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»
на тему «Розробка десктоп-застосунків в середовищі Visual Studio»

ХАІ.301. 175. 318.08 ЛР

Виконав студент гр. 318

Каріна ГЛЄБОВА
(підпис, дата) (П.І.Б.)

Перевірив
к.т.н., доц. Олена ГАВРИЛЕНКО
(підпис, дата) (П.І.Б.)

2025

МЕТА РОБОТИ

Ознайомитися 3 основами розробки програм з використанням Windows Forms і навчитися розробляти десктоп-застосунки із графічним користувацьким інтерфейсом для введення/виведення даних на мові програмування C++ в середовищі Visual Studio.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Вивчити алгоритм створення проекту Windows Forms в середовищі Visual Studio. Ознайомитись з налаштуваннями основних елементів для введення, виведення і управління. *Опрацювати навички створення та налаштування десктоп-застосунку у Visual Studio.

Завдання 2. Для вирішення завдання відповідно до варіанта 52:

A. Спроектувати і реалізувати в конструкторі форм графічний інтерфейс програми з об'єктами Label, TextBox і Button. *Використати інші елементи управління.

B. Додати програмний код для введення вхідних даних, обчислень і виведення результатів. *Відтестувати і налагодити десктоп-застосунок

C*. Передбачити зчитування даних з файлу з використанням стандартного діалогу для вибору файла, а також збереження результатів в файл із відповідним діалогом. Відтестувати і налагодити десктоп-застосунок.

Варіант 52:

Швидкість першого автомобіля V1 км/год, другого — V2 км/год, відстань між ними S км. Визначити відстань між ними через T годин, якщо автомобілі спочатку рухаються назустріч один одному. Дана відстань рівна модулю різниці початкової відстані і загального шляху, пройденого автомобілями; загальний шлях = час * сумарна швидкість.

ВИКОНАННЯ РОБОТИ

Завдання 1.

Крок 1. Запуск Visual Studio та створення нового проекту

1. Відкрити Visual Studio.
2. Оберати Create a new project (Створити новий проект).
3. У полі пошуку ввести Windows Forms App.
4. Знайти шаблон Windows Forms App (.NET Framework) з мовою C++/CLI.
5. Обрати його і натиснути Next.
6. Ввести назву проекту, вибрати розташування і натиснути Create.

Крок 2. Ознайомлення з проектом і файлами

Після створення проекту буде головне вікно редактора форм — Form1.
У проекті будуть файли:
Form1.h — код форми та UI елементів.
main.cpp — точка входу, де запускається форма.

Крок 3. Відкриття конструктора форм

Відкрити файл Form1.h.

У верхній частині або через контекстне меню вибрати View Designer (Переглянути конструктор).

У конструкторі буде візуальне представлення форми, на яку можна додавати елементи.

Крок 4. Додавання основних елементів керування (Controls)

1. Toolbox (Панель елементів)
   Зліва або справа у Visual Studio знайти вкладку Toolbox.
   Там є різні елементи: Button, TextBox, Label, ComboBox, CheckBox тощо.
2. Додавання елемента
   Перетягнути потрібний елемент, наприклад Button, на форму.
   Аналогічно додати TextBox для введення тексту і Label для виведення інформації.
3. Налаштування властивостей

Обрати елемент на формі.

У вікні Properties змінити властивості, наприклад:

Name — ім’я змінної елемента (щоб посилатися на нього в коді).

Text — текст, який відображається на кнопці або мітці.

Size і Location — розмір і позиція елемента.

Крок 5. Додавання обробників подій (наприклад, кнопки)

1. Виділити кнопку на формі.
2. У вікні Properties перейти на вкладку Events (блискавка).
3. Знайти подію `Click`.
4. Двічі клікнути на полі праворуч — Visual Studio створить метод-обробник у коді.
5. У методі написати логіку, наприклад, для виведення тексту з TextBox у

Label:

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

    label1->Text = textBox1->Text;

}
```

Крок 6. Запуск проекту

Натиснути F5 або кнопку Start для компіляції і запуску.

Відкриється ваша форма Windows Forms.

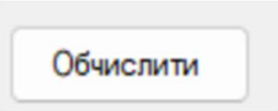
Можна ввести текст у текстове поле, натиснути кнопку і побачити результат.

Крок 7. Поради по роботі з елементами

Для введення тексту — використовувати TextBox.

Для виведення — Label або TextBox в режимі ReadOnly.

Для кнопок — Button з обробниками подій.



Завдання 2.

1. Імпорти та оголошення простору і класу:

```
#pragma once
#include <cmath>
#include <fstream>

namespace CarDistanceApp {
    using namespace System;
    using namespace System::Windows::Forms;
    ...
```

Що відбувається:

Імпортуються стандартні бібліотеки `cmath` (для `abs`) і `fstream` (для роботи з файлами).

Відкривається простір імен `CarDistanceApp`, в якому міститься клас форми `Form1`.

2. Оголошення елементів інтерфейсу:

```
private:
    Label^ labelV1;
    Label^ labelV2;
    Label^ labelS;
    Label^ labelT;
    TextBox^ txtV1;
    TextBox^ txtV2;
     TextBox^ txtS;
    TextBox^ txtT;
    Button^ btnCalculate;
    TextBox^ txtResult;
    Button^ btnLoadFile;
    Button^ btnSaveFile;
    OpenFileDialog^ openFileDialog;
```

```
    SaveFileDialog^ saveFileDialog;
```

Призначення:

Створюються компоненти UI: мітки (`Label`), текстові поля (`TextBox`), кнопки (`Button`) і діалоги для файлів.

Кожен елемент згодом буде ініціалізований в `InitializeComponent()`.

3. Побудова інтерфейсу у методі InitializeComponent():

```
// labelV1->Text = "V1 (км/год):", txtV1->Location = ...
// btnCalculate->Text = "Обчислити", btnCalculate->Click += ...
```

Вхідні дані (ім’я, опис, тип, обмеження):

V1 – швидкість першого автомобіля, число із плаваючою точкою, дійсний тип.

V2 – швидкість другого автомобіля, число із плаваючою точкою, дійсний тип.

S – відстань між двома автомобілями, число із плаваючою точкою, дійсний тип.

T – час, за який автомобілі пройдуть обчислювану відстань, число із плаваючою точкою, дійсний тип.

Вихідні дані (ім’я, опис, тип):

Result – відстань між автомобілями через заданий час, число із плаваючою точкою, дійсний тип.

Керування:

btnCalculate – кнопка «Обчислити».

btnLoadFile – кнопка «Зчитати з файлу».

btnSaveFile – кнопка «Зберегти у файл».

4. Обчислення результату

```
    private:  System::Void  btnCalculate_Click(System::Object^  sender,
System::EventArgs^ e)
    {
        double V1 = Double::Parse(txtV1->Text);
        double V2 = Double::Parse(txtV2->Text);
        double S = Double::Parse(txtS->Text);
```

```
        double T = Double::Parse(txtT->Text);

        double distance = Math::Abs(S - T * (V1 + V2));
        txtResult->Text = "Відстань між автомобілями через " +
T.ToString("F2") + " год: " + distance.ToString("F2") + " км";
    }
```

Що відбувається:

Читаються значення з `TextBox` і перетворюються в числа.

Перевіряється, що числа не від’ємні.

Згідно з умовою: відстань = |S − T * (V1 + V2)|.

Результат записується у `txtResult` у зрозумілому вигляді.

Обробка помилок try-catch забезпечує перевірку формату введених даних і виводить повідомлення користувачу у разі помилки.

5. Зчитування даних з файлу

```
    private: System::Void btnLoadFile_Click(System::Object^ sender,
System::EventArgs^ e)
    {
        if (openFileDialog->ShowDialog() == DialogResult::OK) {
            array<String^>^ lines = File::ReadAllLines(openFileDialog-
>FileName);
            if (lines->Length >= 4) {
                txtV1->Text = lines[0];
                txtV2->Text = lines[1];
                txtS->Text = lines[2];
                txtT->Text = lines[3];
            }
        }
    }
```

Після вибору файлу через стандартний діалог OpenFileDialog, зчитуються рядки з файлу.

Кожен рядок відповідає одному з параметрів (V1, V2, S, T).

Дані вставляються у відповідні TextBox.

6. Збереження результату у файл

```
    private:   System::Void   btnSaveFile_Click(System::Object^   sender,
System::EventArgs^ e)
    {
        if (saveFileDialog->ShowDialog() == DialogResult::OK) {
            File::WriteAllText(saveFileDialog->FileName,       txtResult-
>Text);
            MessageBox::Show("Результат успішно збережено.");
        }
    }
```

Через SaveFileDialog користувач обирає місце для збереження.

Вміст txtResult записується у текстовий файл.

Виводиться підтвердження успішного збереження.

7. Головна функція запуску форми

```
[STAThreadAttribute]
int main(array<String^>^ args)
{
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);
    CarDistanceApp::Form1 form;
    Application::Run(%form);
    return 0;
}
```

Лістинг коду вирішення задачі 2 наведено в дод. А (стор. 10-16).

Екран роботи програми показаний на рис. Б.1-Б.5. (дод. Б, стор. 17-21)

Приклад діаграми для завдання 2 наведено на рис. Б.6-Б.7. (дод. Б, стор. 17-21)

ВИСНОВКИ

Було ознайомлено з основами розробки програм з використанням Windows Forms. Було розроблено десктоп-застосунки із графічним користувацьким інтерфейсом для введення/виведення даних на мові програмування C++ в середовищі Visual Studio

ДОДАТОК А
Лістинг коду програми

```
#include "pch.h" // Підключення пре-компільованого заголовка. Зазвичай використовується для прискорення компіляції, містить часто використовувані заголовки.


#include "Form1.h" // Підключення заголовка форми Form1, яка містить весь графічний інтерфейс програми.


using namespace System; // Підключення простору імен .NET System, який містить базові типи (наприклад, String, Int32 тощо).

using namespace System::ComponentModel; // Простір імен для компонентів (не обов'язково прямо використовується тут, але може бути потрібен для WinForms).

using namespace System::Collections; // Простір імен для колекцій .NET (ArrayList, Hashtable тощо).

using namespace System::Windows::Forms; // Основний простір імен для роботи з Windows Forms – інтерфейсом користувача.

using namespace System::Data; // Простір імен для роботи з даними, як-от DataSet, DataTable тощо (може бути не використаний прямо).

using namespace System::Drawing; // Простір імен для роботи з графікою – кольорами, шрифтами, формами тощо.

using namespace System::IO; // Простір імен для роботи з файлами та потоками введення/виведення.


[STAThreadAttribute] // Атрибут, що вказує на використання одного потоку (Single Threaded Apartment), обов'язковий для коректної роботи WinForms (особливо з елементами інтерфейсу, як-от діалоги).


int main(array<String^>^ args) // Точка входу в програму. Приймає масив аргументів командного рядка (якщо є).

{

    Application::EnableVisualStyles(); // Увімкнення стилів Windows для сучасного вигляду кнопок, форм та інших елементів.
```

```
    Application::SetCompatibleTextRenderingDefault(false);    //    Встановлює    режим рендерингу тексту. false означає використання GDI+ для відображення тексту (краще для локалізації та нових шрифтів).


    CarDistanceApp::Form1 form; // Створення екземпляру головної форми Form1 з простору імен CarDistanceApp.


    Application::Run(% form); // Запускає форму як головне вікно програми. % означає передачу посилання на form (tracking reference у C++/CLI).

    return 0; // Повернення коду завершення.

}


#pragma once // Гарантує, що заголовочний файл буде включений лише один раз при компіляції

#include <cmath>          // Підключення математичних функцій (хоча не використовується напряму)
#include <fstream>         // Для роботи з файлами (не використовується безпосередньо, але можливо для розширення)

namespace CarDistanceApp { // Оголошення простору імен для уникнення конфліктів назв

      using namespace System; // Імпортування системного простору імен .NET
      using namespace System::ComponentModel; // Для компонентної моделі (може бути потрібне для форми)
      using namespace System::Collections;      // Колекції .NET (не використовується тут явно)
      using namespace System::Windows::Forms;  // Основні класи для створення WinForms GUI
      using namespace System::Data;          // Для роботи з даними (не використовується)
      using namespace System::Drawing;          // Для роботи з графікою/координатами в UI
      using namespace System::IO;               // Для читання/запису у файли

      public ref class Form1 : public System::Windows::Forms::Form // Оголошення головної форми, яка наслідує від Form
      {
      public:
            Form1(void) // Конструктор форми
            {
                  InitializeComponent(); // Ініціалізація UI-компонентів
            }

      protected:
```

```
        ~Form1() // Деструктор форми
        {
            if (components) // Перевірка, чи існують компоненти
            {
                delete components; // Звільнення ресурсів
            }
        }

    private:
        // Оголошення елементів інтерфейсу
        Label^ labelV1;          // Мітка для введення швидкості V1
        Label^ labelV2;          // Мітка для V2
        Label^ labelS;           // Мітка для початкової відстані S
        Label^ labelT;           // Мітка для часу T
        TextBox^ txtV1;          // Поле введення V1
        TextBox^ txtV2;          // Поле введення V2
        TextBox^ txtS;           // Поле введення відстані
        TextBox^ txtT;           // Поле введення часу
        Button^ btnCalculate;    // Кнопка для обчислення
        TextBox^ txtResult;      // Поле виводу результату
        Button^ btnLoadFile;     // Кнопка для завантаження даних з файлу
        Button^ btnSaveFile;     // Кнопка для збереження результату у файл
        OpenFileDialog^ openFileDialog; // Діалог вибору файлу для читання
        SaveFileDialog^ saveFileDialog; // Діалог вибору файлу для збереження
        System::ComponentModel::IContainer^   components;   //   Контейнер   для
компонентів

#pragma region Windows Form Designer generated code
        void InitializeComponent(void) // Метод для ініціалізації UI-компонентів
        {
            // Ініціалізація міток
            this->labelV1 = gcnew Label();
            this->labelV2 = gcnew Label();
            this->labelS = gcnew Label();
            this->labelT = gcnew Label();

            // Ініціалізація текстбоксів
            this->txtV1 = gcnew TextBox();
            this->txtV2 = gcnew TextBox();
            this->txtS = gcnew TextBox();
            this->txtT = gcnew TextBox();

            // Ініціалізація кнопок
            this->btnCalculate = gcnew Button();
            this->btnLoadFile = gcnew Button();
            this->btnSaveFile = gcnew Button();

            // Ініціалізація поля результату
            this->txtResult = gcnew TextBox();

            // Діалоги для роботи з файлами
```

```
                this->openFileDialog = gcnew OpenFileDialog();
                this->saveFileDialog = gcnew SaveFileDialog();

                this->SuspendLayout();  //  Призупинення  розмітки  для  швидшої
ініціалізації

                // ---- Налаштування властивостей міток ----
                this->labelV1->Text = L"V1 (км/год):";
                this->labelV1->Location = Point(20, 20);
                this->labelV1->Size = Drawing::Size(80, 20);

                this->labelV2->Text = L"V2 (км/год):";
                this->labelV2->Location = Point(20, 60);
                this->labelV2->Size = Drawing::Size(80, 20);

                this->labelS->Text = L"S (км):";
                this->labelS->Location = Point(20, 100);
                this->labelS->Size = Drawing::Size(80, 20);

                this->labelT->Text = L"T (год):";
                this->labelT->Location = Point(20, 140);
                this->labelT->Size = Drawing::Size(80, 20);

                // ---- Налаштування текстбоксів ----
                this->txtV1->Location = Point(110, 18);
                this->txtV1->Size = Drawing::Size(150, 25);

                this->txtV2->Location = Point(110, 58);
                this->txtV2->Size = Drawing::Size(150, 25);

                this->txtS->Location = Point(110, 98);
                this->txtS->Size = Drawing::Size(150, 25);

                this->txtT->Location = Point(110, 138);
                this->txtT->Size = Drawing::Size(150, 25);

                // ---- Кнопки ----
                this->btnCalculate->Text = L"Обчислити";
                this->btnCalculate->Location = Point(20, 180);
                this->btnCalculate->Size = Drawing::Size(90, 30);
                this->btnCalculate->Click     +=     gcnew     EventHandler(this,
&Form1::btnCalculate_Click); // Прив'язка події натискання
                this->btnLoadFile->Text = L"Зчитати з файлу";
                this->btnLoadFile->Location = Point(130, 180);
                this->btnLoadFile->Size = Drawing::Size(110, 30);
                this->btnLoadFile->Click     +=     gcnew     EventHandler(this,
&Form1::btnLoadFile_Click);
                this->btnSaveFile->Text = L"Зберегти у файл";
                this->btnSaveFile->Location = Point(260, 180);
                this->btnSaveFile->Size = Drawing::Size(110, 30);
```

```
                this->btnSaveFile->Click      +=      gcnew      EventHandler(this,
&Form1::btnSaveFile_Click);

                // ---- Поле результату ----
                this->txtResult->Location = Point(20, 220);
                this->txtResult->Size = Drawing::Size(350, 60);
                this->txtResult->Multiline = true;
                this->txtResult->ReadOnly = true;
                this->txtResult->ScrollBars = ScrollBars::Vertical;

                // ---- Властивості форми ----
                this->ClientSize = Drawing::Size(400, 300); // Розмір вікна
                this->Controls->Add(this->labelV1);
                this->Controls->Add(this->labelV2);
                this->Controls->Add(this->labelS);
                this->Controls->Add(this->labelT);
                this->Controls->Add(this->txtV1);
                this->Controls->Add(this->txtV2);
                this->Controls->Add(this->txtS);
                this->Controls->Add(this->txtT);
                this->Controls->Add(this->btnCalculate);
                this->Controls->Add(this->btnLoadFile);
                this->Controls->Add(this->btnSaveFile);
                this->Controls->Add(this->txtResult);

                this->Text = L"Обчислення відстані між автомобілями"; // Заголовок
вікна
                this->FormBorderStyle                                                =
System::Windows::Forms::FormBorderStyle::FixedDialog; // Фіксований розмір
                this->MaximizeBox = false; // Заборонити розгортання вікна

                this->ResumeLayout(false); // Відновити автоматичну розмітку
                this->PerformLayout();     // Примусово виконати розмітку
            }
#pragma endregion

      private:      System::Void      btnCalculate_Click(System::Object^      sender,
System::EventArgs^ e) {
            try {
                  // Зчитування значень з текстбоксів
                  double V1 = Double::Parse(txtV1->Text);
                  double V2 = Double::Parse(txtV2->Text);
                  double S = Double::Parse(txtS->Text);
                  double T = Double::Parse(txtT->Text);

                  // Перевірка на додатність
                  if (V1 < 0 || V2 < 0 || S < 0 || T < 0) {
                        MessageBox::Show("Швидкості, відстань і час повинні бути
додатними       числами.",       "Помилка       введення",       MessageBoxButtons::OK,
MessageBoxIcon::Warning);
                        return;
```

```
                }

                // Обчислення відстані між автомобілями через T годин
                double distance = Math::Abs(S - T * (V1 + V2));

                // Вивід результату у текстбокс
                txtResult->Text = "Відстань між автомобілями через " +
T.ToString("F2") + " год: " + distance.ToString("F2") + " км";
            }
            catch (FormatException^) {
                // Помилка при перетворенні введених значень
                MessageBox::Show("Будь ласка, введіть коректні числові значення.",
"Помилка формату", MessageBoxButtons::OK, MessageBoxIcon::Error);
            }
            catch (Exception^ ex) {
                // Інші виключення
                MessageBox::Show("Сталася помилка: " + ex->Message, "Помилка",
MessageBoxButtons::OK, MessageBoxIcon::Error);
            }
      }

      private:      System::Void      btnLoadFile_Click(System::Object^      sender,
System::EventArgs^ e) {
            if                  (openFileDialog->ShowDialog()                  ==
System::Windows::Forms::DialogResult::OK) {
                try {
                    // Зчитування всіх рядків з вибраного файлу
                    array<String^>^ lines = File::ReadAllLines(openFileDialog-
>FileName);
                    if (lines->Length >= 4) {
                        // Присвоєння значень текстбоксам
                        txtV1->Text = lines[0];
                        txtV2->Text = lines[1];
                        txtS->Text = lines[2];
                        txtT->Text = lines[3];
                    }
                    else {
                        // Файл має недостатньо рядків
                        MessageBox::Show("Файл повинен містити щонайменше 4
рядки: V1, V2, S, T.", "Помилка файлу", MessageBoxButtons::OK, MessageBoxIcon::Warning);
                    }
                }
                catch (Exception^ ex) {
                    // Помилка читання файлу
                    MessageBox::Show("Не вдалося прочитати файл: " + ex->Message,
"Помилка файлу", MessageBoxButtons::OK, MessageBoxIcon::Error);
                }
            }
      }
```

```
      private:       System::Void      btnSaveFile_Click(System::Object^       sender,
System::EventArgs^ e) {
             if                    (saveFileDialog->ShowDialog()                    ==
System::Windows::Forms::DialogResult::OK) {
                   try {
                         // Запис результату у вибраний файл
                         File::WriteAllText(saveFileDialog->FileName,       txtResult-
>Text);
                         MessageBox::Show("Результат       успішно       збережено.",
"Збереження", MessageBoxButtons::OK, MessageBoxIcon::Information);
                   }
                   catch (Exception^ ex) {
                         // Помилка при збереженні
                         MessageBox::Show("Не вдалося зберегти файл: " + ex->Message,
"Помилка файлу", MessageBoxButtons::OK, MessageBoxIcon::Error);
                   }
             }
      }
      };
}
```

ДОДАТОК Б
Скрін-шоти вікна виконання програми

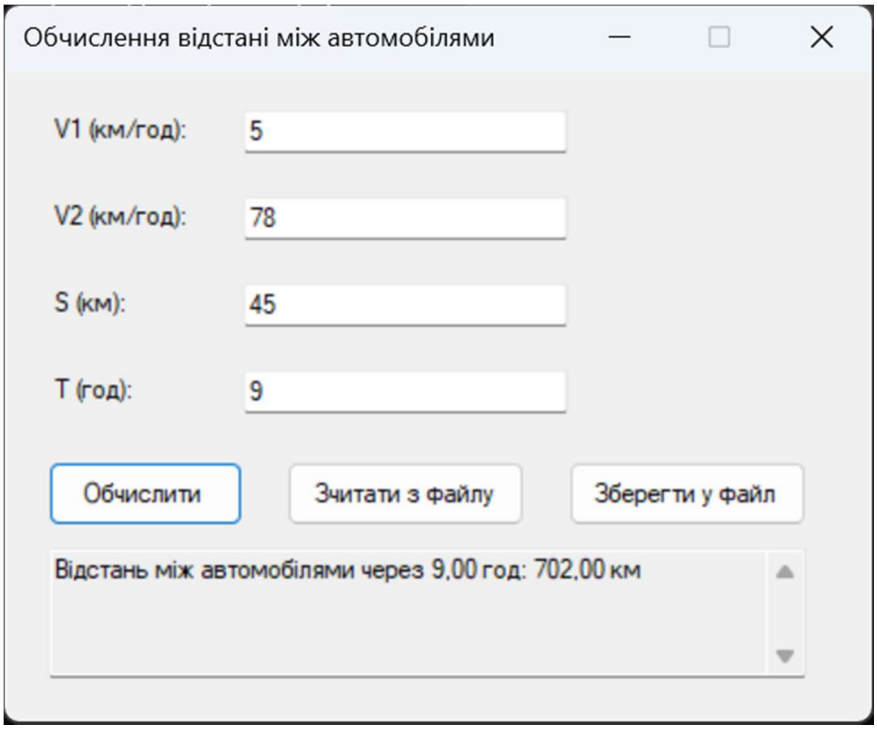


Рисунок Б.1 – Екран виконання програми для вирішення завдання 2

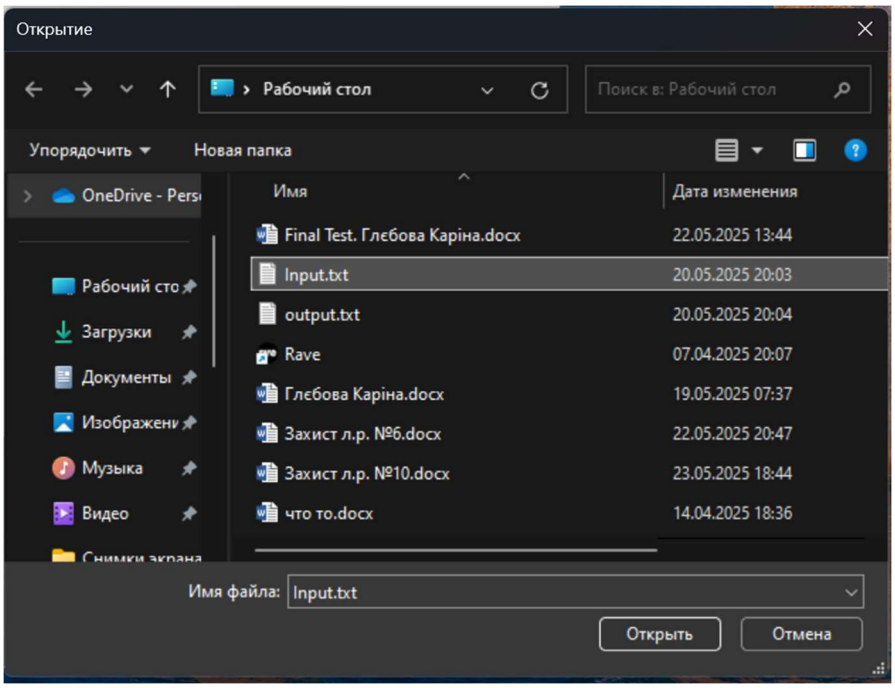


Рисунок Б.2 – Екран виконання програми для вирішення завдання 2

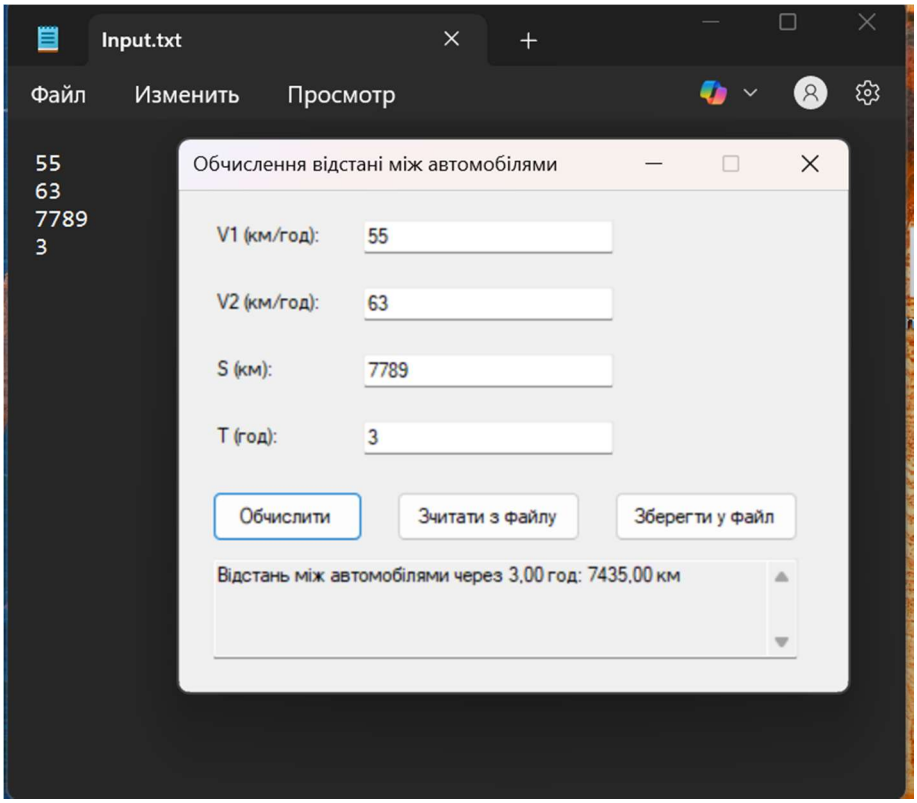


Рисунок Б.3 – Екран виконання програми для вирішення завдання 2

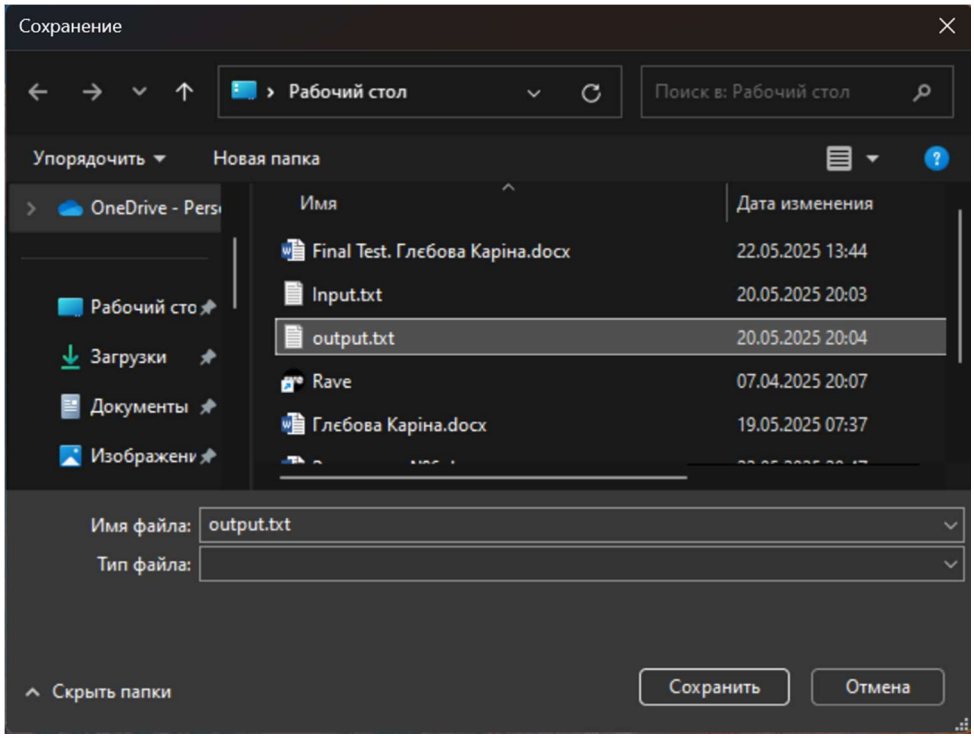


Рисунок Б.4 – Екран виконання програми для вирішення завдання 2

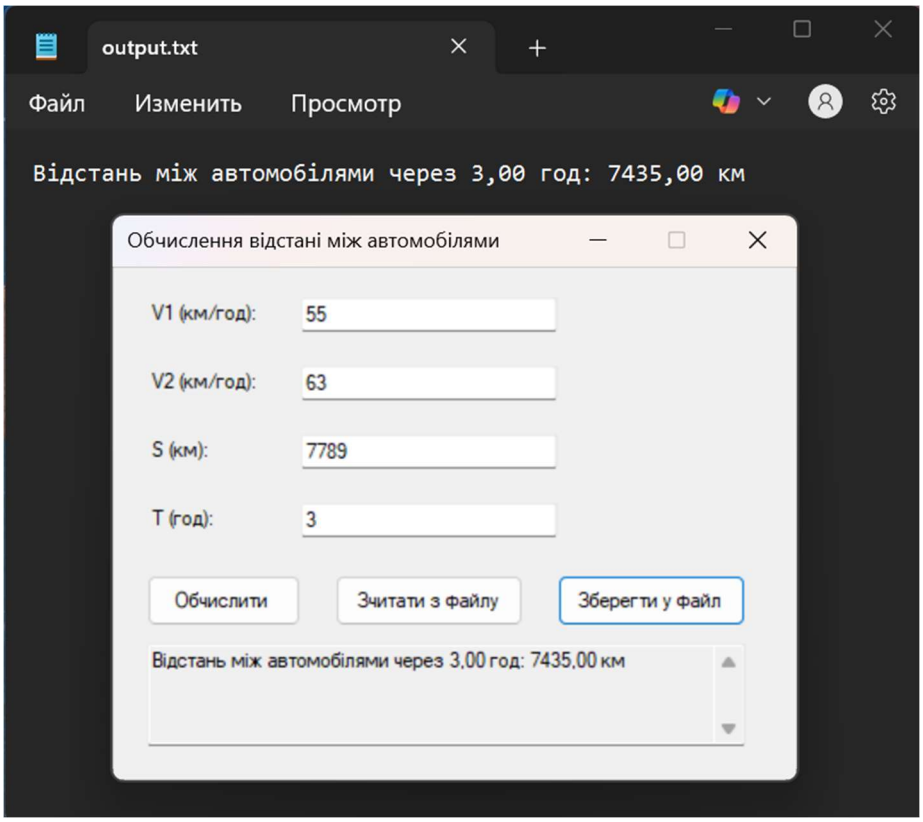


Рисунок Б.5 – Екран виконання програми для вирішення завдання 2

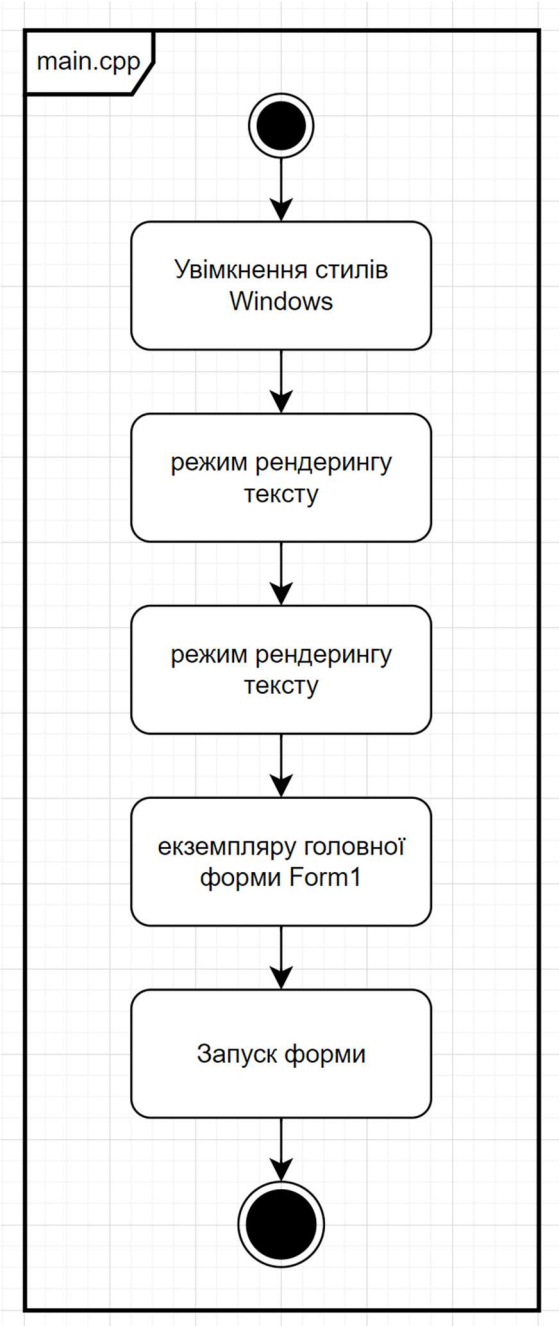


Рисунок Б.6 – Діаграма для головного меню

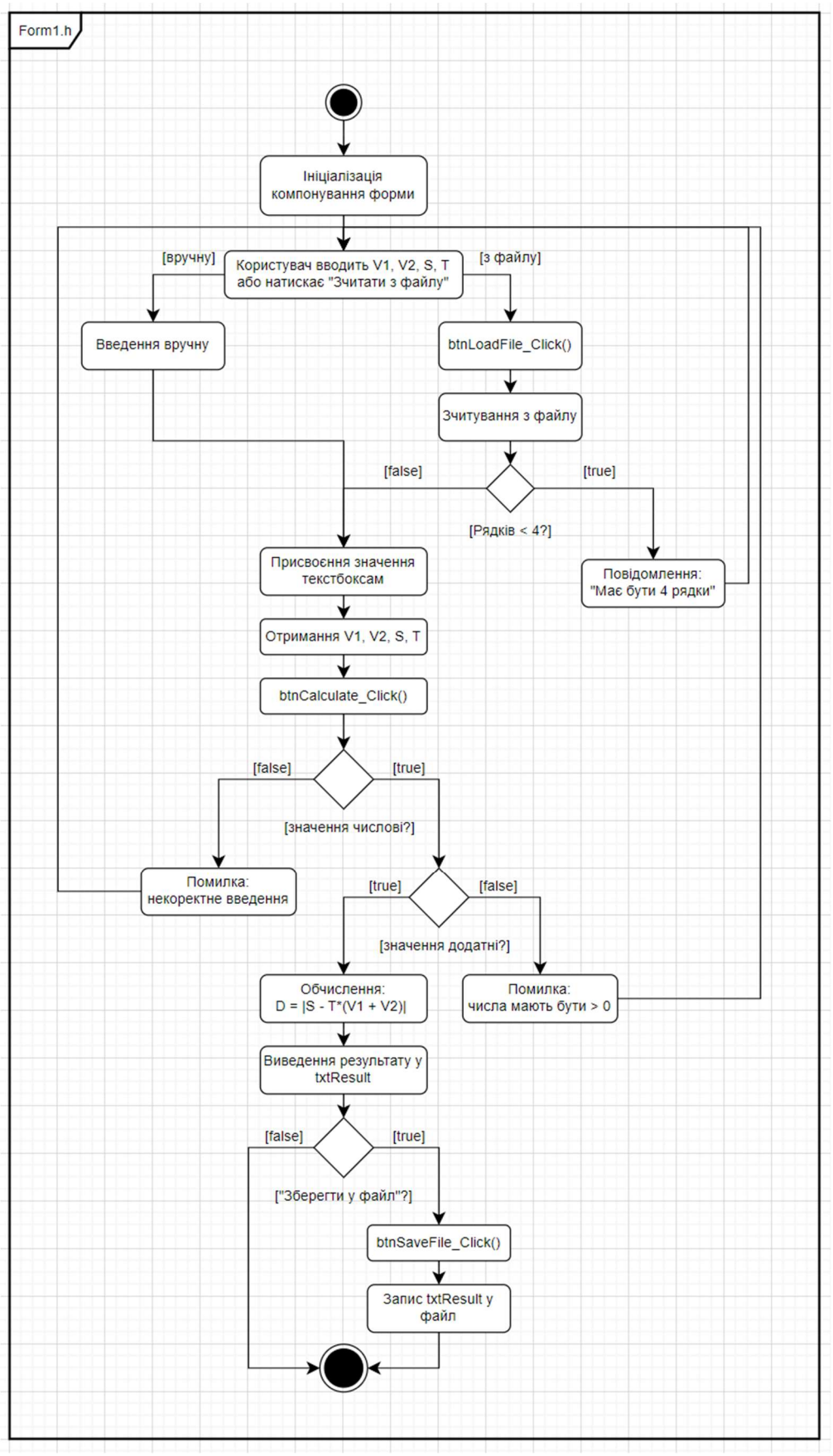


Рисунок Б.7 – Діаграма для завдання 1