МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів

Кафедра систем управління літальних апаратів

**Лабораторна робота №11**

з дисципліни «Алгоритмізація та програмування»

Тема: "Розробка десктоп-застосунків в середовищі QtCreator"

ХАІ.301.175.318.15 ЛР

Виконав студент гр. № 318

Мальківська В. А

(підпис, дата) (П.І.Б.)

Перевірив к.т.н., доцент

( вчена ступінь, вчене звання)

  Гавриленко О. В

(підпис, дата) (П.І.Б.)

2025

# МЕТА РОБОТИ

Ознайомитися з основами розробки програм з використанням QtDesigner і навчитися розробляти десктоп-застосунки із графічним користувацьким

інтерфейсом для введення/виведення даних на мові програмування C++ в середовищі QtCreator.

ПОСТАНОВКА ЗАДАЧІ

**Завдання 1.** Вивчити алгоритм створення проекту Qt Widgets Application в середовищі QtCreator. Ознайомитись з налаштуваннями основних елементів для введення, виведення, компоновки форми і управління.

**Завдання 2.** Для вирішеннязавдання відповідно до варіанта (Version6):

**А.** Спроектувати і реалізувати в конструкторі форм графічний інтерфейс

програми з віджетами QLabel, QLineEdit і QPushButton. \*Використати додаткові віджети.

**B.** Додати і відлагодити програмний код для введення вхідних даних з

перевіркою на коректність (використати QMessageBox для виведення

сповіщень), обчислень і виведення результатів.

**C\*.** Додати пункти меню у QMenuBar для зчитування вхідних даних і

збереження результатів в файл з використанням стандартних діалогів для вибору файла.

**Version6.** Дано тризначне число. Знайти суму і добуток його цифр.

ВИКОНАННЯ РОБОТИ

**Завдання 1. Алгоритм створення проєкту десктоп-застосунку у Windows Forms**

**Крок 1.** **Запуск середовища Visual Studio**

1. Відкрий Visual Studio з робочого столу або меню "Пуск";

2. Дочекатися завантаження початкового екрана;

3. Оберіть "Create a new project" на головному екрані;

4. Якщо Visual Studio англійською — знайти пункт "Create a new project";

5. У разі відсутності кнопки — натиснути File > New > Project;

**Крок 2. Вибір шаблону проєкту**

1. У полі пошуку шаблонів введіть: CLR Empty Project;

2. Знайдіть шаблон з повною назвою "CLR Empty Project (.NET Framework)";

3. Переконайтеся, що мова вибрана — Visual C++;

4. Виберіть цей шаблон і натисніть Next;

5. Якщо шаблону немає — додати через Visual Studio Installer > C++/CLI support;

**Крок 3. Налаштування параметрів проєкту**

1. Введіть ім’я проєкту, наприклад: LR11\_FormsApp;

2. Оберіть зручну теку для збереження проєкту;

3. Переконайтеся, що поле "Solution" не порожнє (або створити нову solution);

4. Натисніть Create для створення проєкту;

5. Дочекайтеся, поки проєкт з’явиться у Solution Explorer;

**Крок 4. Додавання Windows-форми**

1. У Solution Explorer клацніть правою кнопкою по назві проєкту;

2. Оберіть Add > New Item...;

3. У списку оберіть UI > Windows Form (.h);

4. Назвіть файл, наприклад: Form1.h.;

5. Натисніть Add, щоб додати форму до проєкту;

**Крок 5. Підключення форми до main.cpp**

1. Відкрийте файл main.cpp у проєкті;

2. Видаліть початковий вміст, якщо він є;

3. Вставте наступний код:

#include "Form1.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThread]

void main() {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

LR11\_FormsApp::Form1 form;

Application::Run(%form);

}

4. Замініть LR11\_FormsApp на власну назву проєкту, якщо інша;

5. Збережіть зміни за допомогою Ctrl + S;

**Крок 6. Відкриття конструктора форми (Designer)**

1. У Solution Explorer двічі клікніть на Form1.h;

2. Переконайтеся, що з’явилася вкладка [Design] вгорі;

3. Перейдіть на вкладку Design, щоб побачити візуальний інтерфейс;

4. Якщо дизайнер не відкривається — натисніть правою кнопкою → View Designer;

5. Збільшіть розмір форми, щоб зручно розміщувати елементи;

**Крок 7. Додавання 6 елементів керування (віджетів)**

1. Відкрийте панель Toolbox (якщо не видно — View > Toolbox);

2. Перетягніть на форму такі елементи: Label, TextBox, 3 Buttons, ще один Label;

3. Кожному елементу задайте у вікні Properties власне ім’я (Name);

4. Змініть властивість Text відповідно до призначення кнопки або напису;

5. Вирівняйте елементи у логічному порядку: введення, обчислення, збереження;

**Крок 8. Налаштування властивостей елементів**

Для кожного елемента відкрийте вікно Properties та налаштуйте:

**Label labelInput:**

Name = labelInput;

Text = Введіть тризначне число;

Font = Microsoft Sans Serif, 10pt;

Location = (20, 20);

AutoSize = True;

**TextBox textBoxInput:**

Name = textBoxInput;

Size = (150, 20);

Location = (200, 20);

Text = (залишити порожнім);

Multiline = False;

**Button buttonCalculate:**

Name = buttonCalculate;

Text = Обчислити;

BackColor = LightBlue;

Location = (20, 60);

Подвійний клік — для створення події Click;

**Label labelResult:**

Name = labelResult;

Text = Результат;

ForeColor = DarkGreen;

Font = Bold, 10pt;

Location = (20, 100);

**Button buttonLoad:**

Name = buttonLoad;

Text = Зчитати з файлу;

BackColor = LightYellow;

Location = (20, 140);

Подвійний клік для події Click;

**Button buttonSave:**

Name = buttonSave;

Text = Зберегти у файл;

BackColor = LightGray;

Location = (150, 140);

Подвійний клік для події Click;

**Крок 9. Додавання обробників подій кнопок**

1. У Designer двічі клікніть на кожну кнопку — створиться метод події;

2. Visual Studio відкрийте вкладку Form1.h з кодом;

3. Всередині кожного методу можна реалізувати потрібну логіку;

4. Наприклад: читання числа, обчислення суми цифр, збереження у файл;

5. Код реалізації додається в Завданні 2;

**Крок 10. Збірка та запуск застосунку**

1. Збережіть всі зміни: File > Save All або Ctrl + Shift + S;

2. Зберіть проєкт: Build > Build Solution або Ctrl + Shift + B;

3. Якщо є помилки — перевірте синтаксис, імена, підключення форми;

4. Запустіть застосунок: Debug > Start Without Debugging або Ctrl + F5;

5. У вікні форми перевірте наявність усіх 6 віджетів та їх роботу;

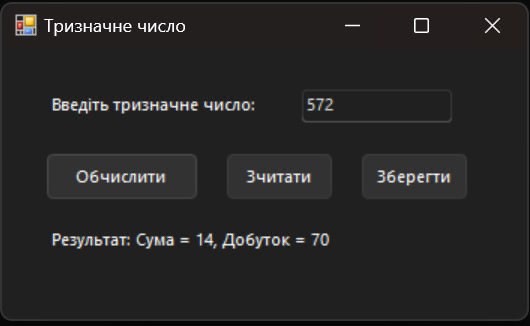
**Завдання 2. Проєкт інтерфейсу**

Рис.1 – Інтерфейс застосунку

Компоненти інтерфейсу:

1. Label – "Введіть тризначне число";

2. TextBox – для введення числа;

3. Button – "Обчислити";

4. Label – "Результат";

5. Button – "Зчитати з файлу";

6. Button – "Зберегти у файл";

Лістинг коду вирішення завдання 2 наведено в дод. А (стор. 8-13)

Екран роботи програми показаний на рис. Б.1-Б.5 (дод. Б, стор.14-16)

Діаграма активності для завдання 2 наведено на рис. Б.6-Б.7 (дод. Б, стор.17-18)

ВИСНОВКИ

У ході роботи було засвоєно основи використання структур у мові C++ та їх передачі в функції. Практична реалізація дозволила зрозуміти, як структури допомагають організувати дані і спростити роботу з ними. Виконання завдань у середовищі Visual Studio сприяло розвитку навичок програмування та налагодження коду.

ДОДАТОК А

Лістинг до коду

Form1.h

#pragma once // Забороняє багаторазове включення цього файлу під час компіляції

namespace LR\_11\_Vik { // Власний простір імен для проєкту

using namespace System; // Базові .NET класи

using namespace System::ComponentModel; // Компоненти форми

using namespace System::Collections; // Колекції (не використано, але підключено)

using namespace System::Windows::Forms; // Елементи управління Windows Forms

using namespace System::Data; // Робота з даними (не використано, але підключено)

using namespace System::Drawing; // Робота з графікою та координатами

using namespace System::IO; // Для роботи з файлами

// Оголошення класу форми, який наслідує базовий клас Form

public ref class Form1 : public System::Windows::Forms::Form

{

public:

Form1(void) // Конструктор класу без параметрів

{

InitializeComponent(); // Ініціалізація елементів інтерфейсу форми

}

protected:

~Form1() // Деструктор класу, звільняє ресурси

{

if (components) // Якщо існують компоненти форми

{

delete components; // Звільняємо пам'ять

}

}

private:

System::Windows::Forms::Label^ labelInput; // Мітка з текстом для вводу числа

System::Windows::Forms::TextBox^ textBoxInput; // Текстове поле для вводу числа

System::Windows::Forms::Button^ buttonCalculate; // Кнопка для обчислення суми і добутку

System::Windows::Forms::Label^ labelResult; // Мітка для виводу результату

System::Windows::Forms::Button^ buttonLoad; // Кнопка для завантаження числа з файлу

System::Windows::Forms::Button^ buttonSave; // Кнопка для збереження результату у файл

System::Windows::Forms::OpenFileDialog^ openFileDialog1; // Діалог відкриття файлу

System::Windows::Forms::SaveFileDialog^ saveFileDialog1; // Діалог збереження файлу

System::ComponentModel::IContainer^ components; // Контейнер компонентів форми

#pragma region Windows Form Designer generated code // Початок блоку автоматично згенерованого коду

void InitializeComponent(void) // Метод ініціалізації всіх елементів форми

{

this->labelInput = (gcnew System::Windows::Forms::Label()); // Створюємо мітку

this->textBoxInput = (gcnew System::Windows::Forms::TextBox()); // Створюємо текстове поле

this->buttonCalculate = (gcnew System::Windows::Forms::Button()); // Створюємо кнопку обчислення

this->labelResult = (gcnew System::Windows::Forms::Label()); // Створюємо мітку результату

this->buttonLoad = (gcnew System::Windows::Forms::Button()); // Створюємо кнопку завантаження

this->buttonSave = (gcnew System::Windows::Forms::Button()); // Створюємо кнопку збереження

this->openFileDialog1 = (gcnew System::Windows::Forms::OpenFileDialog()); // Створюємо діалог відкриття файлу

this->saveFileDialog1 = (gcnew System::Windows::Forms::SaveFileDialog()); // Створюємо діалог збереження файлу

this->SuspendLayout(); // Призупиняємо оновлення форми для налаштування елементів

// Налаштування labelInput (мітки для вводу)

this->labelInput->AutoSize = true; // Автоматичне підлаштування розміру під текст

this->labelInput->Location = System::Drawing::Point(30, 30); // Встановлюємо позицію (X=30, Y=30)

this->labelInput->Name = L"labelInput"; // Встановлюємо ім'я об'єкта

this->labelInput->Size = System::Drawing::Size(160, 13); // Початковий розмір (для дизайнера)

this->labelInput->Text = L"Введіть тризначне число:"; // Текст мітки

// Налаштування textBoxInput (текстове поле вводу)

this->textBoxInput->Location = System::Drawing::Point(200, 27); // Позиція на формі

this->textBoxInput->Name = L"textBoxInput"; // Ім'я об'єкта

this->textBoxInput->Size = System::Drawing::Size(100, 20); // Розмір (ширина х висота)

// Налаштування buttonCalculate (кнопка "Обчислити")

this->buttonCalculate->Location = System::Drawing::Point(30, 70); // Позиція кнопки

this->buttonCalculate->Name = L"buttonCalculate"; // Ім'я кнопки

this->buttonCalculate->Size = System::Drawing::Size(100, 30); // Розмір кнопки

this->buttonCalculate->Text = L"Обчислити"; // Текст на кнопці

this->buttonCalculate->Click += gcnew System::EventHandler(this, &Form1::buttonCalculate\_Click); // Підписка на подію натискання кнопки

// Налаштування labelResult (мітка результату)

this->labelResult->AutoSize = true; // Автоматичний розмір за текстом

this->labelResult->Location = System::Drawing::Point(30, 120); // Позиція

this->labelResult->Size = System::Drawing::Size(70, 13); // Початковий розмір

this->labelResult->Text = L"Результат:"; // Текст мітки

// Налаштування buttonLoad (кнопка "Зчитати" для завантаження з файлу)

this->buttonLoad->Location = System::Drawing::Point(150, 70); // Позиція кнопки

this->buttonLoad->Name = L"buttonLoad"; // Ім'я кнопки

this->buttonLoad->Size = System::Drawing::Size(70, 30); // Розмір кнопки

this->buttonLoad->Text = L"Зчитати"; // Текст на кнопці

this->buttonLoad->Click += gcnew System::EventHandler(this, &Form1::buttonLoad\_Click); // Підписка на подію

// Налаштування buttonSave (кнопка "Зберегти" для збереження результату у файл)

this->buttonSave->Location = System::Drawing::Point(240, 70); // Позиція кнопки

this->buttonSave->Name = L"buttonSave"; // Ім'я кнопки

this->buttonSave->Size = System::Drawing::Size(70, 30); // Розмір кнопки

this->buttonSave->Text = L"Зберегти"; // Текст кнопки

this->buttonSave->Click += gcnew System::EventHandler(this, &Form1::buttonSave\_Click); // Підписка на подію

// Налаштування параметрів форми

this->ClientSize = System::Drawing::Size(350, 180); // Встановлення розміру вікна форми

this->Controls->Add(this->labelInput); // Додаємо елемент labelInput на форму

this->Controls->Add(this->textBoxInput); // Додаємо текстове поле

this->Controls->Add(this->buttonCalculate); // Додаємо кнопку обчислення

this->Controls->Add(this->labelResult); // Додаємо мітку для результату

this->Controls->Add(this->buttonLoad); // Додаємо кнопку завантаження

this->Controls->Add(this->buttonSave); // Додаємо кнопку збереження

this->Name = L"Form1"; // Ім'я форми

this->Text = L"Тризначне число"; // Заголовок вікна

this->ResumeLayout(false); // Продовжуємо оновлення макету

this->PerformLayout(); // Примусове оновлення макету з урахуванням нових елементів

}

#pragma endregion

// Обробник події натискання кнопки "Обчислити"

private: System::Void buttonCalculate\_Click(System::Object^ sender, System::EventArgs^ e) {

int number; // Змінна для збереження введеного числа

// Перевіряємо, чи можна конвертувати текст в ціле число і чи воно тризначне

if (Int32::TryParse(textBoxInput->Text, number) && number >= 100 && number <= 999)

{

int a = number / 100; // Визначаємо сотні (перша цифра)

int b = (number / 10) % 10; // Десятки (друга цифра)

int c = number % 10; // Одиниці (третя цифра)

int sum = a + b + c; // Обчислюємо суму цифр

int product = a \* b \* c; // Обчислюємо добуток цифр

// Виводимо результат у мітку labelResult

labelResult->Text = "Результат: Сума = " + sum.ToString() + ", Добуток = " + product.ToString();

}

else // Якщо введено некоректне число

{

// Виводимо повідомлення про помилку

MessageBox::Show("Будь ласка, введіть коректне тризначне число.", "Помилка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

}

// Обробник натискання кнопки "Зчитати" (завантажити число з файлу)

private: System::Void buttonLoad\_Click(System::Object^ sender, System::EventArgs^ e) {

// Відкриваємо діалог вибору файлу

if (openFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

try {

// Читаємо весь текст із обраного файлу

String^ content = File::ReadAllText(openFileDialog1->FileName);

// Записуємо текст у поле вводу, обрізаючи зайві пропуски

textBoxInput->Text = content->Trim();

}

catch (Exception^ ex) { // Обробка помилок зчитування файлу

MessageBox::Show("Помилка при зчитуванні файлу: " + ex->Message);

}

}

}

// Обробник натискання кнопки "Зберегти" (запис результату у файл)

private: System::Void buttonSave\_Click(System::Object^ sender, System::EventArgs^ e) {

// Відкриваємо діалог збереження файлу

if (saveFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

try {

// Записуємо текст з мітки результату у файл

File::WriteAllText(saveFileDialog1->FileName, labelResult->Text);

}

catch (Exception^ ex) { // Обробка помилок запису

MessageBox::Show("Помилка при збереженні файлу: " + ex->Message);

}

}

}

};

}

Main.cpp

#include "Form1.h" // Підключаємо файл з описом форми

using namespace System; // Базові .NET класи

using namespace System::Windows::Forms; // Класи для Windows Forms

// Основна точка входу в програму

[STAThreadAttribute] // Вказуємо, що програма буде працювати в однопоточному STA режимі (необхідно для Windows Forms)

int main(array<String^>^ args)

{

Application::EnableVisualStyles(); // Включаємо сучасний вигляд елементів управління

Application::SetCompatibleTextRenderingDefault(false); // Встановлюємо параметр рендерингу тексту

LR\_11\_Vik::Form1 form; // Створюємо об'єкт форми

Application::Run(%form); // Запускаємо цикл обробки подій з цією формою

return 0; // Завершення програми

}

ДОДАТОК Б

Скрін-шоти вікна виконання програми

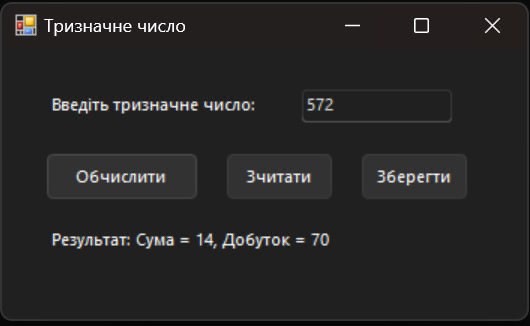


Рисунок Б.1 — Екран виконання програми для вирішення завдання 2

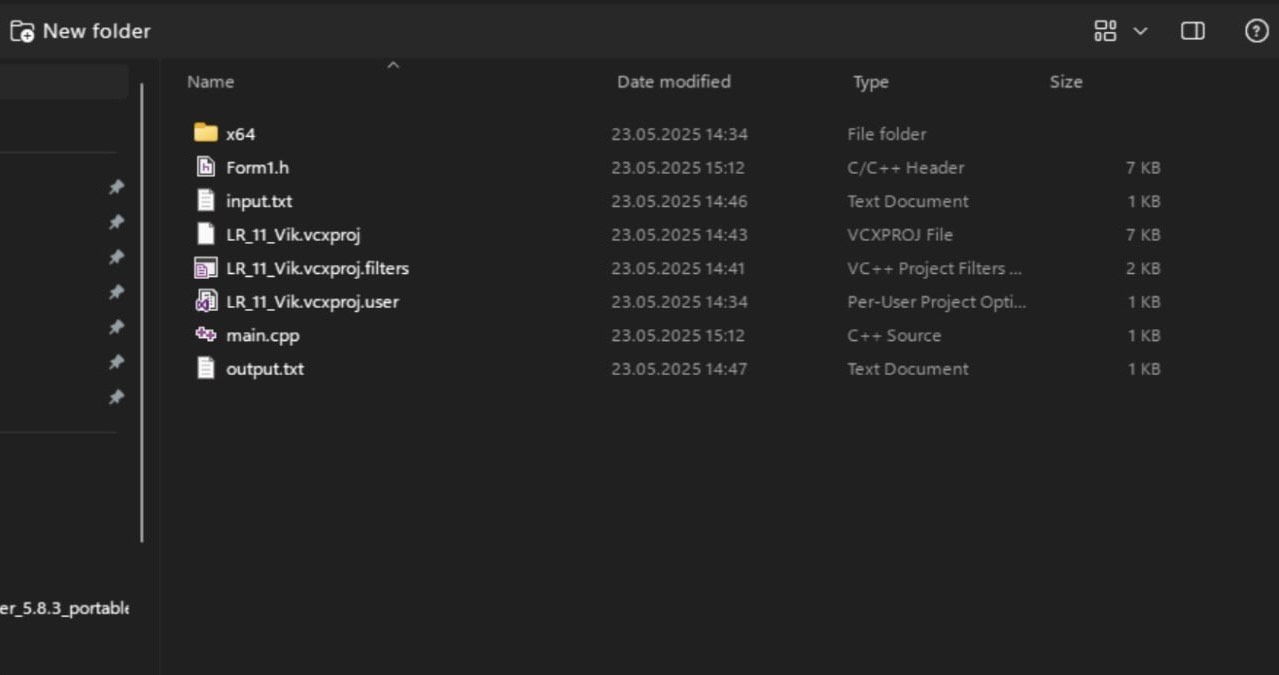


Рисунок Б.2 — Екран виконання програми для вирішення завдання 2

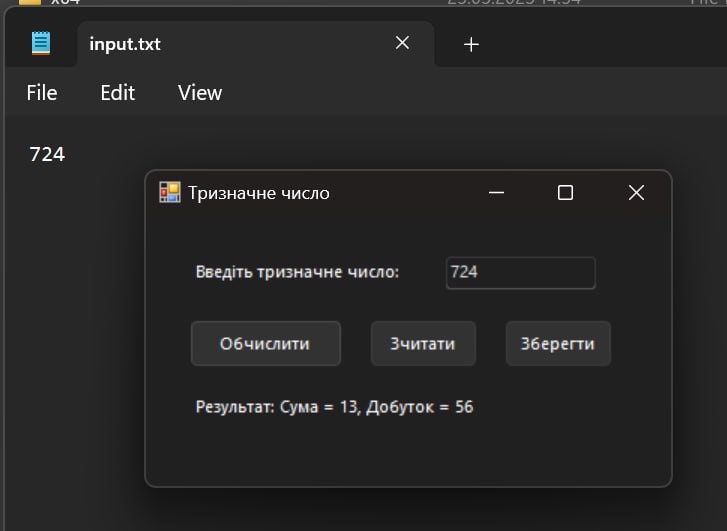


Рисунок Б.3 — Екран виконання програми для вирішення завдання 2

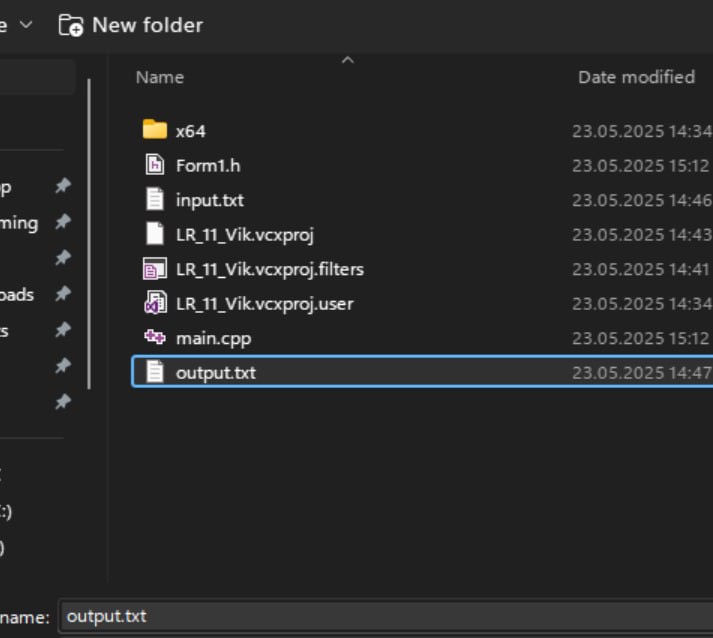
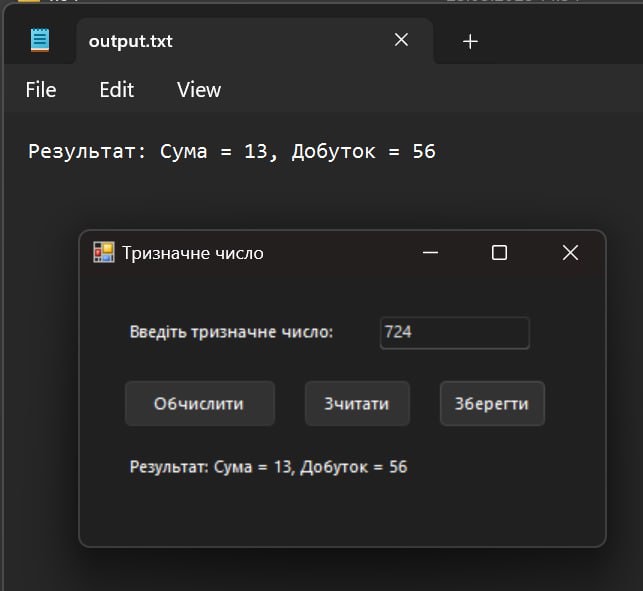


Рисунок Б.4 — Екран виконання програми для вирішення завдання 2

 Рисунок Б.5 — Екран виконання програми для вирішення завдання 2

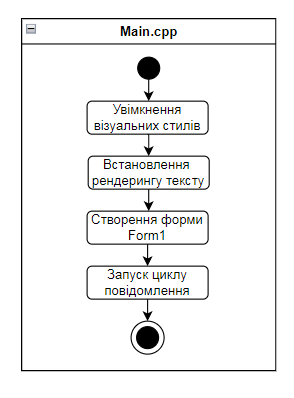


Рисунок Б.7 — Діаграма активності для меню

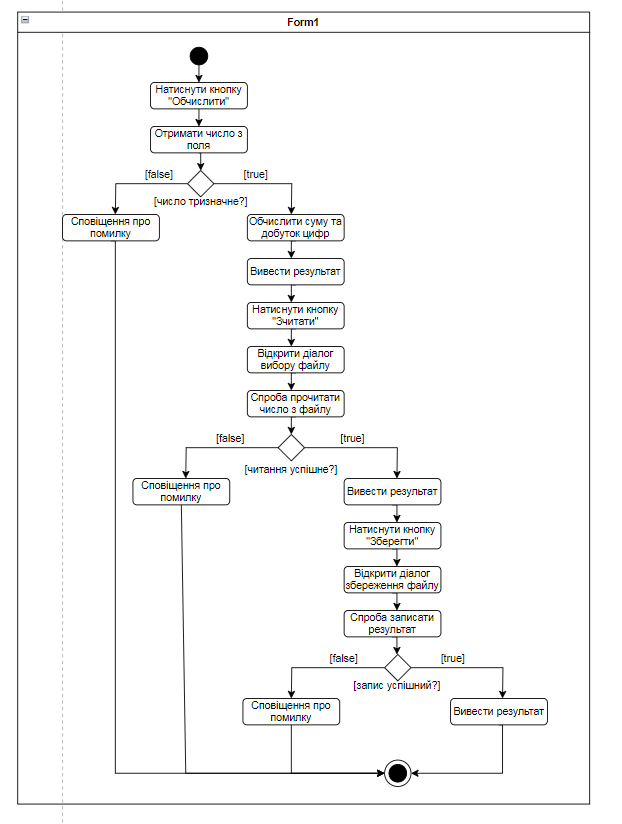


Рисунок Б.8 — Діаграма активності для завдання 1