комп'ютери і хочуть більше додатків, більше іграшок для телефону, загалом — більше радості.

Так і виникла Java (Ява). Там теж пов'язана досить кумедна історія, як назва з'явилася у цієї мови. Там програмісти, вони весь час п'ють каву і тоді було

модним пити каву, яка на острові Ява зростала. Мова замислювалася як мова для вбудованих приладів, зокрема для кавомашини. Ось так і вийшла назва.

Що взагалі почалося з неї, що в ній було гарне і чому вона завоювала велику популярність? По-перше, вони позбулися спадщини Сі, повністю. Жодних вказівників, значно менше способів відстрілити собі якусь частину тіла і все зламати. По-друге, вони впровадили набагато свіжіші ідеї щодо об'єктної моделі — тобто C++ з'явився значно раніше, ніж Java і використовував більш архаїчну, «дику» модель об'єктну. Ну а тут вона була вже більш продуманою тоді вже, і теоретично люди думали, і на практиці застосовували і зробили все набагато крутіше.

І, нарешті, третє. У нас програми на Джаві збиралися не в машинний код, а в код для віртуальної машини. Програми збиралися в якесь проміжне уявлення і потім, за допомогою цієї машини вже виконувались. Що це дало? По-перше, воно гальмувало, по-друге, воно жерло пам'ять зі страшною силою, по-третє воно було переносимо куди завгодно (теоретично) — хоч на кавоварку, хоч на кавомолку, хоч на комп'ютер, хоч на мобільний телефон. Це, з одного боку, добре, тобто ви написали просто реалізацію віртуальної машини, потім свої джавівські програми запускаєте скрізь. Але, з іншого боку, погано, що на тому ж телефоні тоді мало пам'яті, була низька продуктивність і все це ще додатково починало тупити і гальмувати.

Але навіть не це головне, для чого взагалі мова вигадувалась. Мова Джава вигадувалась щоб знизити вимоги до кваліфікації програмістів. Тобто, погані програмісти можуть писати хороші програми на Java, тому що вона не дозволяє писати погані програми — там немає коштів, щоб писати програми погано. Там можна писати лише добре, програми. Добре, у розумінні творців мови. Тобто, якщо на Сі, на C++, на Пітоні, на чому завгодно ми можемо розвести зі свого проекту моторошний смітник якийсь, де у нас все лежить упереміш, збирається годинами і там ще чогось. То в Java смітник розвести теж можна, але для цього вже треба докласти якихось зусиль. Тобто, за замовчуванням, там виходить не “смітник”, виходять інші проблеми, що там щось успадкували-спадкували —

загалом на один осмислений рядок виходить десять не дуже осмислених. Зате може такий, середньої кваліфікації, програміст писати досить якісний код.

Ми майже прийшли до кінця. У нас, що з'явилося — це .Net (дотне), ну і зокрема нас цікавить C# (майже те саме [лектор вказує на Java], тобто там відмінності в деталях, якщо вибиратимете між ними — дивіться, де грошей більше сплачують).

І ще одна штука – JavaScript. Не має жодного відношення до мови Java, з'явився в тому ж році - слово було модне, вони ліцензували торгову марку, щоб використати.

На що, головне, слід звернути увагу? (Лектор малює стрілки від C++ Java, .Net, C#, JavaScript і PHP). Щоб написати простеньку програмку однією з цих мов, та й на багатьох інших - якщо ви знаєте C++, вам взагалі нічого більше знати не треба - ви берете і пишете на C++, а потім додаєте долари на початку, ще щось робите по дрібниці і у вас вона починає працювати на що завгодно (лектор свідчить про мови, яких були відведені стрілочки від C++). Тобто вони гранично схожі у якихось простих речах. Якщо ви вирішуєте якісь шкільні завдання, навчальні завдання, ще щось (не проектуєте великий проект - у вас один файлик, який читає числа, виводить числа в консолі, ще чогось робить), то різниці майже ніякої немає між цими мовами. Відомо, що JavaScript і PHP вони спеціалізовані, у них все трохи інакше. А ось тут (лектор вказує на Java та C#) взагалі гранично мало різниці.

Ну і що треба взагалі сказати про сучасні мови. Зараз багато проектів не живуть якоюсь однією мовою, тобто у них частина якась живе однією мовою, частина — іншою, ще якась частина — третьою. Наприклад, якщо у вас якийсь веб-додаток, який обробляє дикі обсяги інформації, звернення до дисків (навіть не до баз даних, вони настільки величезні, що там навіть база даних не тягне якась вже написана), напевно, написані на якому- то низькорівневому Сі, щоб дико швидко писати на диск і таке інше. Звичайно, писати весь проект на Сі не варто. Можливо, там якась проміжна логіка, написана на Java, яка звертається до Сишних функцій для швидких звернень. Ну а фронтенд (то на що дивиться користувач), звичайно, вже написано на чомусь, на якихось скриптах, на тому,

що безпосередньо виконується браузером (JavaScript). І все це живе разом та успішно взаємодіє.

У розробці якихось програм, навіть великих, іноді люди що роблять? Вони беруть і на Пітоні пишуть прототип (як воно все працюватиме), накидають якусь архітектуру продумують. Писати на ньому реально дуже швидко - вони накидали прототип, поекспериментували з ним і сказали: «О! Ось так круто! І повністю переписали. Здавалося б, вони двічі зробили роботу, від цього вдвічі більше часу пішло (ну, у півтора). Але немає! Часто виявляється, що такий спосіб непоганий, тому що, якщо ви напишете відразу на чомусь, наприклад на Java, а потім вирішите: «Ні, давайте рефакторинг, міняємо архітектуру повністю і таке інше,» — то витратите в 10 разів більше часу . Такі речі також існують і живуть.

Тепер поговоримо про те, чому деякі хороші на вигляд мови не вижили, чи живуть в дуже обмеженому просторі. Коли Вірт побачив, що зробили з його Паскалем погані фірми Apple, Borland і таке інше, він придумав мову ще краще - Oberon. Він тільки був дико мінімалістичний - тобто там було дуже мало команд (Рядки? Навіщо нам рядки? Ми зробимо масив символів!). Ну і чогось не пішло в нього, як могло б піти.

Ще одна штука. Американські військові попросили розробити їм теж круту мову, якою все працює і все можна написати. В результаті вийшла досить монструозна мова Ada, якою, щоправда, досі щось пишуть, але знову ж таки — для військових тільки.

В чому проблема? Чому деякі мови на зразок Python, який ніяка компанія його не підтримувала спочатку, захопили ринок. PHP, який ще й погано спроектований, теж сам по собі взяв та захопив ринок (більшу частину). А всякі мільярди доларів вкладені (лектор вказує на Ada) і нікуди не пішли, нічого не сталося. З чим це пов'язано? Це пов'язано з тим, що немає інфраструктури навколо цих мов. Тобто мова може бути відмінна, але поки що немає документації, поки що немає спільноти, яка вміє відповідати на запитання (на Stack Overflow) і, нарешті, найголовніше, поки що немає великої кількості бібліотек, мова не вистрілює. Тобто ви, наприклад, захотіли на Обероні написати сайт. А що такого, чому б і ні? І починається морока ... Веб-сервер ви не можете

підняти свій на Обероні, щоб потестувати легенько, які бібліотеки ви підключити не можете, тому що їх на Обероні немає. І все це через якісь милиці робиться, сили йдуть, і загалом ви плюєте та пишете на чистому Сі свій сайт замість Оберона. А добре живуть ті мови, які вміють користуватися бібліотеками від інших мов. Той самий Пітон у тих місцях, де він гальмує. Та й взагалі будь-які стандартні речі типу сортування і ще чогось написані на Сі, і він (Python) вміє з ними взаємодіяти.

Java також має Java Native Interface. Це по суті Сі, тобто там (на мою думку, весь час хочуть заборонити, але здається ще не заборонили) ці мови можуть взаємодіяти з вже існуючими бібліотеками (в основному Сишними). І рахунок цього беруть і працюють. Зрозуміла ідея, яку я намагаюся донести до вас, так? Не пишіть мовами, які не вміють підключати Сишну бібліотеку. Ну якщо ви хочете користуватися чимось класним. Ну, і поступово вони (мови) обростають своєю якоюсь інфраструктурою. І живуть якось добре.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПЛАТФОРМИ

3.1 Варіантний аналіз і обгрунтування вибору програмних засобів

Перед початком розробки програми необхідно провести огляд доступних програмних засобів та обрати такі, що якнайкраще підходять для реалізації даної програми.

Для розробки буде проведено огляд наступних мов програмування:

— C++;

— Python;

— Java;

— C#.

C++ — компільована мова програмування високого рівня, розроблена Б’ярном Страуструпом. Перша версія мови була випущена у далекому 1979 році. Найактуальніша версія мови C++ 17 вийшла відносно нещодавно у 2017 році. C++ успадкувала синтаксис і основні аспекти мови C. По суті, C++ є надмножиною мови C, на що також вказує початкова назва — «C with classes» — «Сі з класами». Мова C++ підтримує парадигму об’єктно-орієнтованого програмування, характеризується високою швидкодією за рахунок того, що код програми компілюється напряму у машинний код, а також має значну кількість сторонніх бібліотек, що спрощують процес розробки. До недоліків мови відноситься відсутність автоматичного керування пам’яттю, що є потенційно небезпечним, адже допускає існування помилок, що призводять до витоку пам’яті (явище, коли частина пам’яті, виділеної для виконання програми, не вивільняється після завершення її використання). однак, ручне керування пам’яттю дає розробнику більше контролю над роботою програми.

C# — мова програмування високого рівня, розроблена працівниками корпорації Microsoft Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде. Перша версія мови була випущена у 2002 році. Остання актуальна версія мови C# 7.0 вийшла 7 березня 2017 року. Мова програмування C# була

розроблена під впливом C++ та Java, має С-подібний синтаксис. Підтримує парадигму об’єктно-обієнтованого програмування. Має дещо нижчу швидкість роботи, ніж мова C++, проте вищу, ніж Java. Виділяється наявністю великої кількості сторонніх бібліотек, автоматичного регулювання пам’ятті та вбудованою реалізацією деяких шаблонів проектування.

Java — мова програмування високого рівня, створена компанією «Sun Microsystems» у 1995 році. Найактуальніша версія мови Java Standard Edition 10 випущена 20 березня 2018 року. Має Сі-подібний синтаксис. Підтримує парадигму об’єктно-орієнтованого програмування. Має суттєво гіршу швидкодію у порівнянні з мовами C++ та C#. Характеризується тим, що програми написані з використанням даної мови можуть виконуватися на найрізноманітніших пристроях Java використовує автоматичний збирач сміття для керування пам'яттю під час життєвого циклу об'єкта.

Python — інтерпретована ООП мова програмування високого рівня з динамічною типізацією. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. В даній мові програмування підтримується кілька парадигм програмування, зокрема:

— об'єктно-орієнтована;

— процедурна;

— функціональна;

— аспектно-орієнтована.

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Те, що це інтерпретована мова з динамічною обробкою типів, робить її ідеальною для написання скриптів та швидкої розробки прикладних програм, однак прикладні

програми будуть повільніше працювати, ніж у випадку якби вони були написані на компільованій мові програмування такій як C++ чи C#.

У результаті проведеного огляду для розробки було обрано мову C++ на якій будуть реалізовані всі елементи ПЗ. Це обгрунтовано потужністю цієї мови програмування, та зручністю роботи з використанням ООП підходу.

3.2 Вибір середовища розробки

Для більш зручної розробки програмних засобів використовуються IDE (інтегровані середовища розробки). Інтегроване середовище розробки — це комплекс програмних засобів для розробки, що складається з текстового редактора коду, інструментів для компіляції та відлагодження проектів.

Для порівняння середовищ розробки та подальшого вибору буде розглянуто середовища:

— JetBrains CLion;

— Microsoft Visual Studio;

— Visual Studio Code.

Visual Studio Code — текстовий редактор призначений для розробки застосунків для хмарних систем та веб-додатків. Visual Studio Code розповсюджується безкоштовно і є кросплатформеним, тобто його можна запустити на всіх сучасних операційних системах.

VS Code підтримує різноманітні плагіни, які з легістю можна інтегрувати в нього. Однак дане середовище більше підходить для написання коду інтерпретованими мовами програмування, тому з VS Code буде не дуже зручно працювати.

Microsoft Visual Studio — інтегроване середовище розробки, що розробляється компанією Microsoft. Підтримує такі мови програмування як C, C++, C#, F#, VisualBasic та інші. Дане середовище запускається лише під управлінням операційної системи «Windows».

CLion — це середовище розробки для написання коду на мовах програмування C та C++, що розробляється компанією JetBrains. Також є кросплатформеним та підтримує всі сучасні операційні системи.

Було обрано середовище Microsoft Visual Studio через свою доступність, підтримку всього необхідного для виконання.

3.3 Розробка програмних модулів системи

Для ефективної розробки додатку процесс було розділено на декілька етапів. З самого початку було спроектовано графічний головного вікна додатку, розподілено та розставлено кнопки для інтерфейсу. Було створено такі вікна:

— вікно із назвою «Програма перекладу», в ньому ввівши правильно пароль можна зайти у панель адміністратора або натиснувши кнопку «Старт» розпочати тестування;

— вікно із адміністраторською частиною, яке дозволяє додавати нові слова для перекладу та виставлення налаштувань на переклад цих слів;

— вікно для вивведення статистики проходження тестування;

— вікно для проходження тестування;

— вікно інформаіції про програму.

Вікно із назвою «Програма перекладу» має вигляд який вказано на рис.3.1.

Для створення методу входу в панель адміністратора було реалізовано наступний метод, що предствілений на рис.3.2.

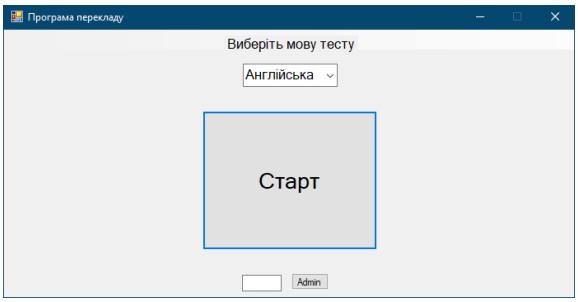

Рисунок 3.1 — Головне вікно програми до виконання

```
252    private: System::Void button6_Click(System::Object^  sender, System::EventArgs^  e) {
253                 if (this->f && (textBox1->Text == gcnew String(this->f->Translator->pass.c_str()))) {
254                     this->fa = gcnew Admin();
255                     this->fa->Show();
256                 }
257                 else  MessageBox::Show("Неправильний пароль!", "Info", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
258     }
```

Рисунок 3.2 — Метод для входу в панель адміністратора

Також, для запуску вікна тестування знань було розроблено метод, що представлений на рис. 3.3.

```
245    private: System::Void button5_Click(System::Object^  sender, System::EventArgs^  e) {
246
247                 this->f = gcnew TranslateForm;
248                 this->f->Start(comboBox1->SelectedIndex);
249                 this->f->Show();
250     }
```

Рисунок 3.3 — Метод для відкривання форми тестування знань

Вікно для проведення тестування показано на рис. 3.4.

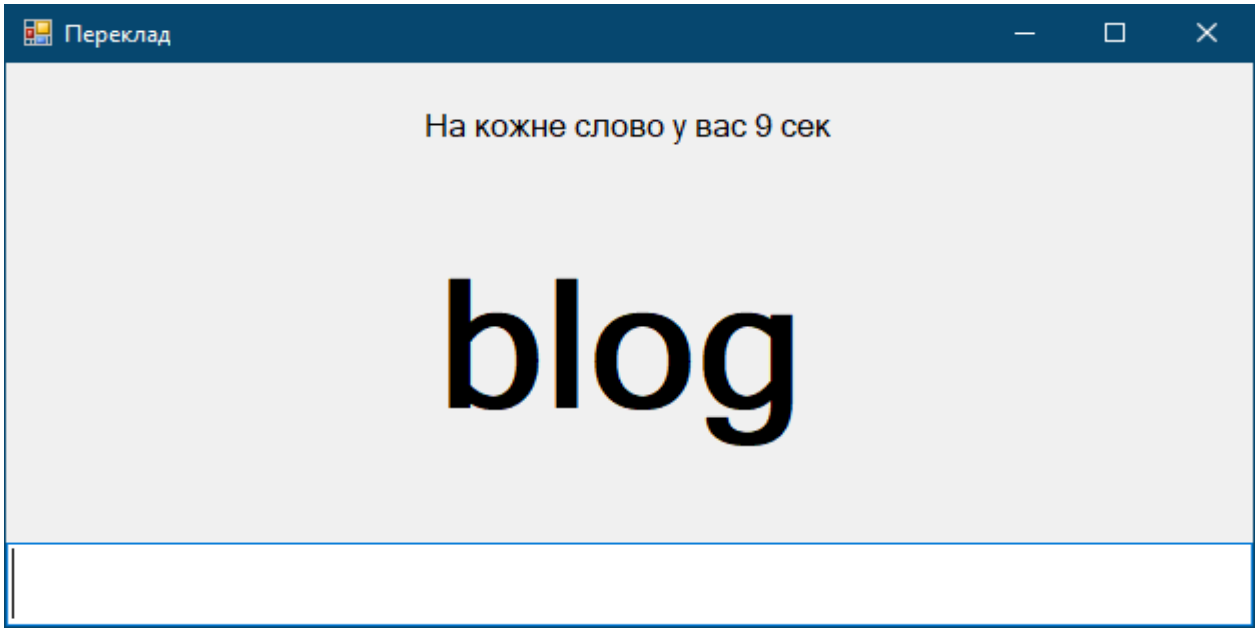

Рисунок 3.4 — Вікно для тестування знань

Як можна побачити із рис. 3.4 користувачу виводиться повідомлення про те, скільки часу йому залишилося до кінця тестування. Для контролю часу було використано подію таймера «timer1_Tick», код якої показано на рис. 3.5.

```
171     private: System::Void timer1_Tick(System::Object^  sender, System::EventArgs^  e) {
172                 if (!Translator->finished())
173                 {
174
175                     time_t ctime, timediff;
176                     time(&ctime);
177                     timediff = ctime - Translator->starttime;
178                     if (timediff > Translator->params[1]) {
179                         Translator->CheckTranslation(textBox1->Text);
180                         NextWord();
181                     } else
182                         this->label2->Text = "На кожне слово у вас " + (Translator->params[1] - timediff) + " сек";
183                 }
184     }
```

Рисунок 3.5 — Код реалізації події таймера «timer1_Tick»

Вікно для редагування інформації про тести представлено на рис. 3.6.

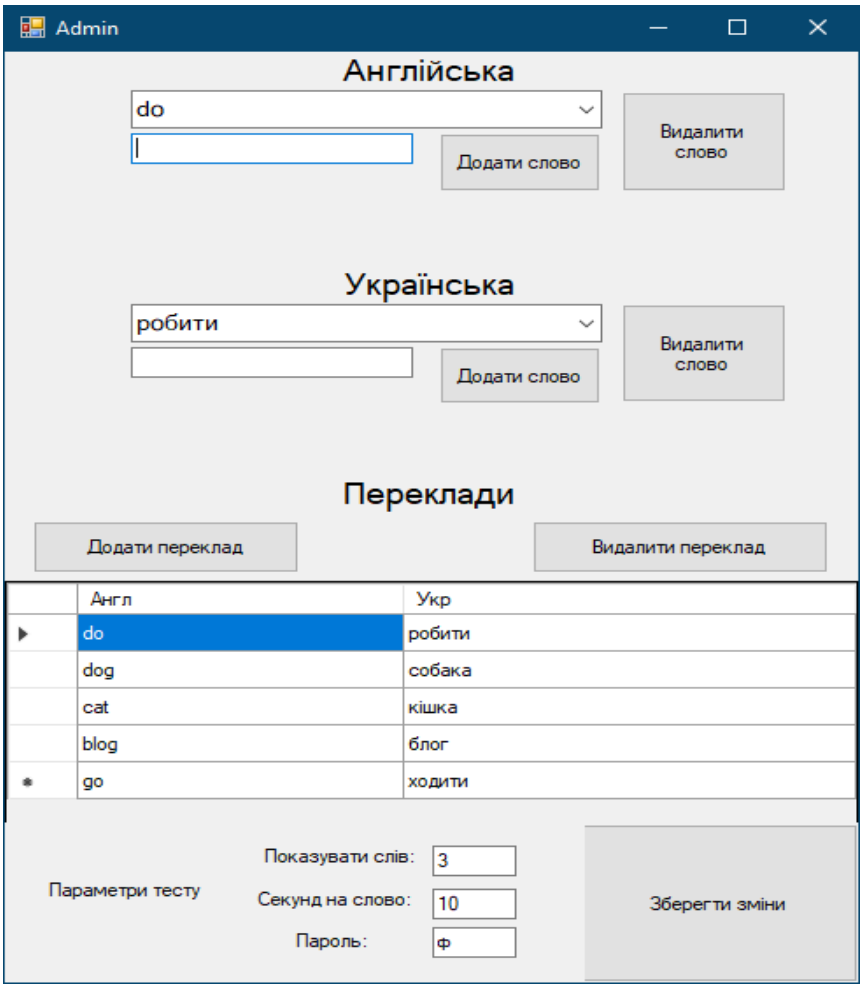

Рисунок 3.6 — Вікно керування тестами

Для функціоналу цього вікна було реалізовано цілий ряд методів. Серед них метод для додавання слів, який представлений на рис. 3.7.

```
504     private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
505                  if ((textBox1->Text != "")) {
506                      Translator->AddWord(0, textBox1->Text);
507                      textBox1->Text = "";
508                      Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2, textBox4, textBox5);
509                  }
510     }
```

Рисунок 3.7 — Метод для додавання нового слова у словник

Код методу для видалення слова із словника показаний на рис. 3.8.

```
void FillTables(System::Windows::Forms::ComboBox^ Combo3, System::Windows::Forms::ComboBox^ Combo4, System::Windows::Forms::DataGridView^ Grid3,
    System::Windows::Forms::TextBox^ Text1, System::Windows::Forms::TextBox^ Text2, System::Windows::Forms::TextBox^ Text3){

    Text1->Text = params[1].ToString();
    Text2->Text = gcnew String(pass.c_str());
    Text3->Text = params[3].ToString();

    Combo3->Items->Clear();
    for (int i = 0; i < words[0].size(); i++)
    {
        Combo3->Items->Add(gcnew String(words[0][i].c_str()));
        Combo3->SelectedIndex = 0;
    }

    Combo4->Items->Clear();
    for (int i = 0; i < words[1].size(); i++)
    {
        Combo4->Items->Add(gcnew String(words[1][i].c_str()));
        Combo4->SelectedIndex = 0;
    }

    Grid3->Rows->Clear();
    for (int i = 0; i < links.size(); i++)
    {
        if (i < links.size() - 1)
        Grid3->Rows->Add();
        Grid3->Rows[i]->Cells[0]->Value = gcnew String(words[0][links[i].l[0]].c_str());
        Grid3->Rows[i]->Cells[1]->Value = gcnew String(words[1][links[i].l[1]].c_str());
    }
}
```

Рисунок 3.8 — Метод для видалення слова із словника

Метод, який викликається при натисненні кнопки для зберігання налаштувань представлений на рис. 3.9.

```
531     private: System::Void button3_Click(System::Object^  sender, System::EventArgs^  e) {
532                  //Translator->params[0] = comboBox2->SelectedIndex;
533                  Translator->params[1] = Int32::Parse(textBox2->Text);
534                  Translator->params[2] = Int32::Parse(textBox5->Text);
535                  Translator->SetPass(textBox4->Text);
536                  Translator->SaveFiles();
537     }
```

Рисунок 3.9 — Метод для зберігання налаштувань

3.4 Тестування програмного модуля

Тестування — це процес аналізу та дослідження, який надає змогу виявити інформацію про якість продукту та покращити її шляхом виправлення помилок та неточностей. Методика тестування також включає в себе пошук дефектів, помилок, несправностей. Також це є перевірка програмних складових з метою оцінити готовність програмного продукту до використання. Результат тестування оцінюється за наступними критеріями:

— відповідність вимогам, які надавалися розробниками та проектувальниками;

— відповідність вихідних даних;

— прийнятний час виконання функцій;

— практичність;

— відповідність вимогам замовника.

Кількість тестів навіть для простих програмних компонентів може бути рівна кільком тисячам та більше, тому тактика тестування полягає в тому, що будуть проведені тільки необхідні тести з урахуванням доступного часу та ресурсів. Як результат, програмні засоби тестуються ручним виконанням програми з метою виявлення багів, помилок або інших дефектів.

Існує багато видів тестування: одні зазвичай виконують самі розробники, а інші — спеціалізовані групи. В нашому ж випадку буде проведено ручне тестування.

Тестування системи — це виконання програмного забезпечення в його остаточній конфігурації, інтегрованого з іншими програмними та апаратними системами з метою виявлення недоліків та помилок.

Є дві основних методики тестування: тестування «білої скриньки» та тестування «чорної скриньки».

У випадку методики «білої скриньки», об'єктом тестування є не зовнішня, а внутрішня поведінка програми. Перевіряється коректність побудови всіх

елементів програми та правильність їхньої взаємодії один з одним. Зазвичай аналізуються керуючі зв'язки елементів, рідше—інформаційні зв'язки. Тестування за принципом «білої скриньки» характеризується ступенем, в якому тести виконують або покривають логіку (вихідний текст) програми.

Одним із способів вивчення поставленого питання є дослідження методики «чорної скриньки», основне місце програми тестів «чорної скриньки» це інтерфейс ПЗ. Ці тести демонструють:

— як виконуються функції програми;

— як приймаються вихідні дані;

— як виробляються результати;

— як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе. Наприклад, якщо в програмі 10 вхідних величин і кожна приймає по 10 значень, то кількість тестових варіантів становитиме 1010. Тестування «чорної скриньки» не реагує на багато особливостей програмних помилок.

QA (Quality Assurance) — забезпечення якості продукту. QA-спеціаліст контролює та забезпечує якість роботи продукту компанії. Він відповідає за окремі етапи розробки софту. Зокрема, за вибір інструментів для розробки, запобігання можливим проблемам. Ще він бере участь у процесі вдосконалення товару. QA охоплює всі етапи розробки, включаючи опис проекту, власне, тестування, реліз і часто пост-релізний етап.

QC (Quality Control) — контроль якості продукту. Завдання QC-фахівця – перевірка конкретного продукту, що включає аналіз коду продукту, дизайну плюс тестування. QC-інженер розробляє стратегію тестування цілком певного тестування, взаємодіє з розробниками та організує саме тестування.

Фахівець із тестування займається виконанням тестів. Тестуванням називають перевірку відповідності результатів роботи програмного продукту на відповідність заданим критеріям. Тестувальники займаються тестуванням всього продукту в цілому або окремих компонентів. Тестування грає найважливішу роль забезпечення якості продукту.

До речі, є зовнішнє відгалуження – сучасний напрямок тестування Developer in test. Фахівці цього напряму — начебто і розробники, але займаються забезпеченням якості продукту, що розробляється.

Чомусь все більш поширеною стає хибна думка, згідно з якою тестувальники займаються тим, що просто натискають на кнопки і вводять рандомну інформацію в різні поля програми. Насправді це не так, якби тестувальники хаотично натискали на кнопки і вводили випадкові дані, результати тестування ніякої цінності для розробника не принесли б. Результати являли собою неструктуровану інформацію з якої неможливо отримати уявлення про те, наскільки якісним вийшов продукт і наскільки зручний він для користувачів. У тестувальників завжди є стратегія роботи, план, який дає змогу отримати об'єктивний опис актуального стану продукту.

Другий міф полягає у твердженні, що тестувальники відповідальні за якість ПЗ. Насправді відповідальність за якість розробки продукту несе вся команда. Тестувальники допомагають покращувати якість розробки, а також виявляють проблеми на ранніх стадіях.

Третій міф – тестувальників дуже багато. Насправді добрих спеціалістів на ринку мало. Багато тих, хто викладає резюме із позначкою «тестувальник», не розуміючи суті тестування ПЗ.

Тестування програмних засобів буде доречно проводити використовуючи методику «чорної скриньки».

Вона базується на використанні шаблонів тестування, бо ж тест-кейсів. Це означає що буде створено декілько ситуацій у яких перевіряється працездатність додатку, коректності основних функцій.

Розроблені тест-кейси описано в таб. 3.1.

Таблиця 3.1 —Тест-кейси

Код тест-кейса	Опис тест-кейса		
	Хід тестування		Очікуваний результат
	Дата тестування	Результат	Примітка
001	Перевірка створення коректності створення дат		
	1. Запустити додаток 2. Зайти в адмін панель.		Коректне заходження в адмін панель
	24.04.2022	Пройдено	-
002	Створення подій у різні місяці		
	1. Запустити додаток. 2. Зайти в адмін пан. 3. Додати нове слово.		Додане слово появилося у словнику.
	24.04.2022	Пройдено	-
003	Перевірка роботи запису даних у файл		
	1. Запустити додаток. 2. Зайти в адмін пан. 3. Змінити налаштув. 4. Зберегти зміни.		Змінено налаштування програми та успішно збережено.
	24.04.2022	Пройдено	-
004	Перевірка можливості редагування та видалення подій		
	1. Запустити додаток. 2. Пройти тест. 3. Результат .		Тестування пройдено. В кінці тестування виведено результат і слова, на які не було дано правильної відповіді
	24.04.2022	Пройдено	-

3.5 Інструкція користувача

Для початку потрібно запустити додаток. Перед користувачем буде виведено вікно в якому він зможе пройти тестування або ввівши правильний пароль редагувати словники та здійснювати налаштування системи. Вибравши мову, по якій буде проходити тестування та натиснувши кнопку «Старт»

користувач починає проходити тестування. На кожне питання відведено відповідну кількість часу. Якщо користувач не встиг відповісти протягом цього часу, то програма виведе наступне запитання і так до того часу, поки не буде завершено всі запитання. В кінці програма виведе результати тестування. Ввівши пароль адміністратора відкриється вікно, в якому користувач зможе редагувати словники та налаштовувати систему.

ВИСНОВКИ

У ході виконання бакалаврської дипломної роботи було проведено дослідження щодо теоретичних основ створення клавіатурного тринажеру з вивчення англійської мови.

У першому розділі було проведено аналітичний огляд існуючих програм та сервісів тестування, клавіатурних тринажерів. Порівняння існуючих систем, виявлення їх основних переваг та недоліків.

У другому розділі роботи проведено розробку структури клавіатурного тринажеру та алгоритмів тестування знань.

У третьому розділі описано програмну реалізацію та тестування програми. Також приведений загальний опис розробленої системи. Результати тестування були успішними, всі виявлені помилки було виправлено. Доведено актуальність розроблльованого прошрамного продукту.

Клавіатурний тринажер дає можливість як вивчити правила написання іншомовних слів, та вивчити їх зміст в режимі реального часу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Pro C# 7: With .NET and .NET Core 8-th edition / Andrew Troelse, Philip Japikse. –2017.

2. Об'єктно-орієнтоване програмування [Електронний ресурс] Режим доступу: http://ruslan.rv.ua/python-essential/oop/oop_basis/

3. Об'єктно-орієнтований підхід до програмування [Електронний ресурс] Режим доступу: http://www.znannya.org/?view=csharp-oop

4. Офіційна документація .NET [Електронний ресурс] Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/

5. Мова програмування C# і платформа .NET Core [Електронний ресурс] Режим доступу: https://metanit.com/sharp/tutorial/1.1.php

6. What's new in .NET 5 [Електронний ресурс] Режим доступу: https://docs.microsoft.com/en-us/dotnet/core/dotnet-five

7. .NET Core/5+ vs. .NET Framework for server apps [Електронний ресурс] Режим доступу: https://docs.microsoft.com/en-us/dotnet/standard/choosing-core-framework-server

8. Представляем .NET 5 [Електронний ресурс] Режим доступу: https://temofeev.ru/info/articles/predstavlyaem-net-5/

ДОДАТОК А

Міністерство освіти та науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ ВНТУ
д.т.н., проф.
_____ О. Д. Азаров
«19» квітня 2022 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання бакалаврської дипломної роботи
«Програмне забезпечення для перекладу слів в середовищі Visual Studio 2019»
08–23.БДР.021.00.000 ПЗ

Науковий керівник к.т.н. доц. каф. ОТ
_____ Снігур А.В.
Студент групи 1КІ–20мсз
_____ Крупський А.О.

Вінниця 2022

1 Підстави для виконання бакалаврської дипломної роботи (БДР)наступні:

— актуальність розробки, яка полягає у необхідності вирішення проблеми можливості органазіції та оптимізації часу, які дозволять отримати додаток для вирішення проблеми, на основі якого буде можливість коректно організовувати та оптимізовувати час та робочий процес;

— наказ про затвердження теми бакалаврської дипломної роботи.

2 Мета і призначення БДР наступні:

— метою БДР є розробка програмного забезпечення організації та оптимізації часу і виробничого процесу на .NET 5.0 в середовищі Visual Studio 2019, яке дозволить вводити події та отримати увесь розпорядок користувача;

— призначення розробки полягає у виконанні бакалаврської дипломноїроботи із подальшим впровадженням та розвитком.

3 Вихідні дані для виконання БДР наступні:

— технічний опис програмного застосунку;

— мова програмування C#;

— віконний додаток;

— середовище розробки Microsoft Visual Studio.

4 Вимога до виконання БДР наступна:

— створення віконного додатку;

— можливість створювати та редагувати події.

5 Етапи БДР та очікувані результати виконуються за вісім етапів, таблиця А.1.

6 Матеріали, що подаються до захисту БДР, наступні: пояснювальна записка БДР, графічні і ілюстративні матеріали, протокол попереднього захисту БДР на кафедрі, відгук наукового керівника, відгук опонента, протоколи складання державних екзаменів, анотації до БДР українською та іноземною мовами, нормоконтроль про відповідність оформлення БДР діючим вимогам.

7 Виконання етапів графічної та розрахункової документації БДР контролюється науковим керівником згідно зі встановленими термінами. Захист

БДР відбувається на засіданні Державної екзаменаційної комісії, затвердженою наказом ректора.

Таблиця А.1 — Етапи виконання роботи

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Постановка задачі роботи	09.03.22	09.03.22	Вступ
2	Пошук матеріалів по технологіям розробки віконних додатків	10.03.22	25.03.22	Розділ 1
3	Структурне проектування додатку тестування знань	26.03.22	28.03.22	Розділ 2
4	Обгрунтування та вибір засобів реалізації системи	29.03.22	04.04.22	Розділ 2
5	Розробка головного вікна	05.04.22	29.04.22	Розділ 3
6	Підготовка матеріалів та розробка алгоритму проведення тестування	30.04.22	27.05.22	Розділ 3, Працююча система
7	Оформлення пояснювальної записки та ілюстративного матеріалу	28.05.22	06.06.22	Пояснювальна записка
8	Перевірка якості виконання бакалаврської роботи та усунення недоліків	07.06.22	07.06.22	Презентація

8 Вимоги до оформлення БДР викладені в:

— ДСТУ 3008 : 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302 : 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— ГОСТ 2.104-2006 «Единa система конструкторської документації. Основні написи»;

— документи на які посилаються у вище вказаних.

Технічне завдання до виконання отримав _____ Крупський А.О.

ДОДАТОК Б

Код проекту додатку

Файл Admin.h

```
#pragma once

namespace TaskManager {

	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;

	/// <summary>
	/// Сводка для Admin
	/// </summary>
	public ref class Admin : public System::Windows::Forms::Form
	{
	public:
Admin(void)
{
	InitializeComponent();
	//
	//TODO: добавьте код конструктора
	//
	TranslatorC *tr = new TranslatorC();
	this->Translator = tr;
	if (!tr->LoadFiles()) {
this->label1->Text = "Помилка доступу до файлів";
	}
	else {
tr->FillTables(comboBox3, comboBox4, dataGridView3, textBox2, textBox4,
textBox5);
	}
}

	protected:
/// <summary>
/// Освободить все используемые ресурсы.
```

```
/// </summary>
~Admin()
{
	if (components)
	{
delete components;
	}
}

	private: System::Windows::Forms::Label^  label1;
	private: System::Windows::Forms::Label^  label2;
	private: System::Windows::Forms::Label^  label3;
	private: System::Windows::Forms::DataGridView^  dataGridView3;
	private: System::Windows::Forms::DataGridViewTextBoxColumn^
dataGridViewTextBoxColumn3;
	private: System::Windows::Forms::DataGridViewTextBoxColumn^
dataGridViewTextBoxColumn4;
	private: System::Windows::Forms::TextBox^  textBox1;
	private: System::Windows::Forms::Button^  button1;
	private: System::Windows::Forms::Button^  button2;
	protected:
	public: TranslatorC *Translator;
	private: System::Windows::Forms::Button^  button3;
	public:
	private: System::Windows::Forms::Button^  button4;
	private: System::Windows::Forms::Button^  button5;
	private: System::Windows::Forms::Panel^  panel1;
	private: System::Windows::Forms::Label^  label4;
	private: System::Windows::Forms::TextBox^  textBox2;

	private: System::Windows::Forms::Label^  label5;
	private: System::Windows::Forms::Label^  label6;

	private: System::Windows::Forms::ComboBox^  comboBox3;
	private: System::Windows::Forms::Panel^  panel2;
	private: System::Windows::Forms::Panel^  panel3;
	private: System::Windows::Forms::Button^  button7;
	private: System::Windows::Forms::ComboBox^  comboBox4;
	private: System::Windows::Forms::TextBox^  textBox3;
	private: System::Windows::Forms::Button^  button8;
	private: System::Windows::Forms::Panel^  panel4;
	private: System::Windows::Forms::Panel^  panel5;
	private: System::Windows::Forms::Panel^  panel6;
```

```
        private: System::Windows::Forms::Panel^  panel7;
        private: System::Windows::Forms::Label^  label7;
        private: System::Windows::Forms::TextBox^  textBox5;
        private: System::Windows::Forms::TextBox^  textBox4;



        private:
/// <summary>
/// Требуется переменная конструктора.
/// </summary>
System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
/// <summary>
/// Обязательный метод для поддержки конструктора - не изменяйте
/// содержимое данного метода при помощи редактора кода.
/// </summary>
void InitializeComponent(void)
{
        this->label1 = (gcnew System::Windows::Forms::Label());
        this->label2 = (gcnew System::Windows::Forms::Label());
        this->label3 = (gcnew System::Windows::Forms::Label());
        this->dataGridView3 = (gcnew System::Windows::Forms::DataGridView());
        this->dataGridViewTextBoxColumn3 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->dataGridViewTextBoxColumn4 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->textBox1 = (gcnew System::Windows::Forms::TextBox());
        this->button1 = (gcnew System::Windows::Forms::Button());
        this->button2 = (gcnew System::Windows::Forms::Button());
        this->button3 = (gcnew System::Windows::Forms::Button());
        this->button4 = (gcnew System::Windows::Forms::Button());
        this->button5 = (gcnew System::Windows::Forms::Button());
        this->panel1 = (gcnew System::Windows::Forms::Panel());
        this->label7 = (gcnew System::Windows::Forms::Label());
        this->textBox5 = (gcnew System::Windows::Forms::TextBox());
        this->textBox4 = (gcnew System::Windows::Forms::TextBox());
        this->label6 = (gcnew System::Windows::Forms::Label());
        this->label5 = (gcnew System::Windows::Forms::Label());
        this->label4 = (gcnew System::Windows::Forms::Label());
        this->textBox2 = (gcnew System::Windows::Forms::TextBox());
        this->comboBox3 = (gcnew System::Windows::Forms::ComboBox());
```

```
        this->panel2 = (gcnew System::Windows::Forms::Panel());
        this->panel3 = (gcnew System::Windows::Forms::Panel());
        this->button7 = (gcnew System::Windows::Forms::Button());
        this->comboBox4 = (gcnew System::Windows::Forms::ComboBox());
        this->textBox3 = (gcnew System::Windows::Forms::TextBox());
        this->button8 = (gcnew System::Windows::Forms::Button());
        this->panel4 = (gcnew System::Windows::Forms::Panel());
        this->panel5 = (gcnew System::Windows::Forms::Panel());
        this->panel6 = (gcnew System::Windows::Forms::Panel());
        this->panel7 = (gcnew System::Windows::Forms::Panel());
        (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this-
>dataGridView3))->BeginInit();
        this->panel1->SuspendLayout();
        this->panel2->SuspendLayout();
        this->panel3->SuspendLayout();
        this->panel4->SuspendLayout();
        this->panel5->SuspendLayout();
        this->panel6->SuspendLayout();
        this->panel7->SuspendLayout();
        this->SuspendLayout();
        //
        // label1
        //
        this->label1->Dock = System::Windows::Forms::DockStyle::Top;
        this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
13.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label1->Location = System::Drawing::Point(0, 0);
        this->label1->Name = L"label1";
        this->label1->Size = System::Drawing::Size(649, 29);
        this->label1->TabIndex = 2;
        this->label1->Text = L"Англійська";
        this->label1->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;
        //
        // label2
        //
        this->label2->Dock = System::Windows::Forms::DockStyle::Top;
        this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
13.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label2->Location = System::Drawing::Point(0, 0);
        this->label2->Name = L"label2";
        this->label2->Size = System::Drawing::Size(649, 29);
```

```
        this->label2->TabIndex = 4;
        this->label2->Text = L"Українська";
        this->label2->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;
        //
        // label3
        //
        this->label3->Dock = System::Windows::Forms::DockStyle::Top;
        this->label3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
13.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label3->Location = System::Drawing::Point(0, 0);
        this->label3->Name = L"label3";
        this->label3->Size = System::Drawing::Size(649, 36);
        this->label3->TabIndex = 6;
        this->label3->Text = L"Переклади";
        this->label3->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;
        //
        // dataGridView3
        //
        this->dataGridView3->Anchor =
System::Windows::Forms::AnchorStyles::None;
        this->dataGridView3->BackgroundColor =
System::Drawing::SystemColors::Control;
        this->dataGridView3->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
        this->dataGridView3->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^  >(2) {
this->dataGridViewTextBoxColumn3,
        this->dataGridViewTextBoxColumn4
        });
        this->dataGridView3->Location = System::Drawing::Point(0, 87);
        this->dataGridView3->Name = L"dataGridView3";
        this->dataGridView3->RowTemplate->Height = 24;
        this->dataGridView3->Size = System::Drawing::Size(649, 194);
        this->dataGridView3->TabIndex = 5;
        //
        // dataGridViewTextBoxColumn3
        //
        this->dataGridViewTextBoxColumn3->AutoSizeMode =
System::Windows::Forms::DataGridViewAutoSizeColumnMode::Fill;
        this->dataGridViewTextBoxColumn3->FillWeight = 46.49077F;
        this->dataGridViewTextBoxColumn3->HeaderText = L"Англ";
```

```
        this->dataGridViewTextBoxColumn3->Name =
L"dataGridViewTextBoxColumn3";
        this->dataGridViewTextBoxColumn3->ReadOnly = true;
        //
        // dataGridViewTextBoxColumn4
        //
        this->dataGridViewTextBoxColumn4->AutoSizeMode =
System::Windows::Forms::DataGridViewAutoSizeColumnMode::Fill;
        this->dataGridViewTextBoxColumn4->FillWeight = 64.33948F;
        this->dataGridViewTextBoxColumn4->HeaderText = L"Укр";
        this->dataGridViewTextBoxColumn4->Name =
L"dataGridViewTextBoxColumn4";
        this->dataGridViewTextBoxColumn4->ReadOnly = true;
        //
        // textBox1
        //
        this->textBox1->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->textBox1->CharacterCasing =
System::Windows::Forms::CharacterCasing::Lower;
        this->textBox1->Location = System::Drawing::Point(96, 65);
        this->textBox1->Name = L"textBox1";
        this->textBox1->Size = System::Drawing::Size(213, 22);
        this->textBox1->TabIndex = 8;
        //
        // button1
        //
        this->button1->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button1->Location = System::Drawing::Point(331, 65);
        this->button1->Name = L"button1";
        this->button1->Size = System::Drawing::Size(122, 47);
        this->button1->TabIndex = 9;
        this->button1->Text = L"Додати слово";
        this->button1->UseVisualStyleBackColor = true;
        this->button1->Click += gcnew System::EventHandler(this,
&Admin::button1_Click);
        //
        // button2
        //
        this->button2->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button2->Location = System::Drawing::Point(470, 32);
        this->button2->Name = L"button2";
        this->button2->Size = System::Drawing::Size(126, 80);
        this->button2->TabIndex = 10;
```

```
        this->button2->Text = L"Видалити слово";
        this->button2->UseVisualStyleBackColor = true;
        this->button2->Click += gcnew System::EventHandler(this,
&Admin::button2_Click);
        //
        // button3
        //
        this->button3->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button3->Location = System::Drawing::Point(439, 0);
        this->button3->Name = L"button3";
        this->button3->Size = System::Drawing::Size(210, 127);
        this->button3->TabIndex = 11;
        this->button3->Text = L"Зберегти зміни";
        this->button3->UseVisualStyleBackColor = true;
        this->button3->Click += gcnew System::EventHandler(this,
&Admin::button3_Click);
        //
        // button4
        //
        this->button4->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button4->Location = System::Drawing::Point(22, 39);
        this->button4->Name = L"button4";
        this->button4->Size = System::Drawing::Size(203, 42);
        this->button4->TabIndex = 12;
        this->button4->Text = L"Додати переклад";
        this->button4->UseVisualStyleBackColor = true;
        this->button4->Click += gcnew System::EventHandler(this,
&Admin::button4_Click);
        //
        // button5
        //
        this->button5->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button5->Location = System::Drawing::Point(401, 39);
        this->button5->Name = L"button5";
        this->button5->Size = System::Drawing::Size(225, 42);
        this->button5->TabIndex = 13;
        this->button5->Text = L"Видалити переклад";
        this->button5->UseVisualStyleBackColor = true;
        this->button5->Click += gcnew System::EventHandler(this,
&Admin::button5_Click);
        //
        // panel1
        //
```

```
        this->panel1->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->panel1->Controls->Add(this->label7);
        this->panel1->Controls->Add(this->textBox5);
        this->panel1->Controls->Add(this->textBox4);
        this->panel1->Controls->Add(this->label6);
        this->panel1->Controls->Add(this->label5);
        this->panel1->Controls->Add(this->label4);
        this->panel1->Controls->Add(this->textBox2);
        this->panel1->Location = System::Drawing::Point(0, 0);
        this->panel1->Name = L"panel1";
        this->panel1->Size = System::Drawing::Size(441, 127);
        this->panel1->TabIndex = 16;
        //
        // label7
        //
        this->label7->AutoSize = true;
        this->label7->Location = System::Drawing::Point(193, 17);
        this->label7->Name = L"label7";
        this->label7->Size = System::Drawing::Size(118, 17);
        this->label7->TabIndex = 23;
        this->label7->Text = L"Показувати слів:";
        //
        // textBox5
        //
        this->textBox5->Location = System::Drawing::Point(325, 17);
        this->textBox5->Name = L"textBox5";
        this->textBox5->Size = System::Drawing::Size(62, 22);
        this->textBox5->TabIndex = 22;
        //
        // textBox4
        //
        this->textBox4->CharacterCasing =
System::Windows::Forms::CharacterCasing::Lower;
        this->textBox4->Location = System::Drawing::Point(325, 82);
        this->textBox4->Name = L"textBox4";
        this->textBox4->Size = System::Drawing::Size(62, 22);
        this->textBox4->TabIndex = 21;
        //
        // label6
        //
        this->label6->AutoSize = true;
        this->label6->Location = System::Drawing::Point(30, 44);
        this->label6->Name = L"label6";
```

```
        this->label6->Size = System::Drawing::Size(122, 17);
        this->label6->TabIndex = 20;
        this->label6->Text = L"Параметри тесту";
        //
        // label5
        //
        this->label5->AutoSize = true;
        this->label5->Location = System::Drawing::Point(217, 85);
        this->label5->Name = L"label5";
        this->label5->Size = System::Drawing::Size(61, 17);
        this->label5->TabIndex = 19;
        this->label5->Text = L"Пароль:";
        //
        // label4
        //
        this->label4->AutoSize = true;
        this->label4->Location = System::Drawing::Point(190, 52);
        this->label4->Name = L"label4";
        this->label4->Size = System::Drawing::Size(121, 17);
        this->label4->TabIndex = 17;
        this->label4->Text = L"Секунд на слово:";
        //
        // textBox2
        //
        this->textBox2->Location = System::Drawing::Point(325, 52);
        this->textBox2->Name = L"textBox2";
        this->textBox2->Size = System::Drawing::Size(62, 22);
        this->textBox2->TabIndex = 16;
        //
        // comboBox3
        //
        this->comboBox3->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->comboBox3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
Serif", 10.2F, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->comboBox3->FormattingEnabled = true;
        this->comboBox3->Location = System::Drawing::Point(96, 31);
        this->comboBox3->Name = L"comboBox3";
        this->comboBox3->Size = System::Drawing::Size(357, 28);
        this->comboBox3->TabIndex = 18;
        //
        // panel2
```

```
        //
        this->panel2->Controls->Add(this->button1);
        this->panel2->Controls->Add(this->comboBox3);
        this->panel2->Controls->Add(this->label1);
        this->panel2->Controls->Add(this->textBox1);
        this->panel2->Controls->Add(this->button2);
        this->panel2->Dock = System::Windows::Forms::DockStyle::Top;
        this->panel2->Location = System::Drawing::Point(0, 0);
        this->panel2->Name = L"panel2";
        this->panel2->Size = System::Drawing::Size(649, 159);
        this->panel2->TabIndex = 19;
        //
        // panel3
        //
        this->panel3->Controls->Add(this->button7);
        this->panel3->Controls->Add(this->comboBox4);
        this->panel3->Controls->Add(this->textBox3);
        this->panel3->Controls->Add(this->button8);
        this->panel3->Controls->Add(this->label2);
        this->panel3->Dock = System::Windows::Forms::DockStyle::Bottom;
        this->panel3->Location = System::Drawing::Point(0, 171);
        this->panel3->Name = L"panel3";
        this->panel3->Size = System::Drawing::Size(649, 163);
        this->panel3->TabIndex = 20;
        //
        // button7
        //
        this->button7->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button7->Location = System::Drawing::Point(331, 65);
        this->button7->Name = L"button7";
        this->button7->Size = System::Drawing::Size(122, 45);
        this->button7->TabIndex = 9;
        this->button7->Text = L"Додати слово";
        this->button7->UseVisualStyleBackColor = true;
        this->button7->Click += gcnew System::EventHandler(this,
&Admin::button7_Click);
        //
        // comboBox4
        //
        this->comboBox4->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->comboBox4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
Serif", 10.2F, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
```

```
static_cast<System::Byte>(204)));
        this->comboBox4->FormattingEnabled = true;
        this->comboBox4->Location = System::Drawing::Point(96, 31);
        this->comboBox4->Name = L"comboBox4";
        this->comboBox4->Size = System::Drawing::Size(357, 28);
        this->comboBox4->TabIndex = 18;
        //
        // textBox3
        //
        this->textBox3->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->textBox3->CharacterCasing =
System::Windows::Forms::CharacterCasing::Lower;
        this->textBox3->Location = System::Drawing::Point(96, 65);
        this->textBox3->Name = L"textBox3";
        this->textBox3->Size = System::Drawing::Size(213, 22);
        this->textBox3->TabIndex = 8;
        //
        // button8
        //
        this->button8->Anchor = System::Windows::Forms::AnchorStyles::None;
        this->button8->Location = System::Drawing::Point(470, 31);
        this->button8->Name = L"button8";
        this->button8->Size = System::Drawing::Size(126, 79);
        this->button8->TabIndex = 10;
        this->button8->Text = L"Видалити слово";
        this->button8->UseVisualStyleBackColor = true;
        this->button8->Click += gcnew System::EventHandler(this,
&Admin::button8_Click);
        //
        // panel4
        //
        this->panel4->Controls->Add(this->panel2);
        this->panel4->Controls->Add(this->panel3);
        this->panel4->Dock = System::Windows::Forms::DockStyle::Top;
        this->panel4->Location = System::Drawing::Point(0, 0);
        this->panel4->Name = L"panel4";
        this->panel4->Size = System::Drawing::Size(649, 334);
        this->panel4->TabIndex = 21;
        //
        // panel5
        //
        this->panel5->Controls->Add(this->dataGridView3);
        this->panel5->Controls->Add(this->label3);
```

```
        this->panel5->Controls->Add(this->button4);
        this->panel5->Controls->Add(this->button5);
        this->panel5->Dock = System::Windows::Forms::DockStyle::Top;
        this->panel5->Location = System::Drawing::Point(0, 0);
        this->panel5->Name = L"panel5";
        this->panel5->Size = System::Drawing::Size(649, 281);
        this->panel5->TabIndex = 22;
        //
        // panel6
        //
        this->panel6->Controls->Add(this->panel7);
        this->panel6->Controls->Add(this->panel5);
        this->panel6->Dock = System::Windows::Forms::DockStyle::Bottom;
        this->panel6->Location = System::Drawing::Point(0, 337);
        this->panel6->Name = L"panel6";
        this->panel6->Size = System::Drawing::Size(649, 408);
        this->panel6->TabIndex = 23;
        //
        // panel7
        //
        this->panel7->Controls->Add(this->panel1);
        this->panel7->Controls->Add(this->button3);
        this->panel7->Dock = System::Windows::Forms::DockStyle::Bottom;
        this->panel7->Location = System::Drawing::Point(0, 281);
        this->panel7->Name = L"panel7";
        this->panel7->Size = System::Drawing::Size(649, 127);
        this->panel7->TabIndex = 23;
        //
        // Admin
        //
        this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
        this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
        this->AutoScroll = true;
        this->ClientSize = System::Drawing::Size(649, 745);
        this->Controls->Add(this->panel6);
        this->Controls->Add(this->panel4);
        this->Name = L"Admin";
        this->Text = L"Admin";
        this->Load += gcnew System::EventHandler(this, &Admin::Admin_Load);
        (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this-
>dataGridView3))->EndInit();
        this->panel1->ResumeLayout(false);
        this->panel1->PerformLayout();
```

```
        this->panel2->ResumeLayout(false);
        this->panel2->PerformLayout();
        this->panel3->ResumeLayout(false);
        this->panel3->PerformLayout();
        this->panel4->ResumeLayout(false);
        this->panel5->ResumeLayout(false);
        this->panel6->ResumeLayout(false);
        this->panel7->ResumeLayout(false);
        this->ResumeLayout(false);

}
#pragma endregion
```

ДОДАТОК В

КОД ПОЧАТКУ РОБОТИ ПРОГРАМИ

```
        private: System::Void Admin_Load(System::Object^  sender,
System::EventArgs^  e) {
        }

private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e)
{
 if ((textBox1->Text != "")) {
          Translator->AddWord(0, textBox1->Text);
          textBox1->Text = "";
          Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2,
textBox4, textBox5);
 }
        }

        private: System::Void button4_Click(System::Object^  sender,
System::EventArgs^  e) {
 Translator->AddTranslation(comboBox3, comboBox4);

 Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2,
textBox4, textBox5);

        }

        private: System::Void button2_Click(System::Object^  sender,
System::EventArgs^  e) {
 Translator->DeleteWord(0, comboBox3->SelectedIndex);
 Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2,
textBox4, textBox5);

        }
        private: System::Void button6_Click(System::Object^  sender,
System::EventArgs^  e) {

        }
Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2, textBox4,
textBox5);
        }
```

ДОДАТОК Г

ФРАГМЕНТ КОДУ ЗБЕРЕЖЕННЯ ДАННИХ КОРИСТУВАЧА

```
        private: System::Void button3_Click(System::Object^  sender,
System::EventArgs^  e) {
 //Translator->params[0] = comboBox2->SelectedIndex;
 Translator->params[1] = Int32::Parse(textBox2->Text);
 Translator->params[2] = Int32::Parse(textBox5->Text);
 Translator->SetPass(textBox4->Text);
 Translator->SaveFiles();
        }
        private: System::Void button7_Click(System::Object^  sender,
System::EventArgs^  e) {
 if ((textBox3->Text != "")) {
         Translator->AddWord(1, textBox3->Text);
         textBox3->Text = "";

         Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2,
textBox4, textBox5);
 }
        }
        private: System::Void button8_Click(System::Object^  sender,
System::EventArgs^  e) {
 Translator->DeleteWord(1, comboBox4->SelectedIndex);
 Translator->FillTables(comboBox3, comboBox4, dataGridView3, textBox2,
textBox4, textBox5);
        };
        }
        ; }
```

Файл MyC.h

```
#include <fstream>
#include <vector>
#include <time.h>
#include "stdafx.h"

using namespace System;
using namespace std;
using System::IntPtr;
using System::Runtime::InteropServices::Marshal;

/// <summary>
```

ДОДАТОК Д

КОД ПРОГРАМИ ДЛЯ СТАРТУ РОБОТИ

```
/// Сводка для TranslateForm
/// </summary>
public class LinkC{
public:unsigned int l[2];
public:
	LinkC(unsigned int x, unsigned int y) {
l[0] = x;
l[1] = y;
	}
};

public class TranslatorC {
public: vector<string> words[2];
vector<unsigned int> params;
vector<unsigned int> wrong;
vector<LinkC> links;
bool empty = 1;
unsigned int lang = 0;
unsigned int wordpos = 0;
unsigned int correct = 0;
unsigned int testedwords = 0;
string pass;
time_t starttime = 0, teststart = 0, testend = 0;

String ^getCurrentWord(){
	time(&starttime);
	if (teststart == 0)
time(&teststart);
	srand((unsigned)time(NULL));

	wordpos = rand() % (words[lang].size());
	return  gcnew String(words[lang][wordpos].c_str());
}

void ShowDetails(System::Windows::Forms::DataGridView^ Grid,
System::Windows::Forms::Label^ Label){
	Label->Text = "Часу витрачено:  "+(testend - teststart).ToString()+ "сек \n
Невірні слова:";
```

```
        for (int i = 0; i < wrong.size(); i++)
        {
if (i < wrong.size() - 1)
        Grid->Rows->Add();
Grid->Rows[i]->Cells[lang]->Value = gcnew String(words[lang][wrong[i]].c_str());
int i1 = 0;
while (i1 < links.size()) {

        if (links[i1].l[lang] == wrong[i]) {
Grid->Rows[i]->Cells[1 - lang]->Value = gcnew String(words[1 - lang][links[i1].l[1 -
lang]].c_str());
break;
        }
        i1++;
}
//Grid->Rows[i]->Cells[1]->Value = gcnew String(words[1][links[i].l[1]].c_str());
        }
}

bool finished(){

        return (testedwords>=params[3]);
}

bool CheckTranslation(String^ s)
{
        if (!finished() && (s!="")) {
int i = 0;
while (i < links.size()) {

        string ss = words[1 - lang][links[i].l[1 - lang]];
        if ((links[i].l[lang] == wordpos) && (gcnew String(words[1 - lang][links[i].l[1 -
lang]].c_str()) == s)) {
correct++;
return true;
        }
        i++;
}
        }
        wrong.push_back(wordpos);
        return false;

}
```

ДОДАТОК Е

ФРАГМЕНТ КОДУ C++

```
bool AddWord(int index, String^ s){
	const char* str = (const char*)(void*)
Marshal::StringToHGlobalAnsi(s);
	words[index].push_back(str);
	return true;
}

void SetPass(String^ s){
	const char* str = (const char*)(void*)
Marshal::StringToHGlobalAnsi(s);
	pass = str;

}

bool DeleteWord(int index, int selected){

	words[index].erase(words[index].begin() + selected);

	for (int i = 0; i < links.size(); i++)
	{
if (links[i].l[index] == selected){
	links.erase(links.begin() + i);
	i--;
}
else if (links[i].l[index]>selected)
	links[i].l[index]--;
	}
	return true;
}

bool DeleteTranslation(System::Windows::Forms::DataGridView^ Grid){
	int row = Grid->CurrentCellAddress.Y;
	if (row < 0) row = 0;
	links.erase(links.begin() + row);
	return true;
}
```

```
bool AddTranslation(System::Windows::Forms::ComboBox^ Combo3,
System::Windows::Forms::ComboBox^ Combo4){
	int row1 = Combo3->SelectedIndex;
	int row2 = Combo4->SelectedIndex;
	if (row1>=0 && row2>=0)
links.push_back(LinkC(row1,row2));
	return true;
}

bool ChangeWord(int index, System::Windows::Forms::DataGridView^ Grid){
	int row = Grid->CurrentCellAddress.Y;
	if (row < 0) row = 0;
	const char* str = (const char*)(void*)
Marshal::StringToHGlobalAnsi(Grid->Rows[row]->Cells[1]->Value->ToString());
	words[index][row] = str;
	return true;
}

void SetLang(int i) {
	lang = i;
	params[0] = i;

}

void FillTables(System::Windows::Forms::ComboBox^ Combo3,
System::Windows::Forms::ComboBox^ Combo4,
System::Windows::Forms::DataGridView^ Grid3,
	System::Windows::Forms::TextBox^ Text1,
System::Windows::Forms::TextBox^ Text2, System::Windows::Forms::TextBox^
Text3){

	Text1->Text = params[1].ToString();
	Text2->Text = gcnew String(pass.c_str());
	Text3->Text = params[3].ToString();

	Combo3->Items->Clear();
	for (int i = 0; i < words[0].size(); i++)
	{
Combo3->Items->Add(gcnew String(words[0][i].c_str()));
Combo3->SelectedIndex = 0;
	}

	Combo4->Items->Clear();
```

```
	for (int i = 0; i < words[1].size(); i++)
	{
Combo4->Items->Add(gcnew String(words[1][i].c_str()));
Combo4->SelectedIndex = 0;
	}

	Grid3->Rows->Clear();
	for (int i = 0; i < links.size(); i++)
	{
if (i < links.size() - 1)
Grid3->Rows->Add();
Grid3->Rows[i]->Cells[0]->Value = gcnew String(words[0][links[i].l[0]].c_str());
Grid3->Rows[i]->Cells[1]->Value = gcnew String(words[1][links[i].l[1]].c_str());
	}
}

bool LoadFiles(){
	TCHAR buff[MAX_PATH];
	memset(buff, 0, MAX_PATH);
	::GetModuleFileName(NULL, buff, sizeof(buff));
	String ^strFolder = gcnew String(buff);
	strFolder = strFolder->Substring(0, strFolder->LastIndexOf(('\\')) + 1);

	const char* str1 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "eng.txt");
	const char* str2 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "ukr.txt");
	const char* str3 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "param.txt");
	const char* strl = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "links.txt");
	const char* strp = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "p.txt");
	// use str here for the ofstream filename

	std::ifstream in(str1);

	if (in.is_open())
	{
char name[256];
while (in.getline(name, 256))
{
	words[0].push_back(name);
```

```
}
in.close();
	}
	else return 0;

	std::ifstream in1(str2);

	if (in1.is_open())
	{
char name[256];
while (in1.getline(name, 256))
{
	words[1].push_back(name);

}
in1.close();
	}
	else return 0;

	std::ifstream in2(str3);

	if (in2.is_open())
	{
int par, i=0;
while ((in2 >> par) && (i<4))
{
	params.push_back(par);
	i++;
}

lang = params[0];
in2.close();
	}
	else return 0;

	std::ifstream in3(strl);

	if (in3.is_open())
	{
int l1, l2;
while (in3 >> l1 >> l2)
{
```

```
		links.push_back(LinkC(l1, l2));
}
in3.close();
		}
		else return 0;

		std::ifstream in4(strp);

		if (in4.is_open())
		{
char n[50];
in4.getline(n, 50);
pass = n;
in4.close();
		}
		else return 0;
		empty = 0;
		return 1;
}

bool SaveFiles(){
		TCHAR buff[MAX_PATH];
		memset(buff, 0, MAX_PATH);
		::GetModuleFileName(NULL, buff, sizeof(buff));
		String ^strFolder = gcnew String(buff);
		strFolder = strFolder->Substring(0, strFolder->LastIndexOf(('\\')) + 1);

		const char* str1 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "eng.txt");
		const char* str2 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "ukr.txt");
		const char* str3 = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "param.txt");
		const char* strl = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "links.txt");
		// use str here for the ofstream filename
		const char* strp = (const char*)(void*)
Marshal::StringToHGlobalAnsi(strFolder + "p.txt");

		std::ofstream out(str1);

		if (out.is_open())
		{
```

```
for (int i = 0; i < words[0].size(); i++)
{
	out << words[0][i].c_str() << "\n";
}
out.close();
	}
	else return 0;

	std::ofstream out1(str2);

	if (out1.is_open())
	{
for (int i = 0; i < words[1].size(); i++)
{
	out1 << words[1][i].c_str() << "\n";
}
out1.close();
	}
	else return 0;

	std::ofstream out2(str3);

	if (out2.is_open())
	{
for (int i = 0; i < params.size(); i++)
{
	out2 << params[i] << "\n";
}
out2 << pass.c_str();
out2.close();
	}
	else return 0;

	std::ofstream out3(strl);

	if (out3.is_open())
	{
for (int i = 0; i < links.size(); i++)
{
	out3 << links[i].l[0] << " " << links[i].l[1] << "\n";
}
out3.close();
	}
```

```
        else return 0;

        std::ofstream out4(strp);

        if (out4.is_open())
        {
out4 << pass.c_str();

out4.close();
        }
        else return 0;
        empty = 0;
        return 1;
}
};
```

ДОДАТОК Ж

ФРАГМЕНТ КОДУ ДЛЯ ПЕРЕКЛАДУ СЛІВ

Файл TranslateForm.h

```
#pragma once

#include "stdafx.h"

namespace TaskManager {

	using namespace System;
	using namespace std;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;

	public ref class TranslateForm : public System::Windows::Forms::Form
	{
	public:

TranslateForm(void)
{
	InitializeComponent();
	TranslatorC *tr = new TranslatorC();
	this->Translator = tr;
	if (!tr->LoadFiles()) {
this->label1->Text = "Помилка доступу до файлів";
	}
	else {
	}
}

void Start(int lang) {
	Translator->SetLang(lang);
	this->label1->Text = Translator->getCurrentWord();
	this->label2->Text = "На кожне слово у вас " + Translator-
>params[1].ToString() + " сек";
	timer1->Enabled = 1;
}
```

```
        protected:
/// <summary>
/// Освободить все используемые ресурсы.
/// </summary>
~TranslateForm()
{
        if (components)
        {
delete components;
        }
}
        private: System::Windows::Forms::Panel^  panel1;
        protected:
        private: System::Windows::Forms::TextBox^  textBox1;
        private: System::Windows::Forms::Label^  label1;
        private: System::Windows::Forms::Timer^  timer1;
        private: System::ComponentModel::IContainer^  components;
        private: System::Windows::Forms::Label^  label2;

        public: TranslatorC *Translator;
        private: MyForm^ f;
/// <summary>
/// Требуется переменная конструктора.
/// </summary>


#pragma region Windows Form Designer generated code
/// <summary>
/// Обязательный метод для поддержки конструктора - не изменяйте
/// содержимое данного метода при помощи редактора кода.
/// </summary>
void InitializeComponent(void)
{
        this->components = (gcnew System::ComponentModel::Container());
        this->panel1 = (gcnew System::Windows::Forms::Panel());
        this->label2 = (gcnew System::Windows::Forms::Label());
        this->textBox1 = (gcnew System::Windows::Forms::TextBox());
        this->label1 = (gcnew System::Windows::Forms::Label());
        this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));
        this->panel1->SuspendLayout();
        this->SuspendLayout();
        //
        // panel1
```

```
        //
        this->panel1->Controls->Add(this->label2);
        this->panel1->Controls->Add(this->textBox1);
        this->panel1->Controls->Add(this->label1);
        this->panel1->Dock = System::Windows::Forms::DockStyle::Fill;
        this->panel1->Location = System::Drawing::Point(0, 0);
        this->panel1->Name = L"panel1";
        this->panel1->Size = System::Drawing::Size(831, 366);
        this->panel1->TabIndex = 0;
        //
        // label2
        //
        this->label2->Dock = System::Windows::Forms::DockStyle::Top;
        this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label2->Location = System::Drawing::Point(0, 0);
        this->label2->Name = L"label2";
        this->label2->Size = System::Drawing::Size(831, 82);
        this->label2->TabIndex = 2;
        this->label2->Text = L"На кожне слово у вас";
        this->label2->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;
        this->label2->Click += gcnew System::EventHandler(this,
&TranslateForm::label2_Click);
        //
        // textBox1
        //
        this->textBox1->CharacterCasing =
System::Windows::Forms::CharacterCasing::Lower;
        this->textBox1->Dock = System::Windows::Forms::DockStyle::Bottom;
        this->textBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
Serif", 24, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->textBox1->Location = System::Drawing::Point(0, 313);
        this->textBox1->Name = L"textBox1";
        this->textBox1->Size = System::Drawing::Size(831, 53);
        this->textBox1->TabIndex = 1;
        this->textBox1->KeyDown += gcnew
System::Windows::Forms::KeyEventHandler(this,
&TranslateForm::textBox1_KeyDown);
```

```
        this->textBox1->KeyPress += gcnew
System::Windows::Forms::KeyPressEventHandler(this,
&TranslateForm::textBox1_KeyPress);
        //
        // label1
        //
        this->label1->Dock = System::Windows::Forms::DockStyle::Fill;
        this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
72, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label1->Location = System::Drawing::Point(0, 0);
        this->label1->Name = L"label1";
        this->label1->Size = System::Drawing::Size(831, 366);
        this->label1->TabIndex = 0;
        this->label1->Text = L"WORDS";
        this->label1->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;
        //
        // timer1
        //
        this->timer1->Interval = 800;
        this->timer1->Tick += gcnew System::EventHandler(this,
&TranslateForm::timer1_Tick);
        //
        // TranslateForm
        //
        this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
        this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
        this->ClientSize = System::Drawing::Size(831, 366);
        this->Controls->Add(this->panel1);
        this->Name = L"TranslateForm";
        this->Text = L"Переклад";
        this->panel1->ResumeLayout(false);
        this->panel1->PerformLayout();
        this->ResumeLayout(false);

}
#pragma endregion
        private: System::Void textBox1_KeyPress(System::Object^  sender,
System::Windows::Forms::KeyPressEventArgs^  e) {

        }
         private: System::Void NextWord()
         {
```

```
 Translator->testedwords++;
 if (!Translator->finished()) {
        label1->Text = Translator->getCurrentWord();
        textBox1->Text = "";
 }
 else {
        int all = Translator->params[3];
        time(&Translator->testend);
        label2->Text = "Показано слів: " + all + " \n Правильних відповідей: " +
Translator->correct.ToString() + "; Неправильних: " + Translator->wrong.size() + "\n
Клікніть для перегляду деталей";
 }
       }

      private: System::Void textBox1_KeyDown(System::Object^  sender,
System::Windows::Forms::KeyEventArgs^  e) {

 if ((e->KeyCode == Keys::Enter) && !Translator->empty && !Translator-
>finished()) {
        Translator->CheckTranslation(textBox1->Text);
        NextWord();
 }
      }
private: System::Void timer1_Tick(System::Object^  sender, System::EventArgs^  e) {
        if (!Translator->finished())
        {

 time_t ctime, timediff;
 time(&ctime);
 timediff = ctime - Translator->starttime;
 if (timediff > Translator->params[1]) {
        Translator->CheckTranslation(textBox1->Text);
        NextWord();
 } else
        this->label2->Text = "На кожне слово у вас " + (Translator->params[1] -
timediff) + " сек";
        }
}
private: System::Void label2_Click(System::Object^  sender, System::EventArgs^  e) {
        if (Translator->finished()){
 this->f = gcnew MyForm();
 Translator->ShowDetails(this->f->dataGridView1, this->f->label1);
 this->f->Show();
```

```
        }
}
};


}
```

Файл Form1.h

```
#pragma once

namespace TaskManager {

        using namespace System;
        using namespace System::ComponentModel;
        using namespace System::Collections;
        using namespace System::Windows::Forms;
        using namespace System::Data;
        using namespace System::Drawing;

        /// <summary>
        /// Сводка для Form1
        /// </summary>
        public ref class Form1 : public System::Windows::Forms::Form
        {
        public:
Form1(void)
{
        InitializeComponent();
        //
        //TODO: добавьте код конструктора
        //
        this->f = gcnew TranslateForm;
}

        protected:
/// <summary>
/// Освободить все используемые ресурсы.
/// </summary>
~Form1()
{
        if (components)
```

```
        {
delete components;
        }
}
```

```
        private: Form2^ f2;
        private: TranslateForm^ f;
        private: Admin^ fa;


        private: System::Windows::Forms::Panel^  panel1;
        private: System::Windows::Forms::MenuStrip^  menuStrip1;




        private: System::Windows::Forms::Button^  button5;
        private: System::Windows::Forms::Button^  button6;
        private: System::Windows::Forms::TextBox^  textBox1;
        private: System::Windows::Forms::Label^  label1;
        private: System::Windows::Forms::Panel^  panel2;
        private: System::Windows::Forms::ComboBox^  comboBox1;
```

```
        protected:



        private:
/// <summary>
/// Требуется переменная конструктора.
/// </summary>
System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
/// <summary>
/// Обязательный метод для поддержки конструктора - не изменяйте
/// содержимое данного метода при помощи редактора кода.
/// </summary>
void InitializeComponent(void)
{
        this->panel1 = (gcnew System::Windows::Forms::Panel());
        this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());
        this->label1 = (gcnew System::Windows::Forms::Label());
        this->panel2 = (gcnew System::Windows::Forms::Panel());
        this->textBox1 = (gcnew System::Windows::Forms::TextBox());
        this->button6 = (gcnew System::Windows::Forms::Button());
        this->button5 = (gcnew System::Windows::Forms::Button());
        this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());
        this->panel1->SuspendLayout();
        this->panel2->SuspendLayout();
        this->SuspendLayout();
        //
        // panel1
        //
        this->panel1->Controls->Add(this->comboBox1);
```

```
        this->panel1->Controls->Add(this->label1);
        this->panel1->Controls->Add(this->panel2);
        this->panel1->Controls->Add(this->button5);
        this->panel1->Controls->Add(this->menuStrip1);
        this->panel1->Dock = System::Windows::Forms::DockStyle::Top;
        this->panel1->Location = System::Drawing::Point(0, 0);
        this->panel1->Name = L"panel1";
        this->panel1->Size = System::Drawing::Size(922, 402);
        this->panel1->TabIndex = 5;
        this->panel1->Paint += gcnew
System::Windows::Forms::PaintEventHandler(this, &Form1::panel1_Paint);
        //
        // comboBox1
        //
        this->comboBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
Serif", 12, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->comboBox1->FormattingEnabled = true;
        this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^  >(2)
{ L"Англійська", L"Українська" });
        this->comboBox1->Location = System::Drawing::Point(386, 51);
        this->comboBox1->Name = L"comboBox1";
        this->comboBox1->Size = System::Drawing::Size(152, 33);
        this->comboBox1->TabIndex = 11;

        this->comboBox1->SelectedIndex = 0;
        //
        // label1
        //
        this->label1->AutoSize = true;
        this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->label1->Location = System::Drawing::Point(356, 9);
        this->label1->Name = L"label1";
        this->label1->Size = System::Drawing::Size(205, 25);
        this->label1->TabIndex = 10;
        this->label1->Text = L"Виберіть мову тесту";
        //
        // panel2
        //
        this->panel2->Controls->Add(this->textBox1);
```

```
        this->panel2->Controls->Add(this->button6);
        this->panel2->Location = System::Drawing::Point(332, 344);
        this->panel2->Name = L"panel2";
        this->panel2->Size = System::Drawing::Size(271, 55);
        this->panel2->TabIndex = 9;
        //
        // textBox1
        //
        this->textBox1->Location = System::Drawing::Point(54, 20);
        this->textBox1->Name = L"textBox1";
        this->textBox1->Size = System::Drawing::Size(63, 22);
        this->textBox1->TabIndex = 8;
        //
        // button6
        //
        this->button6->Location = System::Drawing::Point(134, 17);
        this->button6->Name = L"button6";
        this->button6->Size = System::Drawing::Size(60, 25);
        this->button6->TabIndex = 7;
        this->button6->Text = L"Admin";
        this->button6->UseVisualStyleBackColor = true;
        this->button6->Click += gcnew System::EventHandler(this,
&Form1::button6_Click);
        //
        // button5
        //
        this->button5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif",
19.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(204)));
        this->button5->Location = System::Drawing::Point(321, 120);
        this->button5->Name = L"button5";
        this->button5->Size = System::Drawing::Size(282, 207);
        this->button5->TabIndex = 6;
        this->button5->Text = L"Старт";
        this->button5->UseVisualStyleBackColor = true;
        this->button5->Click += gcnew System::EventHandler(this,
&Form1::button5_Click);
        //
        // menuStrip1
        //
        this->menuStrip1->ImageScalingSize = System::Drawing::Size(20, 20);
        this->menuStrip1->Location = System::Drawing::Point(0, 0);
        this->menuStrip1->Name = L"menuStrip1";
```

```
        this->menuStrip1->Size = System::Drawing::Size(922, 24);
        this->menuStrip1->TabIndex = 5;
        this->menuStrip1->Text = L"menuStrip1";
        //
        // Form1
        //
        this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
        this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
        this->ClientSize = System::Drawing::Size(922, 398);
        this->Controls->Add(this->panel1);
        this->MainMenuStrip = this->menuStrip1;
        this->MaximizeBox = false;
        this->Name = L"Form1";
        this->Text = L"Програма перекладу";
        this->panel1->ResumeLayout(false);
        this->panel1->PerformLayout();
        this->panel2->ResumeLayout(false);
        this->panel2->PerformLayout();
        this->ResumeLayout(false);

}
#pragma endregion

private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e)
{
}

private: System::Void button2_Click(System::Object^  sender, System::EventArgs^  e)
{


 }
private: System::Void button3_Click(System::Object^  sender, System::EventArgs^  e)
{
        /*if (!this->f2) {

        }*/
        this->f2=gcnew Form2();
        this->f2->Show();

 }
```

```
private: System::Void panel1_Paint(System::Object^  sender,
System::Windows::Forms::PaintEventArgs^  e) {
 }
private: System::Void button5_Click(System::Object^  sender, System::EventArgs^  e)
{

        this->f = gcnew TranslateForm;
        this->f->Start(comboBox1->SelectedIndex);
        this->f->Show();
}

private: System::Void button6_Click(System::Object^  sender, System::EventArgs^  e)
{
        if (this->f && (textBox1->Text == gcnew String(this->f->Translator-
>pass.c_str()))) {
 this->fa = gcnew Admin();
 this->fa->Show();
        }
        else  MessageBox::Show("Неправильний пароль!", "Info",
MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}
};
}
```

ДОДАТОК И

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ
ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: _____

_____

_____

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ кафедра обчислювальної техніки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 65% Схожість 35%

Аналіз звіту подібності (відмітити потрібне):

- ✓ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.
- ☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
- ☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Захарченко С.М.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ _____
(підпис) (прізвище, ініціали)

Керівник роботи _____ _____
(підпис) (прізвище, ініціали)