**Лабораторна робота №11**. **Створення багатовіконних застосунків. Частина 1.**

**Виконавець:** Лазоренко М.Д.

**Група виконавця:** КН-922в

**Мета роботи:** набути навички у створенні багатовіконних застосунків.

**Індивідуальні завдання**

**Варіант №9**

**Завдання 1**

Використовуючи граф-схему власного варіанту, створити багатовіконний застосунок виклику відповідних форм.

|  |  |
| --- | --- |
| № варіанту | Граф схема |
| 1 | 2 |
| 9. |  |

Назви форм повинні частково відповідати назвам з вершин графів (наприклад Form1, Form2, ... чи FormA, FormB, FormC, ...). Форми мають викликатися в модальному режимі (ShowDialog) та мати різнокольорові характеристики фону та тексту. Виклики форм виконувати за допомогою звичайних кнопок, кнопок з панелей ToolStrip, a6o за допомогою команд з MenuStrip чи ContextMenuStrip.

**Текст програми:**

***Form1.cs:***

using System;

using System.Windows.Forms;

using task01;

namespace task01

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonForm2\_Click(object sender, EventArgs e)

{

Form2 form2 = new Form2();

form2.ShowDialog();

}

private void buttonForm3\_Click(object sender, EventArgs e)

{

Form3 form3 = new Form3();

form3.ShowDialog();

}

private void buttonForm4\_Click(object sender, EventArgs e)

{

Form4 form4 = new Form4();

form4.ShowDialog();

}

}

}

***Form2.cs:***

using System;

using System.Windows.Forms;

using task01;

namespace task01

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void buttonForm5\_Click(object sender, EventArgs e)

{

Form5 form5 = new Form5();

form5.ShowDialog();

}

private void buttonForm6\_Click(object sender, EventArgs e)

{

Form6 form6 = new Form6();

form6.ShowDialog();

}

private void buttonForm7\_Click(object sender, EventArgs e)

{

Form7 form7 = new Form7();

form7.ShowDialog();

}

}

}

***Form3.cs:***

using System;

using System.Windows.Forms;

namespace task01

{

public partial class Form3 : Form

{

public Form3()

{

InitializeComponent();

}

}

}

***Form4.cs:***

using System;

using System.Windows.Forms;

using task01;

namespace task01

{

public partial class Form4 : Form

{

public Form4()

{

InitializeComponent();

}

private void contextMenuItemForm8\_Click(object sender, EventArgs e)

{

Form8 form8 = new Form8();

form8.ShowDialog();

}

}

}

***Form5.cs:***

using System;

using System.Windows.Forms;

namespace task01

{

public partial class Form5 : Form

{

public Form5()

{

InitializeComponent();

}

}

}

***Form6.cs:***

using System;

using System.Windows.Forms;

namespace task01

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

}

}

***Form7.cs:***

using System;

using System.Windows.Forms;

namespace task01

{

public partial class Form7 : Form

{

public Form7()

{

InitializeComponent();

}

}

}

***Form8.cs:***

using System;

using System.Windows.Forms;

namespace task01

{

public partial class Form8 : Form

{

public Form8()

{

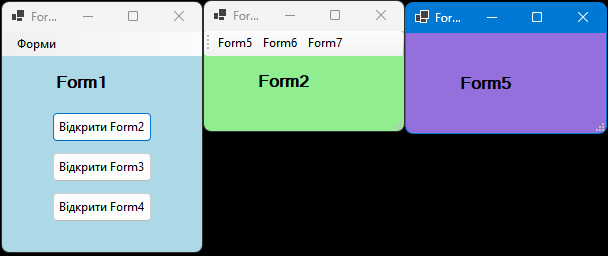
InitializeComponent();

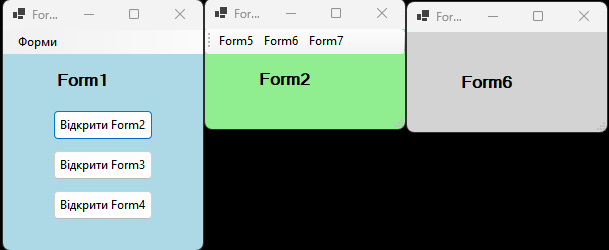
}

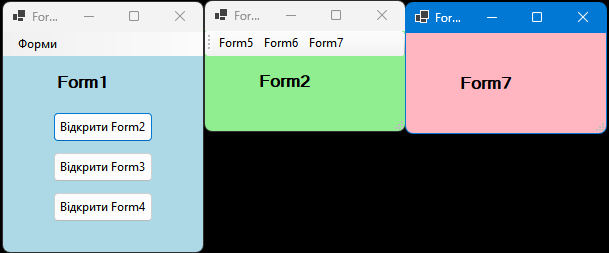
}

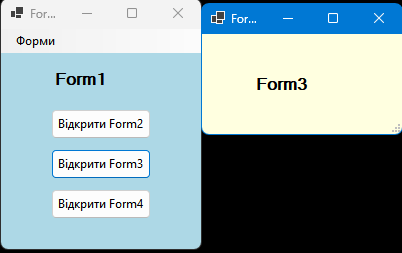
}

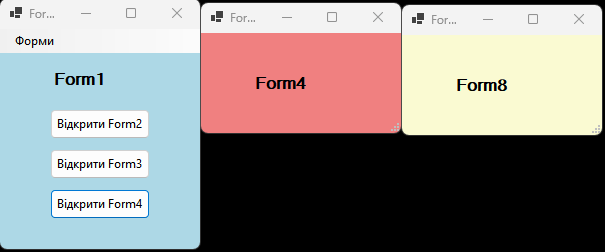
**Результат роботи програми:**











Ця програма на C# створює багатовіконний графічний інтерфейс користувача, який дозволяє переходити між різними формами, виконуючи певні функції або відображаючи різні компоненти.

**Основні етапи виконання програми:**

1. **Головне вікно програми (Form1):**
   * **Елементи інтерфейсу:**  
     Головна форма містить кнопки для відкриття інших форм: Form2, Form3, Form4.
   * **Обробка подій:**  
     При натисканні на відповідну кнопку відображається нове модальне вікно з потрібною формою.
2. **Форма Form2:**
   * **Елементи інтерфейсу:**  
     Містить кнопки для переходу до форм Form5, Form6, Form7.
   * **Обробка подій:**  
     При натисканні на кнопку відповідна форма відкривається як діалогове вікно.
3. **Форма Form3:**
   * **Призначення:**  
     Ця форма служить для відображення окремого функціоналу чи елементів.
   * **Особливості:**  
     Ніяких додаткових подій у коді цієї форми не реалізовано.
4. **Форма Form4:**
   * **Елементи інтерфейсу:**  
     Містить контекстне меню, за допомогою якого можна викликати форму Form8.
   * **Обробка подій:**  
     Користувач може вибрати пункт контекстного меню, після чого відкривається форма Form8.
5. **Форми Form5, Form6, Form7:**
   * **Призначення:**  
     Містять індивідуальний функціонал або відображають потрібну інформацію.
   * **Особливості:**  
     Реалізовано базову ініціалізацію без додаткових функцій.
6. **Форма Form8:**
   * **Призначення:**  
     Використовується для розширеного функціоналу, що доступний із форми Form4 через контекстне меню.

**Завдання 2\***

Модифікувати минуле завдання, додавши до застосунку можливість обмінюватися інформаціею між формами наступним чином:

* додати до кожної форми компонент(и) label, в яких мае відображатися назва та інформація з форми, яка викликала a6o повернула управління до цісї форми;
* додати до кожної форми компонент textbox, де е можливість друкувати інформацію для викликаючої, a6o повертаючої управління форми;
* під час виклику чи повертання з форм відповідні поля та мітки мають заповнюватися відповідної інформаціею з форм.

**Текст програми:**

***Form1.cs:***

using System;

using System.Windows.Forms;

using task02;

namespace task02

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

labelCallerInfo.Text = "Немає попередньої форми";

}

private void buttonForm2\_Click(object sender, EventArgs e)

{

Form2 form2 = new Form2();

// Передача інформації до Form2

form2.CallerName = "Form1";

form2.CallerInfo = textBoxInfo.Text;

form2.ShowDialog();

// Отримання інформації від Form2

string infoFromForm2 = form2.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form2: {infoFromForm2}";

}

private void buttonForm3\_Click(object sender, EventArgs e)

{

Form3 form3 = new Form3();

form3.CallerName = "Form1";

form3.CallerInfo = textBoxInfo.Text;

form3.ShowDialog();

string infoFromForm3 = form3.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form3: {infoFromForm3}";

}

private void buttonForm4\_Click(object sender, EventArgs e)

{

Form4 form4 = new Form4();

form4.CallerName = "Form1";

form4.CallerInfo = textBoxInfo.Text;

form4.ShowDialog();

string infoFromForm4 = form4.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form4: {infoFromForm4}";

}

}

}

***Form2.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form2 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form2\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

private void toolStripButtonForm5\_Click(object sender, EventArgs e)

{

Form5 form5 = new Form5();

form5.CallerName = "Form2";

form5.CallerInfo = textBoxInfo.Text;

form5.ShowDialog();

string infoFromForm5 = form5.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form5: {infoFromForm5}";

}

private void toolStripButtonForm6\_Click(object sender, EventArgs e)

{

Form6 form6 = new Form6();

form6.CallerName = "Form2";

form6.CallerInfo = textBoxInfo.Text;

form6.ShowDialog();

string infoFromForm6 = form6.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form6: {infoFromForm6}";

}

private void toolStripButtonForm7\_Click(object sender, EventArgs e)

{

Form7 form7 = new Form7();

form7.CallerName = "Form2";

form7.CallerInfo = textBoxInfo.Text;

form7.ShowDialog();

string infoFromForm7 = form7.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form7: {infoFromForm7}";

}

}

}

***Form3.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form3 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form3()

{

InitializeComponent();

}

private void Form3\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form3\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

}

}

***Form4.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form4 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form4()

{

InitializeComponent();

}

private void Form4\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form4\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

private void contextMenuItemForm8\_Click(object sender, EventArgs e)

{

Form8 form8 = new Form8();

form8.CallerName = "Form4";

form8.CallerInfo = textBoxInfo.Text;

form8.ShowDialog();

string infoFromForm8 = form8.ReturnInfo;

labelCallerInfo.Text = $"Повернулося з Form8: {infoFromForm8}";

}

}

}

***Form5.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form5 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form5()

{

InitializeComponent();

}

private void Form5\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form5\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

}

}

***Form6.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form6 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form6()

{

InitializeComponent();

}

private void Form6\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form6\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

}

}

***Form7.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form7 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form7()

{

InitializeComponent();

}

private void Form7\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form7\_FormClosing(object sender, FormClosingEventArgs e)

{

ReturnInfo = textBoxInfo.Text;

}

}

}

***Form8.cs:***

using System;

using System.Windows.Forms;

namespace task02

{

public partial class Form8 : Form

{

public string CallerName { get; set; }

public string CallerInfo { get; set; }

public string ReturnInfo { get; set; }

public Form8()

{

InitializeComponent();

}

private void Form8\_Load(object sender, EventArgs e)

{

labelCallerInfo.Text = $"Викликано з {CallerName}: {CallerInfo}";

}

private void Form8\_FormClosing(object sender, FormClosingEventArgs e)

{

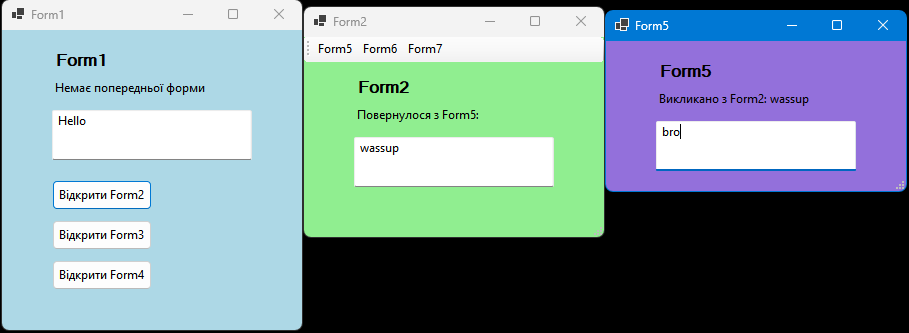
ReturnInfo = textBoxInfo.Text;

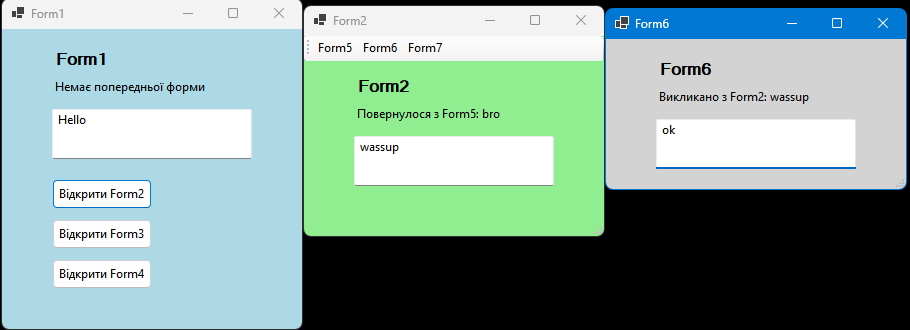
}

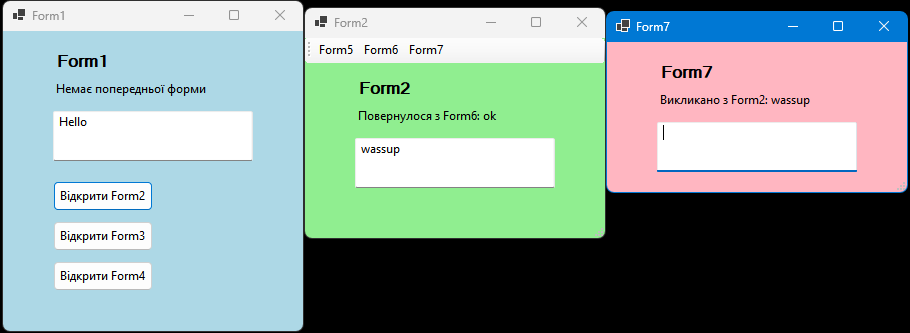
}

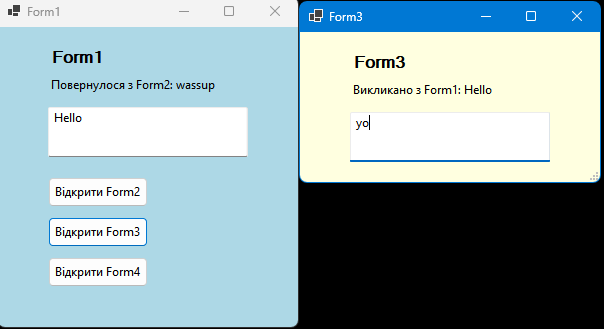
}

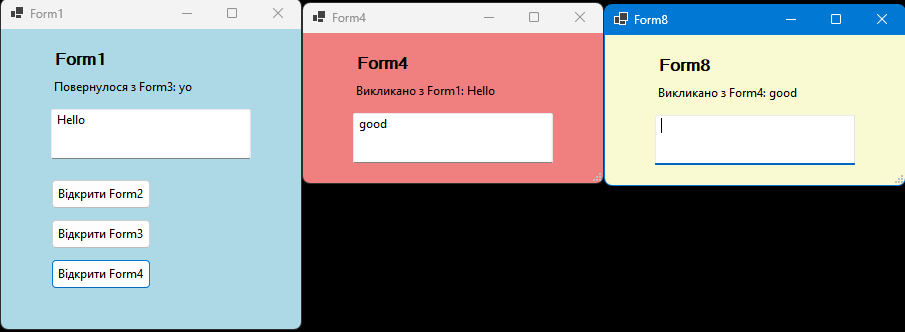
**Результат роботи програми:**

****









Програма є багатовіконним застосунком на Windows Forms, який демонструє взаємодію між формами та передачу даних між ними. Вона складається з головної форми (**Form1**) і кількох дочірніх форм (**Form2**, **Form3**, **Form4**, **Form5**, **Form6**, **Form7**, **Form8**). Основна мета програми — передавати інформацію між формами, обробляти її та відображати результат на викликаючій формі.

**Основна логіка роботи програми:**

1. **Form1**:
   * Відображає текстове поле для введення даних користувачем.
   * Має три кнопки для відкриття дочірніх форм (**Form2**, **Form3**, **Form4**).
   * Передає інформацію з текстового поля на дочірню форму.
   * Отримує дані, які користувач ввів або змінив у дочірній формі, і виводить їх на мітку **labelCallerInfo**.
2. **Form2, Form3, Form4**:
   * Отримують інформацію від **Form1**, включаючи ім'я викликаючої форми (**CallerName**) і текст (**CallerInfo**).
   * Відображають отриману інформацію на формі.
   * Перед закриттям повертають змінену інформацію у властивості **ReturnInfo**.
3. **Form5, Form6, Form7, Form8**:
   * Є дочірніми формами для **Form2** або **Form4**.
   * Взаємодіють аналогічно з викликаючою формою, дозволяючи передавати і повертати дані.

**Завдання 3**

За завданням власного варіанту

|  |  |
| --- | --- |
| № варіанту | Задача |
| 1 | 2 |
| 9. | Дани велика та мала піввісь еліпсу, розрахувати його периметр та площу. |

Створити чотири варіанти застосунку (a6o в окремих проектах, a6o в одному проекті на окремих формах, які викликаються з якоїсь головной форми за допомогою кнопок чи команд меню) для вирішення задачі:

* вхідні дані i результати розрахунків відображаються на одній формі;
* вхідні дані отримуються з окремої форми;
* результати розрахунків виводяться в окрему форму;
* вхідні дані та результати розрахунків відображаються на окремих формах.

Зауваження: під час розрахунків мас перевірятися коректність даних, i разі некоректного введення мае видаватися відповідне сповіщення користувачу.

**Текст програми:**

**1. Одна форма для вхідних даних i результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task03\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonCalculate\_Click(object sender, EventArgs e)

{

if (double.TryParse(textBoxA.Text, out double a) &&

double.TryParse(textBoxB.Text, out double b) &&

a > 0 && b > 0)

{

double area = Math.PI \* a \* b;

double perimeter = CalculatePerimeter(a, b);

labelAreaResult.Text = area.ToString("F2");

labelPerimeterResult.Text = perimeter.ToString("F2");

}

else

{

MessageBox.Show("Будь ласка, введіть коректні додатні числа для великих та малих півосей.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private double CalculatePerimeter(double a, double b)

{

// Використання другого наближення Рамануджана

double h = Math.Pow((a - b) / (a + b), 2);

double perimeter = Math.PI \* (a + b) \* (1 + (3 \* h) / (10 + Math.Sqrt(4 - 3 \* h)));

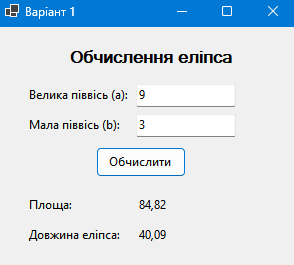
return perimeter;

}

}

}

**Результат роботи програми:**



**2. Окрема форма для вхідних даних.**

***Form1.cs:***

using System;

using System.Windows.Forms;

namespace task03\_2

{

public partial class Form1 : Form

{

private double a;

private double b;

public Form1()

{

InitializeComponent();

}

private void buttonInput\_Click(object sender, EventArgs e)

{

InputForm inputForm = new InputForm();

if (inputForm.ShowDialog() == DialogResult.OK)

{

a = inputForm.A;

b = inputForm.B;

double area = Math.PI \* a \* b;

double perimeter = CalculatePerimeter(a, b);

labelAreaResult.Text = area.ToString("F2");

labelPerimeterResult.Text = perimeter.ToString("F2");

}

}

private double CalculatePerimeter(double a, double b)

{

double h = Math.Pow((a - b) / (a + b), 2);

double perimeter = Math.PI \* (a + b) \* (1 + (3 \* h) / (10 + Math.Sqrt(4 - 3 \* h)));

return perimeter;

}

}

}

***InputForm.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task03\_2

{

public partial class InputForm : Form

{

public double A { get; private set; }

public double B { get; private set; }

public InputForm()

{

InitializeComponent();

}

private void buttonOK\_Click(object sender, EventArgs e)

{

if (double.TryParse(textBoxA.Text, out double a) &&

double.TryParse(textBoxB.Text, out double b) &&

a > 0 && b > 0)

{

A = a;

B = b;

this.DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Будь ласка, введіть коректні додатні числа для великих та малих півосей.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

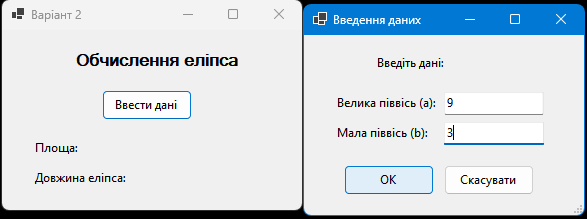
}

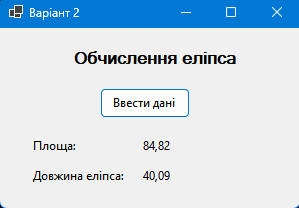
}

}

}

**Результат роботи програми:**





**3. Окрема форма для результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task03\_3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonCalculate\_Click(object sender, EventArgs e)

{

if (double.TryParse(textBoxA.Text, out double a) &&

double.TryParse(textBoxB.Text, out double b) &&

a > 0 && b > 0)

{

double area = Math.PI \* a \* b;

double perimeter = CalculatePerimeter(a, b);

ResultsForm resultsForm = new ResultsForm(area, perimeter);

resultsForm.ShowDialog();

}

else

{

MessageBox.Show("Будь ласка, введіть коректні додатні числа для великих та малих півосей.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private double CalculatePerimeter(double a, double b)

{

double h = Math.Pow((a - b) / (a + b), 2);

double perimeter = Math.PI \* (a + b) \* (1 + (3 \* h) / (10 + Math.Sqrt(4 - 3 \* h)));

return perimeter;

}

}

}

***ResultForm.cs:***

using System;

using System.Windows.Forms;

namespace task03\_3

{

public partial class ResultsForm : Form

{

public ResultsForm(double area, double perimeter)

{

InitializeComponent();

labelAreaResult.Text = area.ToString("F2");

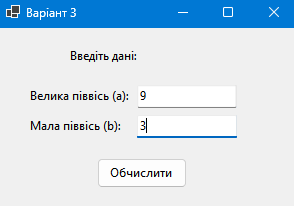
labelPerimeterResult.Text = perimeter.ToString("F2");

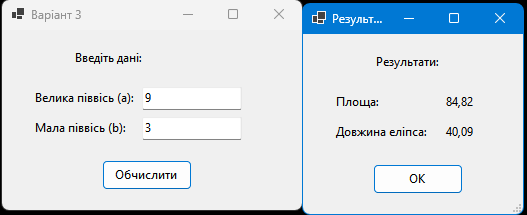
}

}

}

**Результат роботи програми:**

****

****

**4. Окремі форми вхідних даних та результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

namespace task03\_4

{

public partial class Form1 : Form

{

private double a;

private double b;

public Form1()

{

InitializeComponent();

}

private void buttonInput\_Click(object sender, EventArgs e)

{

InputForm inputForm = new InputForm();

if (inputForm.ShowDialog() == DialogResult.OK)

{

a = inputForm.A;

b = inputForm.B;

double area = Math.PI \* a \* b;

double perimeter = CalculatePerimeter(a, b);

ResultsForm resultsForm = new ResultsForm(area, perimeter);

resultsForm.ShowDialog();

}

}

private double CalculatePerimeter(double a, double b)

{

double h = Math.Pow((a - b) / (a + b), 2);

double perimeter = Math.PI \* (a + b) \* (1 + (3 \* h) / (10 + Math.Sqrt(4 - 3 \* h)));

return perimeter;

}

}

}

***InputForm.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task03\_4

{

public partial class InputForm : Form

{

public double A { get; private set; }

public double B { get; private set; }

public InputForm()

{

InitializeComponent();

}

private void buttonOK\_Click(object sender, EventArgs e)

{

if (double.TryParse(textBoxA.Text, out double a) &&

double.TryParse(textBoxB.Text, out double b) &&

a > 0 && b > 0)

{

A = a;

B = b;

this.DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Будь ласка, введіть коректні додатні числа для великих та малих півосей.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

***ResultsForm.cs:***

using System;

using System.Windows.Forms;

namespace task03\_4

{

public partial class ResultsForm : Form

{

public ResultsForm(double area, double perimeter)

{

InitializeComponent();

labelAreaResult.Text = area.ToString("F2");

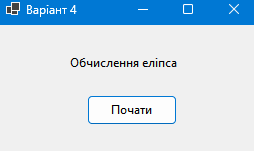
labelPerimeterResult.Text = perimeter.ToString("F2");

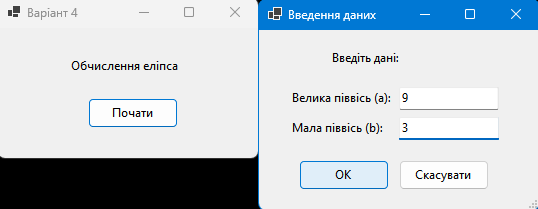
}

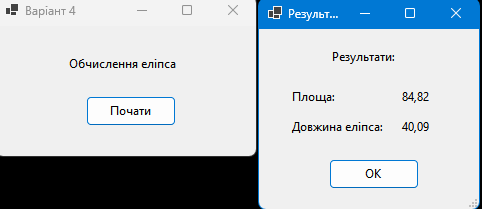
}

}

**Результат роботи програми:**



****

****

Програми розроблені для обчислення площі та периметра еліпса за заданими великими та малими півосями a та b. Використано різні підходи до організації форм введення даних і відображення результатів. Наведено чотири варіанти реалізації:

**1. Одна форма для введення даних і відображення результатів**

У цьому варіанті:

* Дані вводяться безпосередньо в основній формі (Form1).
* В результаті натискання кнопки Calculate виконується перевірка коректності введених значень.
* Обчислюється:
  + Площа S = π ⋅ a ⋅ b
  + Периметр за формулою

* Результати виводяться в елементи форми (Label).
* У разі помилок показується повідомлення про некоректний ввід.

**2. Окрема форма для введення даних**

У цьому варіанті:

* Дані вводяться в окремій формі InputForm.
* Основна форма (Form1) відкриває форму введення, отримує значення aaa та bbb і виконує обчислення.
* Результати відображаються безпосередньо в основній формі.

Ключові особливості:

* Усі обчислення виконуються в основній формі.
* Форма введення закривається після підтвердження даних.

**3. Окрема форма для відображення результатів**

У цьому варіанті:

* Введення даних виконується в основній формі (Form1).
* Результати обчислень передаються у форму ResultsForm, яка відкривається як діалогове вікно.
* У формі результатів відображається площа та периметр.

Переваги:

* Основна форма не перевантажена інформацією.
* Всі результати відображаються окремо, що зручно для користувача.

**4. Окремі форми для введення даних і результатів**

У цьому варіанті:

* Дані вводяться у формі InputForm.
* Основна форма (Form1) передає обчислені результати у форму ResultsForm, яка відкривається після введення даних.
* Реалізовано максимальну ізоляцію введення даних і відображення результатів.

Цей варіант найбільш зручний для сценаріїв, де передбачена чітка розділеність етапів роботи з програмою.

**Завдання 4**

За завданням власного варіанту

|  |  |
| --- | --- |
| № варіанту | Вираз |
| 1 | 2 |
| 9. |  |

Створити чотири варіанти застосунку (a6o в окремих проектах, a6o в одному проекті на окремих формах, які викликаються з якоїсь головной форми за допомогою кнопок чи команд меню) для розрахунку виразу:

* вхідні дані i результати розрахунків виразу відображаються на одній формі;
* вхідні дані отримуються з окремої форми;
* результати розрахунків виводяться в окрему форму;
* вхідні дані та результати розрахунків відображаються на окремих формах.

Зауваження: під час розрахунків мас перевірятися коректність даних, i разі некоректного введення має видаватися відповідне сповіщення користувачу.

**Текст програми:**

**1. Одна форма для вхідних даних i результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task04\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonCalculate\_Click(object sender, EventArgs e)

{

if (int.TryParse(textBoxX.Text, out int x) &&

int.TryParse(textBoxY.Text, out int y) &&

double.TryParse(textBoxZ.Text, out double z))

{

int N = x + y;

if (N < 1)

{

MessageBox.Show("Сума x та y повинна бути цілим числом більше або рівним 1.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

try

{

double product = CalculateProduct(x, y, z);

labelResult.Text = $"Результат: {product:F4}";

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка обчислення", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show("Будь ласка, введіть коректні числові значення для x, y та z.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private double CalculateProduct(int x, int y, double z)

{

int N = x + y;

double product = 1.0;

for (int i = 1; i <= N; i++)

{

double numerator = 2 \* i - z \* x;

double denominator = Math.Pow(x, 3) - i + Math.Pow(i, 2);

if (denominator == 0)

throw new DivideByZeroException($"Знаменник дорівнює нулю при i = {i}.");

double fraction = numerator / denominator;

if (fraction < 0)

throw new ArithmeticException($"Підкореневий вираз від'ємний при i = {i}.");

double term = Math.Sqrt(fraction);

product \*= term;

}

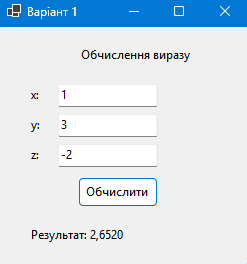
return product;

}

}

}

**Результат роботи програми:**



**2. Окрема форма для вхідних даних.**

***Form1.cs:***

using System;

using System.Windows.Forms;

namespace task04\_2

{

public partial class Form1 : Form

{

private int x;

private int y;

private double z;

public Form1()

{

InitializeComponent();

}

private void buttonInput\_Click(object sender, EventArgs e)

{

InputForm inputForm = new InputForm();

if (inputForm.ShowDialog() == DialogResult.OK)

{

x = inputForm.X;

y = inputForm.Y;

z = inputForm.Z;

try

{

double product = CalculateProduct(x, y, z);

labelResult.Text = $"Результат: {product:F4}";

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка обчислення", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private double CalculateProduct(int x, int y, double z)

{

int N = x + y;

if (N < 1)

throw new ArgumentException("Сума x та y повинна бути цілим числом більше або рівним 1.");

double product = 1.0;

for (int i = 1; i <= N; i++)

{

double numerator = 2 \* i - z \* x;

double denominator = Math.Pow(x, 3) - i + Math.Pow(i, 2);

if (denominator == 0)

throw new DivideByZeroException($"Знаменник дорівнює нулю при i = {i}.");

double fraction = numerator / denominator;

if (fraction < 0)

throw new ArithmeticException($"Підкореневий вираз від'ємний при i = {i}.");

double term = Math.Sqrt(fraction);

product \*= term;

}

return product;

}

}

}

***InputForm.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task04\_2

{

public partial class InputForm : Form

{

public int X { get; private set; }

public int Y { get; private set; }

public double Z { get; private set; }

public InputForm()

{

InitializeComponent();

}

private void buttonOK\_Click(object sender, EventArgs e)

{

if (int.TryParse(textBoxX.Text, out int x) &&

int.TryParse(textBoxY.Text, out int y) &&

double.TryParse(textBoxZ.Text, out double z))

{

X = x;

Y = y;

Z = z;

this.DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Будь ласка, введіть коректні числові значення для x, y та z.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

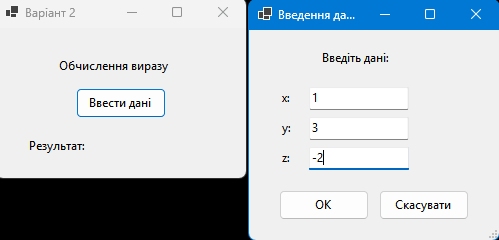
}

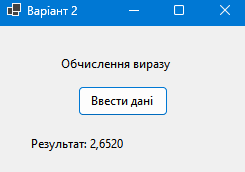
}

}

}

**Результат роботи програми:**





**3. Окрема форма для результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task04\_3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonCalculate\_Click(object sender, EventArgs e)

{

if (int.TryParse(textBoxX.Text, out int x) &&

int.TryParse(textBoxY.Text, out int y) &&

double.TryParse(textBoxZ.Text, out double z))

{

try

{

double product = CalculateProduct(x, y, z);

ResultsForm resultsForm = new ResultsForm(product);

resultsForm.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка обчислення", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show("Будь ласка, введіть коректні числові значення для x, y та z.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private double CalculateProduct(int x, int y, double z)

{

int N = x + y;

if (N < 1)

throw new ArgumentException("Сума x та y повинна бути цілим числом більше або рівним 1.");

double product = 1.0;

for (int i = 1; i <= N; i++)

{

double numerator = 2 \* i - z \* x;

double denominator = Math.Pow(x, 3) - i + Math.Pow(i, 2);

if (denominator == 0)

throw new DivideByZeroException($"Знаменник дорівнює нулю при i = {i}.");

double fraction = numerator / denominator;

if (fraction < 0)

throw new ArithmeticException($"Підкореневий вираз від'ємний при i = {i}.");

double term = Math.Sqrt(fraction);

product \*= term;

}

return product;

}

}

}

***ResultsForm.cs:***

using System;

using System.Windows.Forms;

namespace task04\_3

{

public partial class ResultsForm : Form

{

public ResultsForm(double result)

{

InitializeComponent();

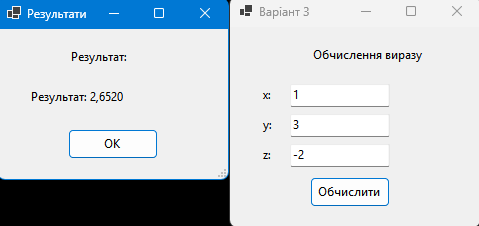
labelResult.Text = $"Результат: {result:F4}";

}

}

}

**Результат роботи програми:**



**4. Окремі форми вхідних даних та результатів.**

***Form1.cs:***

using System;

using System.Windows.Forms;

namespace task04\_4

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void buttonStart\_Click(object sender, EventArgs e)

{

InputForm inputForm = new InputForm();

if (inputForm.ShowDialog() == DialogResult.OK)

{

int x = inputForm.X;

int y = inputForm.Y;

double z = inputForm.Z;

try

{

double product = CalculateProduct(x, y, z);

ResultsForm resultsForm = new ResultsForm(product);

resultsForm.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка обчислення", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private double CalculateProduct(int x, int y, double z)

{

int N = x + y;

if (N < 1)

throw new ArgumentException("Сума x та y повинна бути цілим числом більше або рівним 1.");

double product = 1.0;

for (int i = 1; i <= N; i++)

{

double numerator = 2 \* i - z \* x;

double denominator = Math.Pow(x, 3) - i + Math.Pow(i, 2);

if (denominator == 0)

throw new DivideByZeroException($"Знаменник дорівнює нулю при i = {i}.");

double fraction = numerator / denominator;

if (fraction < 0)

throw new ArithmeticException($"Підкореневий вираз від'ємний при i = {i}.");

double term = Math.Sqrt(fraction);

product \*= term;

}

return product;

}

}

}

***InputForm.cs:***using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace task04\_4

{

public partial class InputForm : Form

{

public int X { get; private set; }

public int Y { get; private set; }

public double Z { get; private set; }

public InputForm()

{

InitializeComponent();

}

private void buttonOK\_Click(object sender, EventArgs e)

{

if (int.TryParse(textBoxX.Text, out int x) &&

int.TryParse(textBoxY.Text, out int y) &&

double.TryParse(textBoxZ.Text, out double z))

{

X = x;

Y = y;

Z = z;

this.DialogResult = DialogResult.OK;

this.Close();

}

else

{

MessageBox.Show("Будь ласка, введіть коректні числові значення для x, y та z.", "Помилка вводу", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

***ResultsForm.cs:***

using System;

using System.Windows.Forms;

namespace task04\_4

{

public partial class ResultsForm : Form

{

public ResultsForm(double result)

{

InitializeComponent();

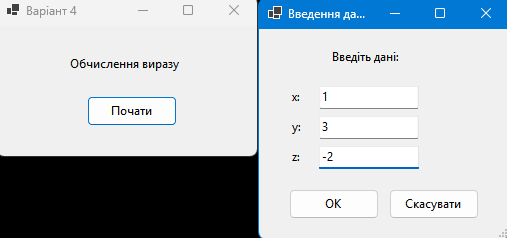
labelResult.Text = $"Результат: {result:F4}";

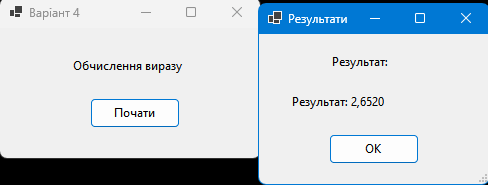
}

}

}

**Результат роботи програми:**





Усі варіанти програм виконують обчислення добутку виразів за заданою математичною формулою, використовуючи дані, введені користувачем. Основна відмінність між варіантами полягає у способі організації інтерфейсу: всі елементи в одній формі або розділення введення даних і відображення результатів на окремі форми.

**Варіант 1: Одна форма для введення даних і результатів**

Програма складається з однієї форми. Користувач вводить значення x, y, z у текстові поля і натискає кнопку *"Обчислити"*. Результат відображається на цій же формі.

* Валідація перевіряє коректність числових значень і цілісність даних.
* У разі помилок (наприклад, ділення на нуль або від'ємного значення під коренем) виводяться відповідні повідомлення.

**Особливості:**

* Простий інтерфейс.
* Швидкий доступ до результату.

**Варіант 2: Окрема форма для введення даних**

Цей варіант має дві форми:

1. **Головна форма** — для запуску процесу обчислень.
2. **Форма введення даних** — для введення значень x, y, z.

Після введення даних у форму введення і натискання кнопки *"OK"* виконується обчислення результату, який одразу відображається на головній формі.

**Особливості:**

* Чітке розділення введення даних і їх обробки.
* Зручність при роботі з великими формами.

**Варіант 3: Окрема форма для результатів**

У цьому варіанті є дві форми:

1. **Головна форма** — для введення значень x, y, z і запуску обчислень.
2. **Форма результатів** — відкривається після виконання обчислень і відображає результат.

Введення даних і обчислення виконуються в головній формі, а результат виводиться у вигляді окремого вікна.

**Особливості:**

* Відокремлений підхід до відображення результатів.
* Візуальне виділення результату.

**Варіант 4: Окремі форми для введення даних і результатів**

Цей варіант поєднує розділення функціоналу введення даних і виведення результатів у три окремі частини:

1. **Головна форма** — запускає програму.
2. **Форма введення даних** — для вводу значень x, y, z.
3. **Форма результатів** — відображає обчислений результат.

Після введення даних форма введення закривається, запускаються обчислення, і результат відображається в новому вікні.

**Особливості:**

* Максимальна модульність.
* Зручність при використанні в багатокрокових задачах.

**Завдання 5**

Створити застосунок, який на головній формі дозволяв би вибирати місто країни згідно до свого варіанту та відобразити фото та короткий опис на додатковій формі

|  |  |  |
| --- | --- | --- |
| № варіанту | Країна | Кількість міст |
| 9. | китай | 7 |

**Текст програми:**

***Form1.cs:***

using System;

using System.Windows.Forms;

using task05;

namespace task05

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void comboBoxCities\_SelectedIndexChanged(object sender, EventArgs e)

{

string selectedCity = comboBoxCities.SelectedItem.ToString();

CityForm cityForm = new CityForm(selectedCity);

cityForm.ShowDialog();

}

}

}

***CityForm.cs:***

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace task05

{

public partial class CityForm : Form

{

private string cityName;

public CityForm(string cityName)

{

InitializeComponent();

this.cityName = cityName;

LoadCityInfo();

}

private void LoadCityInfo()

{

labelCityName.Text = cityName;

// Встановлення опису та зображення міста

switch (cityName)

{

case "Пекін":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Beijing);

textBoxDescription.Text = "Пекін — столиця Китаю, відомий своїми історичними пам'ятками, такими як Заборонене місто та Велика Китайська стіна.";

break;

case "Шанхай":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Shanghai);

textBoxDescription.Text = "Шанхай — найбільше місто Китаю, глобальний фінансовий центр з сучасним горизонтом та жвавими торговими районами.";

break;

case "Гуанчжоу":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Guangzhou);

textBoxDescription.Text = "Гуанчжоу — велике портове місто на півдні Китаю з багатою історією торгівлі та культурними пам'ятками.";

break;

case "Шеньчжень":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Shenzhen);

textBoxDescription.Text = "Шеньчжень — сучасне місто неподалік від Гонконгу, відоме своїми технологічними компаніями та швидким розвитком.";

break;

case "Сіань":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Xian);

textBoxDescription.Text = "Сіань — стародавнє місто, відоме своїми історичними пам'ятками, включаючи теракотову армію імператора Цінь Шихуана.";

break;

case "Ченду":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Chengdu);

textBoxDescription.Text = "Ченду — місто на південному заході Китаю, відоме як батьківщина гігантських панд та своєю смачною кухнею.";

break;

case "Ханчжоу":

pictureBoxCity.Image = ByteArrayToImage(Properties.Resources.Hangzhou);

textBoxDescription.Text = "Ханчжоу — місто, відоме своїм мальовничим Західним озером та історичними храмами і садами.";

break;

default:

pictureBoxCity.Image = null;

textBoxDescription.Text = "Інформація про це місто недоступна.";

break;

}

}

/// <summary>

/// Конвертує масив байтів у об'єкт Image.

/// </summary>

/// <param name="bytes">Масив байтів, що представляє зображення.</param>

/// <returns>Об'єкт Image або null, якщо конвертація не вдалася.</returns>

private Image ByteArrayToImage(byte[] bytes)

{

if (bytes == null || bytes.Length == 0)

return null;

using (MemoryStream ms = new MemoryStream(bytes))

{

try

{

return Image.FromStream(ms);

}

catch

{

return null;

}

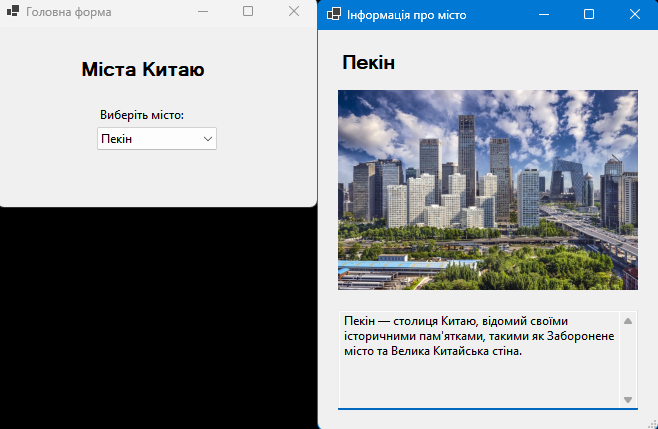
}

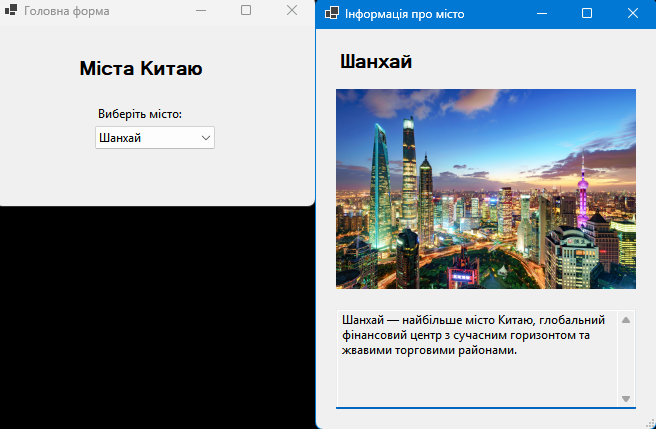
}

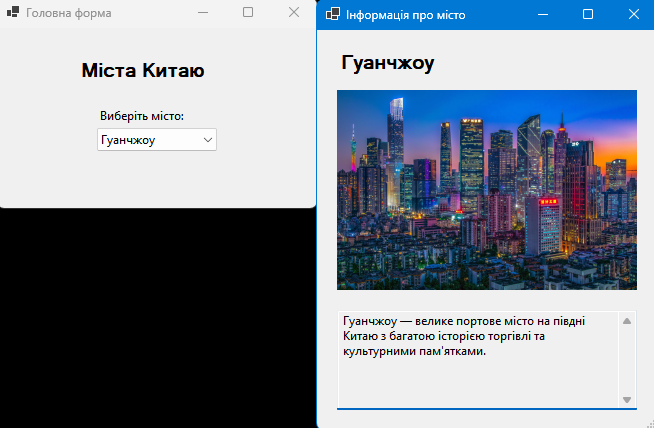
}

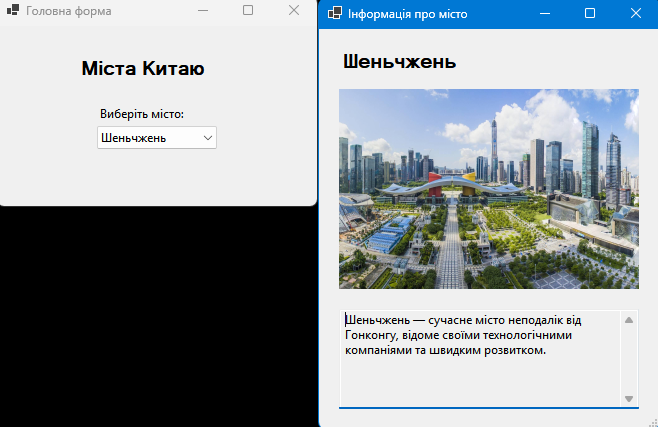
}

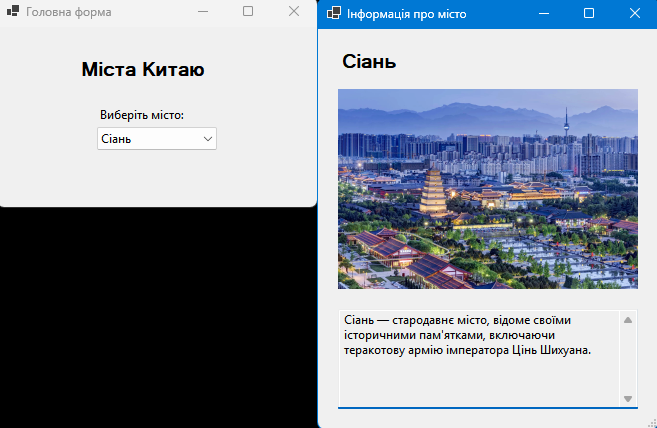
**Результат роботи програми:**

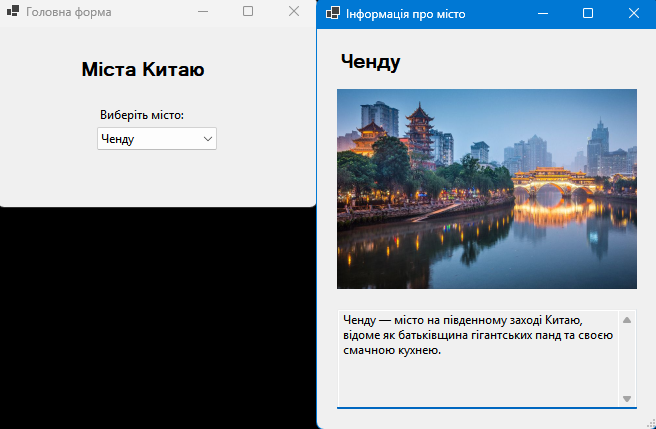


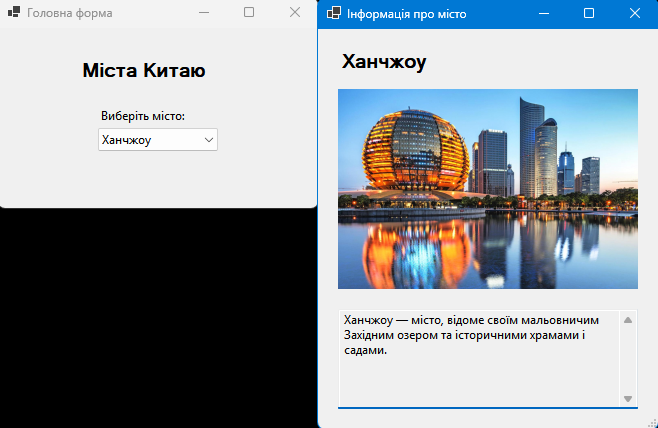












Ця програма є Windows Forms-застосунком, який дозволяє користувачу вибрати китайське місто зі списку та переглянути його опис, а також відповідне зображення. Програма забезпечує інтерактивність через окремі форми для кожного міста.

**Функціональні можливості:**

1. **Головна форма (Form1):**
   * Містить випадаючий список міст (ComboBox).
   * При виборі міста викликається форма з детальною інформацією про обране місто.
2. **Форма міста (CityForm):**
   * Відображає назву міста, його опис та відповідне зображення.
   * Містить текстове поле для опису та графічний компонент (PictureBox) для зображення.

**Технічна реалізація:**

**Form1.cs**

* **Клас Form1:**
  + Ініціалізує головну форму з елементом ComboBox.
  + При зміні вибору в ComboBox створює екземпляр форми CityForm і передає в неї вибране місто.

**CityForm.cs**

* **Клас CityForm:**
  + Отримує назву обраного міста через конструктор.
  + Встановлює назву, опис та зображення залежно від обраного міста.
  + Вибір міста обробляється через конструкцію switch.
  + Використовуються ресурси (зображення) із класу Properties.Resources.
* **Метод LoadCityInfo():**
  + Встановлює текст та зображення залежно від переданого міста.
  + Для невідомих міст відображається текст "Інформація про це місто недоступна" та відсутнє зображення.
* **Метод ByteArrayToImage(byte[] bytes):**
  + Конвертує масив байтів у зображення типу Image для використання в PictureBox.
  + Реалізує обробку помилок для уникнення некоректних даних.

**Завдання 6**

До форми застосунку із четвертого завдання четвертой роботи додати кнопку із назвою, наприклад, Конфігурація a6o Параметри, яка 6 викликала окрему форму для налаштування кольорово-розмірні параметри ()не менше п'яти) компонентів основной форми. Для налаштування рекомендується використовувати компоненти діалогів: ColorDialog та FontDialog.

**Текст програми:**

***Form1.cs:***

using System;

using System.Text;

using System.Windows.Forms;

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

// Обробник події для кнопки "Очистити форму"

private void ClearForm(object sender, EventArgs e)

{

passwordTextBox.Clear();

nameTextBox.Clear();

ageComboBox.SelectedIndex = -1;

maleRadioButton.Checked = false;

femaleRadioButton.Checked = false;

computersCheckBox.Checked = false;

sportsCheckBox.Checked = false;

artCheckBox.Checked = false;

scienceCheckBox.Checked = false;

opinionFileTextBox.Clear();

opinionTextBox.Clear();

}

// Обробник події для кнопки "Огляд..."

private void BrowseFile(object sender, EventArgs e)

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

openFileDialog.Title = "Виберіть файл";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

opinionFileTextBox.Text = openFileDialog.FileName;

}

}

}

// Обробник події для кнопки "Відправити"

private void SubmitData(object sender, EventArgs e)

{

string gender = maleRadioButton.Checked ? "Чоловіча" : "Жіноча";

var interests = new StringBuilder();

if (computersCheckBox.Checked) interests.Append("Комп'ютери, ");

if (sportsCheckBox.Checked) interests.Append("Спорт, ");

if (artCheckBox.Checked) interests.Append("Мистецтво, ");

if (scienceCheckBox.Checked) interests.Append("Наука, ");

string interestsString = interests.Length > 0

? interests.ToString().Substring(0, interests.Length - 2)

: "";

var messageBuilder = new StringBuilder();

messageBuilder.AppendLine("Дані відправлено!\n");

messageBuilder.AppendLine($"Ім'я: {nameTextBox.Text}");

messageBuilder.AppendLine($"Пароль: {passwordTextBox.Text}");

messageBuilder.AppendLine($"Вік: {ageComboBox.SelectedItem}");

messageBuilder.AppendLine($"Стать: {gender}");

messageBuilder.AppendLine($"Інтереси: {interestsString}");

messageBuilder.AppendLine($"Файл: {opinionFileTextBox.Text}");

messageBuilder.AppendLine($"Думка: {opinionTextBox.Text}");

MessageBox.Show(messageBuilder.ToString(), "Підтвердження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

// Обробник події для кнопки "Параметри"

private void OpenSettings(object sender, EventArgs e)

{

SettingsForm settingsForm = new SettingsForm(this);

settingsForm.ShowDialog();

}

// Метод Dispose для звільнення ресурсів

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

// Точка входу в програму

[STAThread]

public static void task05()

{

Application.EnableVisualStyles();

Application.Run(new Form1());

}

}

***SettingsForm.cs:***

using System;

using System.Drawing;

using System.Windows.Forms;

public partial class SettingsForm : Form

{

private Form1 mainForm;

public SettingsForm(Form1 form)

{

InitializeComponent();

mainForm = form;

}

// Обробник події для кнопки "Змінити колір фону форми"

private void buttonChangeBackground\_Click(object sender, EventArgs e)

{

using (ColorDialog colorDialog = new ColorDialog())

{

colorDialog.Color = mainForm.BackColor;

if (colorDialog.ShowDialog() == DialogResult.OK)

{

mainForm.BackColor = colorDialog.Color;

}

}

}

// Обробник події для кнопки "Змінити шрифт кнопки 'Відправити'"

private void buttonChangeButtonFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.submitButton.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

mainForm.submitButton.Font = fontDialog.Font;

}

}

}

// Обробник події для кнопки "Змінити шрифт заголовків"

private void buttonChangeLabelFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.personalInfoGroupBox.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

mainForm.personalInfoGroupBox.Font = fontDialog.Font;

mainForm.opinionGroupBox.Font = fontDialog.Font;

}

}

}

// Обробник події для кнопки "Змінити колір поля 'Думка'"

private void buttonChangeTextBoxColor\_Click(object sender, EventArgs e)

{

using (ColorDialog colorDialog = new ColorDialog())

{

colorDialog.Color = mainForm.opinionTextBox.BackColor;

if (colorDialog.ShowDialog() == DialogResult.OK)

{

mainForm.opinionTextBox.BackColor = colorDialog.Color;

}

}

}

// Обробник події для кнопки "Змінити шрифт чекбоксів 'Інтереси'"

private void buttonChangeCheckBoxFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.computersCheckBox.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

foreach (Control control in mainForm.personalInfoGroupBox.Controls)

{

if (control is CheckBox checkBox)

{

checkBox.Font = fontDialog.Font;

}

}

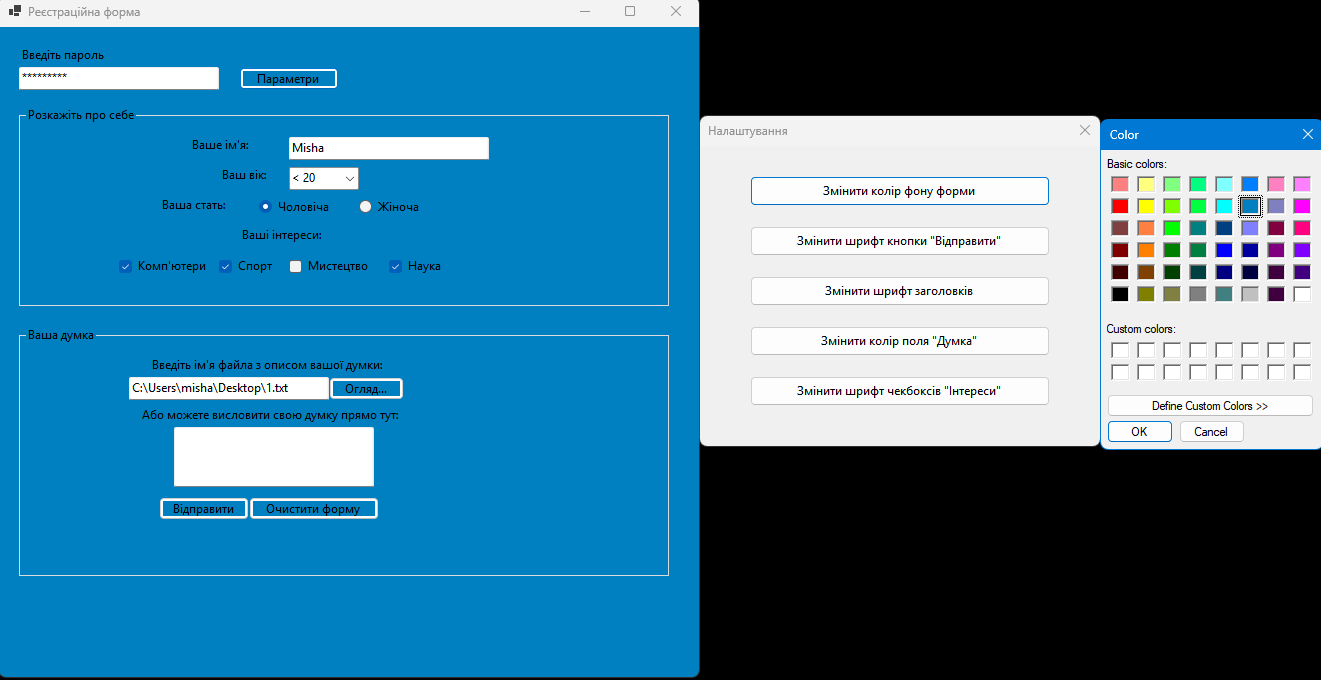
}

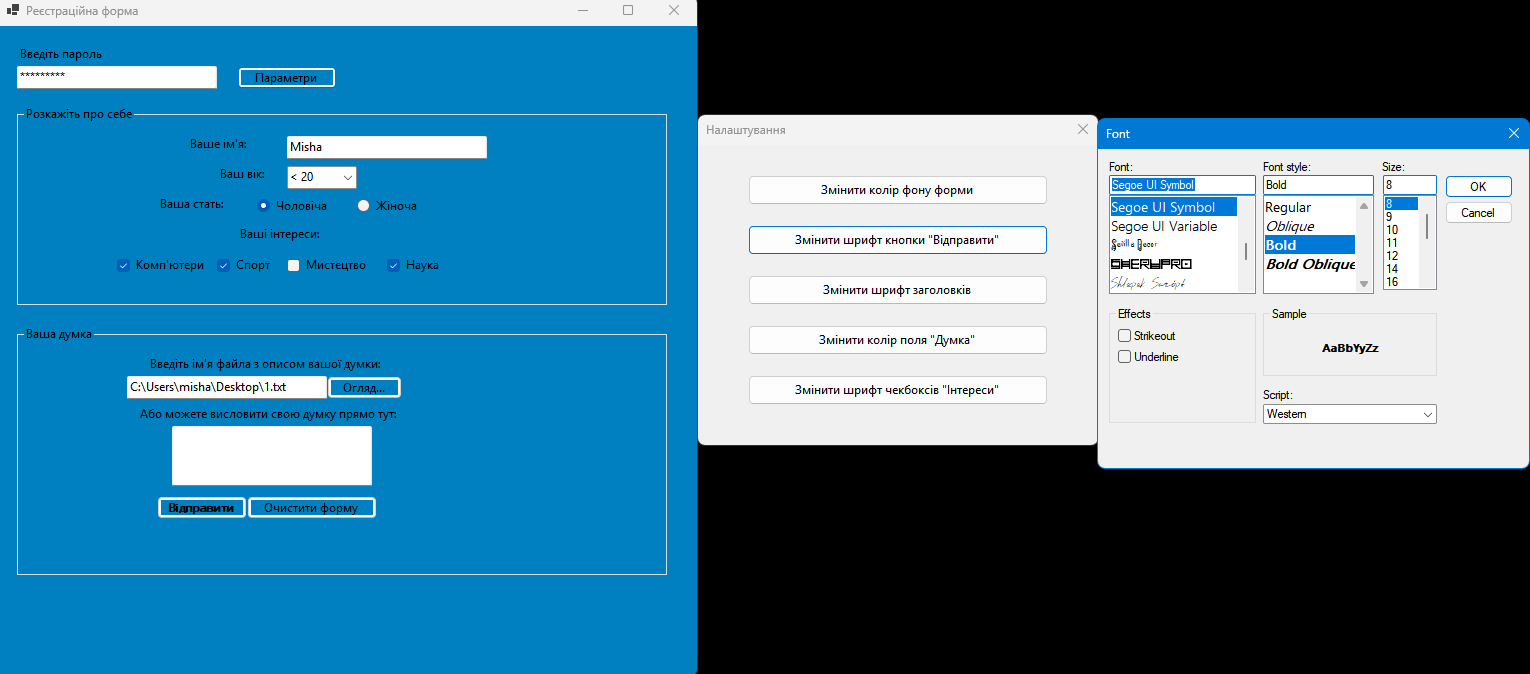
}

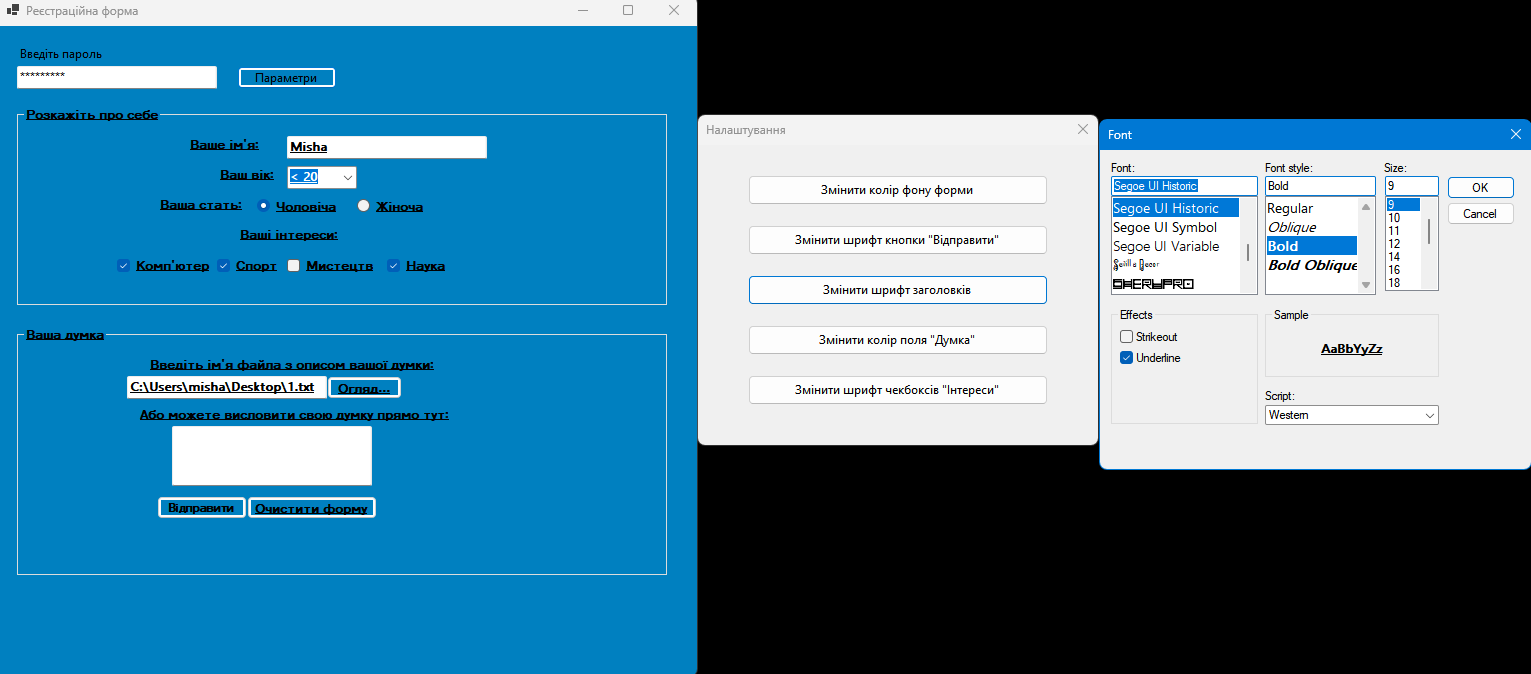
}

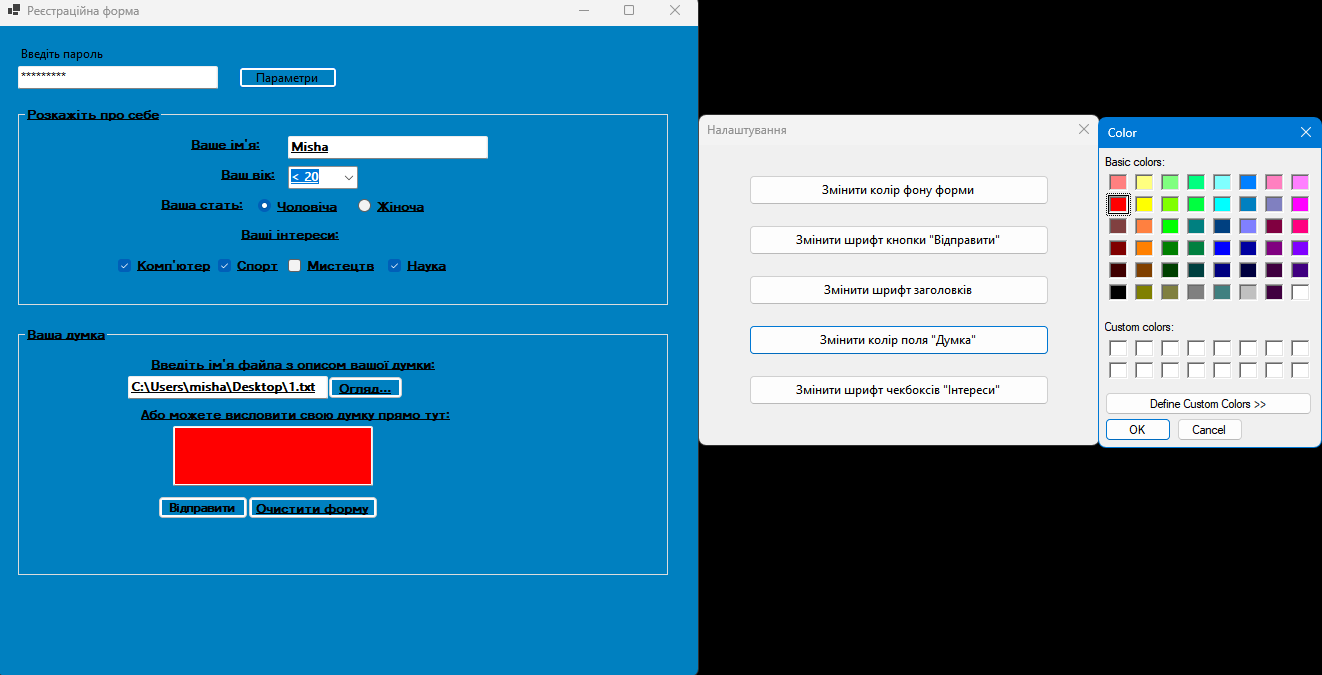
}

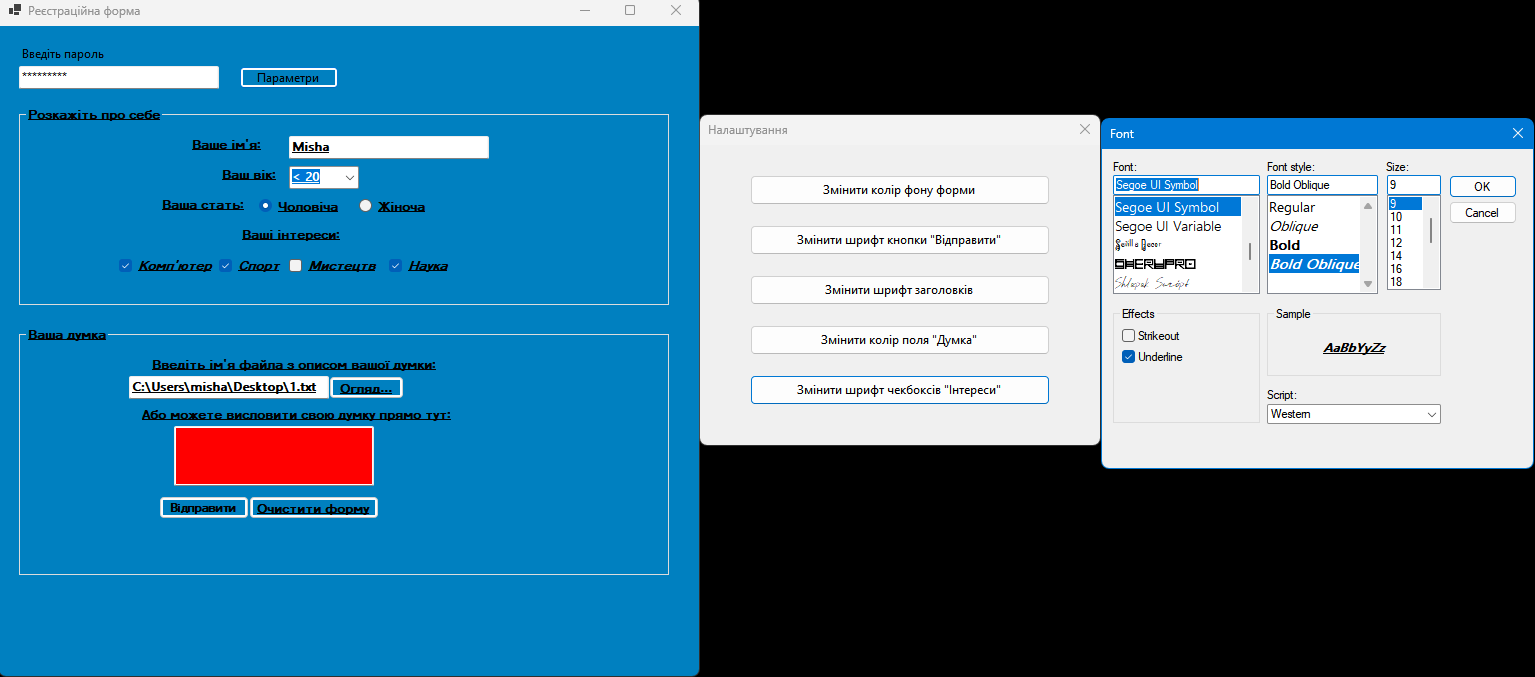
**Результат роботи програми:**











Дана програма на C# реалізує віконний додаток для введення персональних даних користувача з можливістю налаштування інтерфейсу. Основне вікно програми дозволяє вводити інформацію, а друге вікно (форма налаштувань) забезпечує зміну вигляду інтерфейсу.

**Основні функціональні можливості:**

1. **Введення персональних даних:**
   * Поля для введення імені та пароля.
   * Випадаючий список для вибору віку.
   * Радіокнопки для вибору статі.
   * Чекбокси для зазначення інтересів (наприклад, "Комп'ютери", "Спорт", "Мистецтво", "Наука").
   * Поле для введення тексту думки.
   * Поле для вибору файлу через стандартний діалоговий інструмент Windows.
2. **Дії з введеними даними:**
   * **Очищення форми:** Кнопка "Очистити форму" скидає всі введені дані до початкового стану.
   * **Відправка даних:** Кнопка "Відправити" збирає введену інформацію в структурований текст та виводить у вікні повідомлення.
3. **Налаштування інтерфейсу (додаткове вікно):**
   * Кнопка "Параметри" відкриває форму налаштувань, яка дозволяє:
     + Змінювати колір фону основної форми.
     + Змінювати шрифт кнопки "Відправити".
     + Змінювати шрифт заголовків груп (наприклад, "Особиста інформація").
     + Змінювати колір текстового поля для думки.
     + Змінювати шрифт чекбоксів у секції інтересів.
4. **Динамічна робота з файлами:**
   * Можливість вибору текстового файлу через діалогове вікно та відображення його шляху в текстовому полі.

**Завдання 7\***

Модифікувати застосунок минулого завдання додавши до конфігураційної форми можливість зберігати та завантажувати конфігураційні параметри компонентів головной форми у(з) зовнішніх (за бажанням текстових a6o двійкових) файлах, застосовуючи для цього компоненти OpenFileDialog та SaveFileDialog. Передбачити можливість автоматичного завантаження параметрів компонентів головной форми під час її запуску.

**Текст програми:**

***Form1.cs:***using System;

using System.IO;

using System.Text;

using System.Windows.Forms;

using System.Xml.Serialization;

using task07;

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

// Завантаження конфігурації при запуску

LoadConfigurationAtStartup();

}

// Метод для автоматичного завантаження конфігурації

private void LoadConfigurationAtStartup()

{

string configFile = "default.cfg";

if (File.Exists(configFile))

{

try

{

XmlSerializer serializer = new XmlSerializer(typeof(Configuration));

using (FileStream fs = new FileStream(configFile, FileMode.Open))

{

Configuration config = (Configuration)serializer.Deserialize(fs);

config.ApplyConfiguration(this);

}

}

catch (Exception ex)

{

MessageBox.Show("Помилка при завантаженні конфігурації: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

// Обробник події для кнопки "Очистити форму"

private void ClearForm(object sender, EventArgs e)

{

passwordTextBox.Clear();

nameTextBox.Clear();

ageComboBox.SelectedIndex = -1;

maleRadioButton.Checked = false;

femaleRadioButton.Checked = false;

computersCheckBox.Checked = false;

sportsCheckBox.Checked = false;

artCheckBox.Checked = false;

scienceCheckBox.Checked = false;

opinionFileTextBox.Clear();

opinionTextBox.Clear();

}

// Обробник події для кнопки "Огляд..."

private void BrowseFile(object sender, EventArgs e)

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

openFileDialog.Title = "Виберіть файл";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

opinionFileTextBox.Text = openFileDialog.FileName;

}

}

}

// Обробник події для кнопки "Відправити"

private void SubmitData(object sender, EventArgs e)

{

string gender = maleRadioButton.Checked ? "Чоловіча" : "Жіноча";

var interests = new StringBuilder();

if (computersCheckBox.Checked) interests.Append("Комп'ютери, ");

if (sportsCheckBox.Checked) interests.Append("Спорт, ");

if (artCheckBox.Checked) interests.Append("Мистецтво, ");

if (scienceCheckBox.Checked) interests.Append("Наука, ");

string interestsString = interests.Length > 0

? interests.ToString().Substring(0, interests.Length - 2)

: "";

var messageBuilder = new StringBuilder();

messageBuilder.AppendLine("Дані відправлено!\n");

messageBuilder.AppendLine($"Ім'я: {nameTextBox.Text}");

messageBuilder.AppendLine($"Пароль: {passwordTextBox.Text}");

messageBuilder.AppendLine($"Вік: {ageComboBox.SelectedItem}");

messageBuilder.AppendLine($"Стать: {gender}");

messageBuilder.AppendLine($"Інтереси: {interestsString}");

messageBuilder.AppendLine($"Файл: {opinionFileTextBox.Text}");

messageBuilder.AppendLine($"Думка: {opinionTextBox.Text}");

MessageBox.Show(messageBuilder.ToString(), "Підтвердження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

// Обробник події для кнопки "Параметри"

private void OpenSettings(object sender, EventArgs e)

{

SettingsForm settingsForm = new SettingsForm(this);

settingsForm.ShowDialog();

}

// Метод Dispose для звільнення ресурсів

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

// Точка входу в програму

[STAThread]

public static void mainForm()

{

Application.EnableVisualStyles();

Application.Run(new Form1());

}

}

***SettingsForm.cs:***

using System;

using System.Drawing;

using System.IO;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Serialization;

public partial class SettingsForm : Form

{

private Form1 mainForm;

public SettingsForm(Form1 form)

{

InitializeComponent();

mainForm = form;

}

// Обробник події для кнопки "Змінити колір фону форми"

private void buttonChangeBackground\_Click(object sender, EventArgs e)

{

using (ColorDialog colorDialog = new ColorDialog())

{

colorDialog.Color = mainForm.BackColor;

if (colorDialog.ShowDialog() == DialogResult.OK)

{

mainForm.BackColor = colorDialog.Color;

}

}

}

// Обробник події для кнопки "Змінити шрифт кнопки 'Відправити'"

private void buttonChangeButtonFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.submitButton.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

mainForm.submitButton.Font = fontDialog.Font;

}

}

}

// Обробник події для кнопки "Змінити шрифт заголовків"

private void buttonChangeLabelFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.personalInfoGroupBox.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

mainForm.personalInfoGroupBox.Font = fontDialog.Font;

mainForm.opinionGroupBox.Font = fontDialog.Font;

}

}

}

// Обробник події для кнопки "Змінити колір поля 'Думка'"

private void buttonChangeTextBoxColor\_Click(object sender, EventArgs e)

{

using (ColorDialog colorDialog = new ColorDialog())

{

colorDialog.Color = mainForm.opinionTextBox.BackColor;

if (colorDialog.ShowDialog() == DialogResult.OK)

{

mainForm.opinionTextBox.BackColor = colorDialog.Color;

}

}

}

// Обробник події для кнопки "Змінити шрифт чекбоксів 'Інтереси'"

private void buttonChangeCheckBoxFont\_Click(object sender, EventArgs e)

{

using (FontDialog fontDialog = new FontDialog())

{

fontDialog.Font = mainForm.computersCheckBox.Font;

if (fontDialog.ShowDialog() == DialogResult.OK)

{

foreach (Control control in mainForm.personalInfoGroupBox.Controls)

{

if (control is CheckBox checkBox)

{

checkBox.Font = fontDialog.Font;

}

}

}

}

}

// Обробник події для кнопки "Зберегти конфігурацію"

private void buttonSaveConfiguration\_Click(object sender, EventArgs e)

{

using (SaveFileDialog saveFileDialog = new SaveFileDialog())

{

saveFileDialog.Filter = "Файли конфігурації (\*.cfg)|\*.cfg|Всі файли (\*.\*)|\*.\*";

saveFileDialog.Title = "Зберегти конфігурацію";

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

try

{

Configuration config = new Configuration(mainForm);

XmlSerializer serializer = new XmlSerializer(typeof(Configuration));

using (FileStream fs = new FileStream(saveFileDialog.FileName, FileMode.Create))

{

serializer.Serialize(fs, config);

}

MessageBox.Show("Конфігурацію успішно збережено.", "Збереження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Помилка при збереженні конфігурації: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

// Обробник події для кнопки "Завантажити конфігурацію"

private void buttonLoadConfiguration\_Click(object sender, EventArgs e)

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Файли конфігурації (\*.cfg)|\*.cfg|Всі файли (\*.\*)|\*.\*";

openFileDialog.Title = "Завантажити конфігурацію";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

try

{

XmlSerializer serializer = new XmlSerializer(typeof(Configuration));

using (FileStream fs = new FileStream(openFileDialog.FileName, FileMode.Open))

{

Configuration config = (Configuration)serializer.Deserialize(fs);

config.ApplyConfiguration(mainForm);

}

MessageBox.Show("Конфігурацію успішно завантажено.", "Завантаження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Помилка при завантаженні конфігурації: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

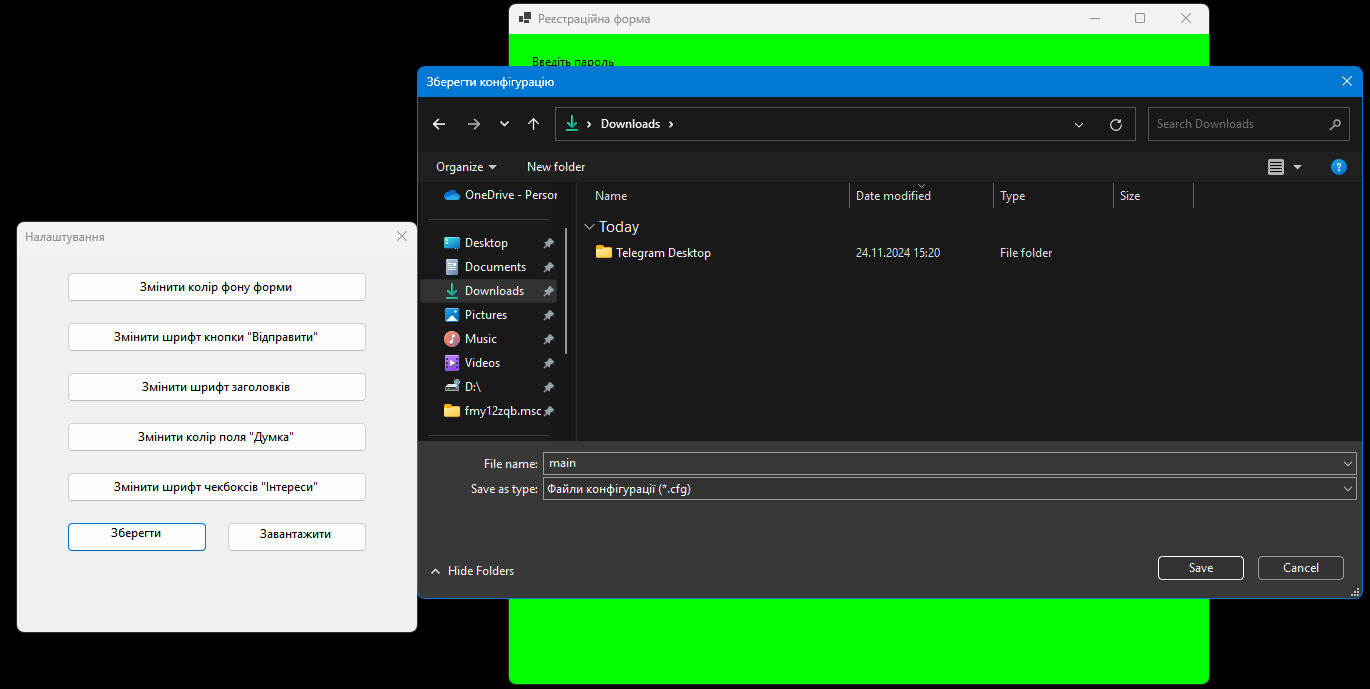
}

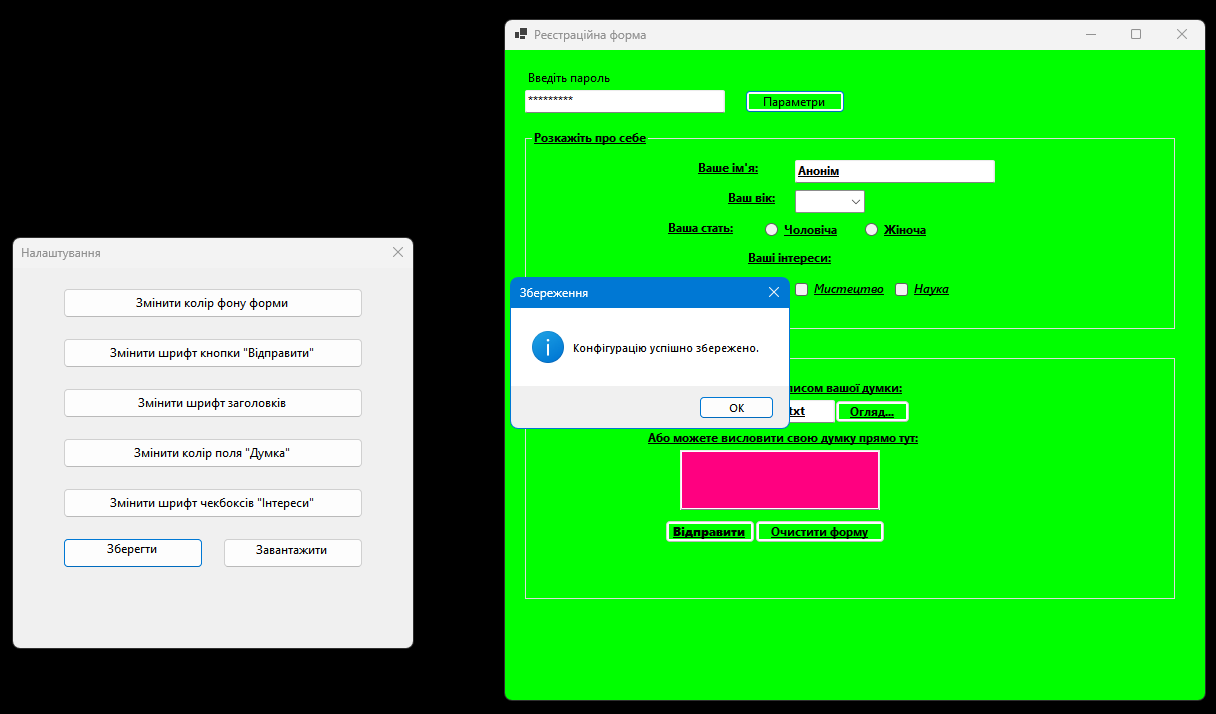
}

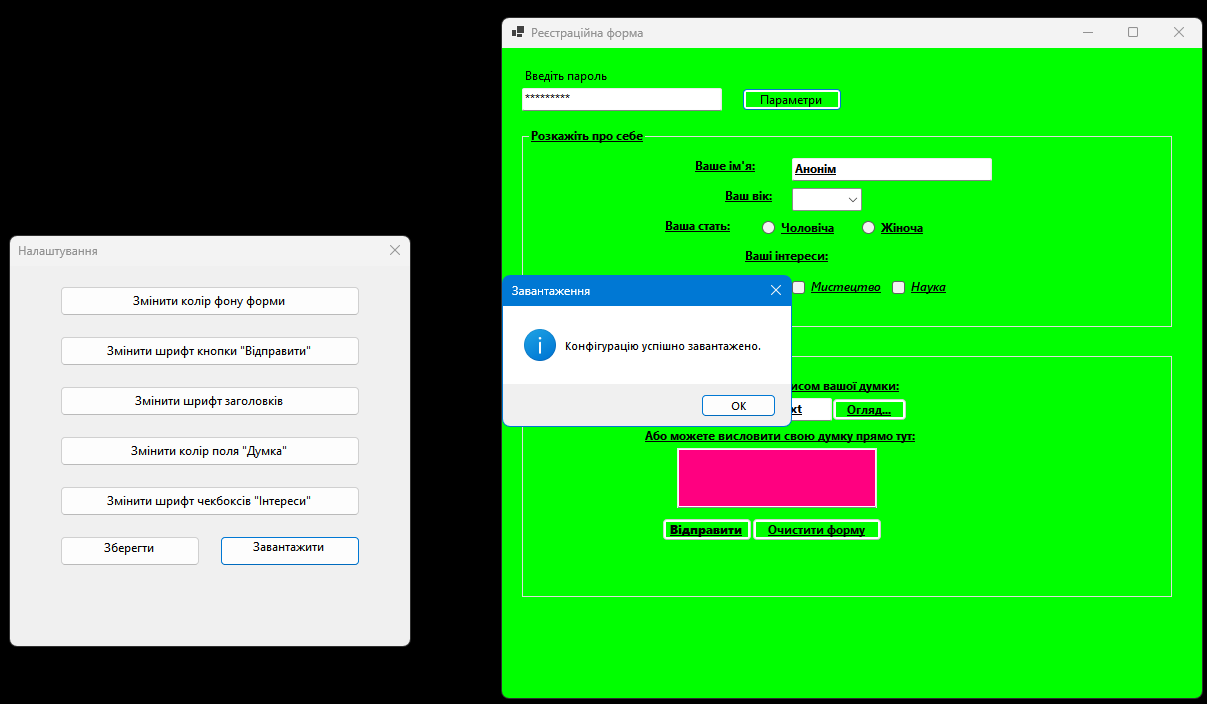
}

}

**Результат роботи програми:**







Ця програма є Windows Forms-додатком на C#, який реалізує форму введення персональних даних із можливістю зміни параметрів зовнішнього вигляду та функціональних налаштувань. Вона підтримує динамічне збереження та завантаження конфігурацій, а також включає функції взаємодії з користувачем через кнопки, текстові поля, чекбокси та випадаючі списки.

**Основні компоненти програми**

**1. Головна форма (Form1)**

* **Елементи вводу:**
  + Поля для введення імені, паролю, вибору віку (ComboBox), статі (RadioButton) та інтересів (CheckBox).
  + Текстове поле для введення думки та обрання файлу.
* **Кнопки:**
  + **"Очистити форму"** – скидає всі введені дані до початкового стану.
  + **"Огляд..."** – відкриває діалогове вікно для вибору файлу.
  + **"Відправити"** – відображає повідомлення із введеними даними.
  + **"Параметри"** – відкриває додаткову форму налаштувань.
* **Автоматичне завантаження конфігурації**: при запуску програми, якщо доступний файл default.cfg, з нього завантажуються налаштування.

**2. Форма налаштувань (SettingsForm)**

* Забезпечує можливість динамічно змінювати зовнішній вигляд та налаштування програми:
  + **Колір фону головної форми.**
  + **Шрифт кнопки "Відправити".**
  + **Шрифт заголовків груп.**
  + **Колір текстового поля для введення думки.**
  + **Шрифт чекбоксів "Інтереси".**
* **Збереження та завантаження конфігурації:**
  + **"Зберегти конфігурацію"** – зберігає поточні налаштування у файл формату .cfg.
  + **"Завантажити конфігурацію"** – завантажує налаштування з вибраного файлу.

**3. Конфігурація**

* Налаштування програми (колір, шрифти) зберігаються у форматі XML, використовуючи серіалізацію.
* Містить метод ApplyConfiguration, який застосовує завантажені налаштування до елементів головної форми.

**Завдання 8**

Розробити застосунок текстовий редактор та забезпечити йому наступні функцїі:

* створення нового файлу, збереженні та завантаження файлу;
* виведення поточної позиції курсора у документі;
* вікно «Про програму» з ім'ям розробника програми.

Додатково доповнити функціями відповідно до варіанта

|  |  |
| --- | --- |
| № варіанту | Додаткова функція |
| 1 | 2 |
| 9. | Організувати пошук у тексті заданого слова, виділивши його |

**Текст програми:**

***Form1.cs:***

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

using task08;

namespace task08

{

public partial class Form1 : Form

{

private string currentFilePath = string.Empty;

public Form1()

{

InitializeComponent();

UpdateCursorPosition();

}

// Оновлення відображення позиції курсора

private void richTextBoxEditor\_SelectionChanged(object sender, EventArgs e)

{

UpdateCursorPosition();

}

private void UpdateCursorPosition()

{

int line = richTextBoxEditor.GetLineFromCharIndex(richTextBoxEditor.SelectionStart) + 1;

int column = richTextBoxEditor.SelectionStart - richTextBoxEditor.GetFirstCharIndexOfCurrentLine() + 1;

toolStripStatusLabel.Text = $"Рядок: {line}, Стовпчик: {column}";

}

// Створення нового файлу

private void newToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ConfirmSaveChanges())

{

richTextBoxEditor.Clear();

currentFilePath = string.Empty;

this.Text = "Новий файл - Текстовий редактор";

}

}

// Відкриття файлу

private void openToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ConfirmSaveChanges())

{

OpenFile();

}

}

private void OpenFile()

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

openFileDialog.Title = "Виберіть файл";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

try

{

richTextBoxEditor.Text = File.ReadAllText(openFileDialog.FileName);

currentFilePath = openFileDialog.FileName;

this.Text = Path.GetFileName(currentFilePath) + " - Текстовий редактор";

}

catch (Exception ex)

{

MessageBox.Show("Помилка при відкритті файлу: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

// Збереження файлу

private void saveToolStripMenuItem\_Click(object sender, EventArgs e)

{

SaveFile();

}

private void SaveFile()

{

if (string.IsNullOrEmpty(currentFilePath))

{

SaveFileAs();

}

else

{

try

{

File.WriteAllText(currentFilePath, richTextBoxEditor.Text);

MessageBox.Show("Файл успішно збережено.", "Збереження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Помилка при збереженні файлу: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

// Збереження файлу як...

private void saveAsToolStripMenuItem\_Click(object sender, EventArgs e)

{

SaveFileAs();

}

private void SaveFileAs()

{

using (SaveFileDialog saveFileDialog = new SaveFileDialog())

{

saveFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

try

{

File.WriteAllText(saveFileDialog.FileName, richTextBoxEditor.Text);

currentFilePath = saveFileDialog.FileName;

this.Text = Path.GetFileName(currentFilePath) + " - Текстовий редактор";

MessageBox.Show("Файл успішно збережено.", "Збереження", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Помилка при збереженні файлу: " + ex.Message, "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

// Вихід з програми

private void exitToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ConfirmSaveChanges())

{

Application.Exit();

}

}

// Відображення вікна "Про програму"

private void aboutToolStripMenuItem\_Click(object sender, EventArgs e)

{

AboutForm aboutForm = new AboutForm();

aboutForm.ShowDialog();

}

// Пошук слова в тексті

private void findToolStripMenuItem\_Click(object sender, EventArgs e)

{

FindWord();

}

private void FindWord()

{

string word = Prompt.ShowDialog("Введіть слово для пошуку:", "Пошук слова");

if (!string.IsNullOrEmpty(word))

{

// Скидання попереднього виділення

richTextBoxEditor.SelectAll();

richTextBoxEditor.SelectionBackColor = Color.White;

richTextBoxEditor.DeselectAll();

int startIndex = 0;

int foundIndex = -1;

bool found = false;

while ((foundIndex = richTextBoxEditor.Text.IndexOf(word, startIndex, StringComparison.CurrentCultureIgnoreCase)) != -1)

{

richTextBoxEditor.Select(foundIndex, word.Length);

richTextBoxEditor.SelectionBackColor = Color.Yellow;

startIndex = foundIndex + word.Length;

found = true;

}

if (!found)

{

MessageBox.Show($"Слово \"{word}\" не знайдено.", "Пошук", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

// Метод для підтвердження збереження змін

private bool ConfirmSaveChanges()

{

if (richTextBoxEditor.Modified)

{

DialogResult result = MessageBox.Show("Бажаєте зберегти зміни?", "Збереження", MessageBoxButtons.YesNoCancel, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

SaveFile();

return true;

}

else if (result == DialogResult.No)

{

return true;

}

else // Cancel

{

return false;

}

}

return true;

}

// Метод Dispose для звільнення ресурсів

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

// Точка входу в програму

[STAThread]

public static void form()

{

Application.EnableVisualStyles();

Application.Run(new Form1());

}

}

// Допоміжний клас для введення слова

public static class Prompt

{

public static string ShowDialog(string text, string caption)

{

Form prompt = new Form()

{

Width = 400,

Height = 150,

FormBorderStyle = FormBorderStyle.FixedDialog,

Text = caption,

StartPosition = FormStartPosition.CenterParent

};

Label textLabel = new Label() { Left = 20, Top = 20, Text = text, Width = 340 };

TextBox inputBox = new TextBox() { Left = 20, Top = 50, Width = 340 };

Button confirmation = new Button() { Text = "ОК", Left = 280, Width = 80, Top = 80, DialogResult = DialogResult.OK };

confirmation.Click += (sender, e) => { prompt.Close(); };

prompt.Controls.Add(textLabel);

prompt.Controls.Add(inputBox);

prompt.Controls.Add(confirmation);

prompt.AcceptButton = confirmation;

return prompt.ShowDialog() == DialogResult.OK ? inputBox.Text : "";

}

}

}

***AboutForm.cs:***

using System.Windows.Forms;

namespace task08

{

public partial class AboutForm : Form

{

public AboutForm()

{

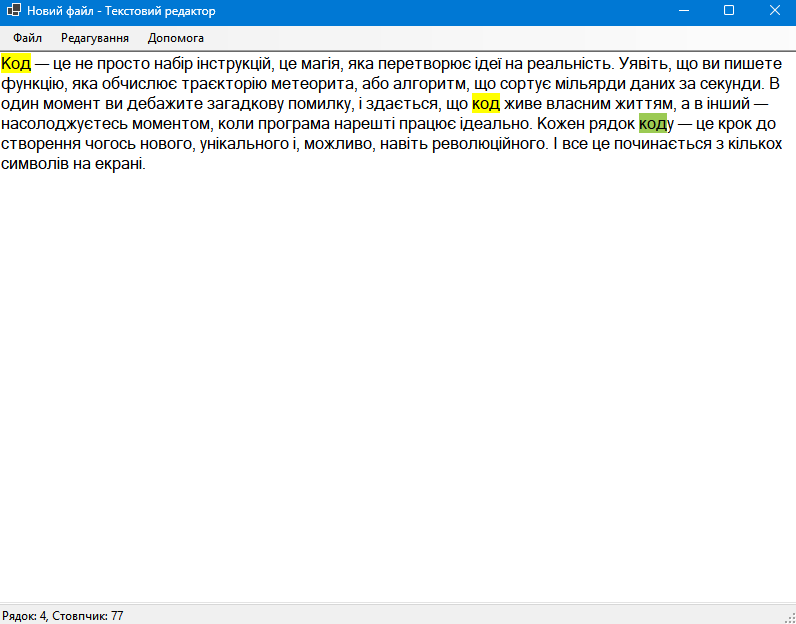
InitializeComponent();

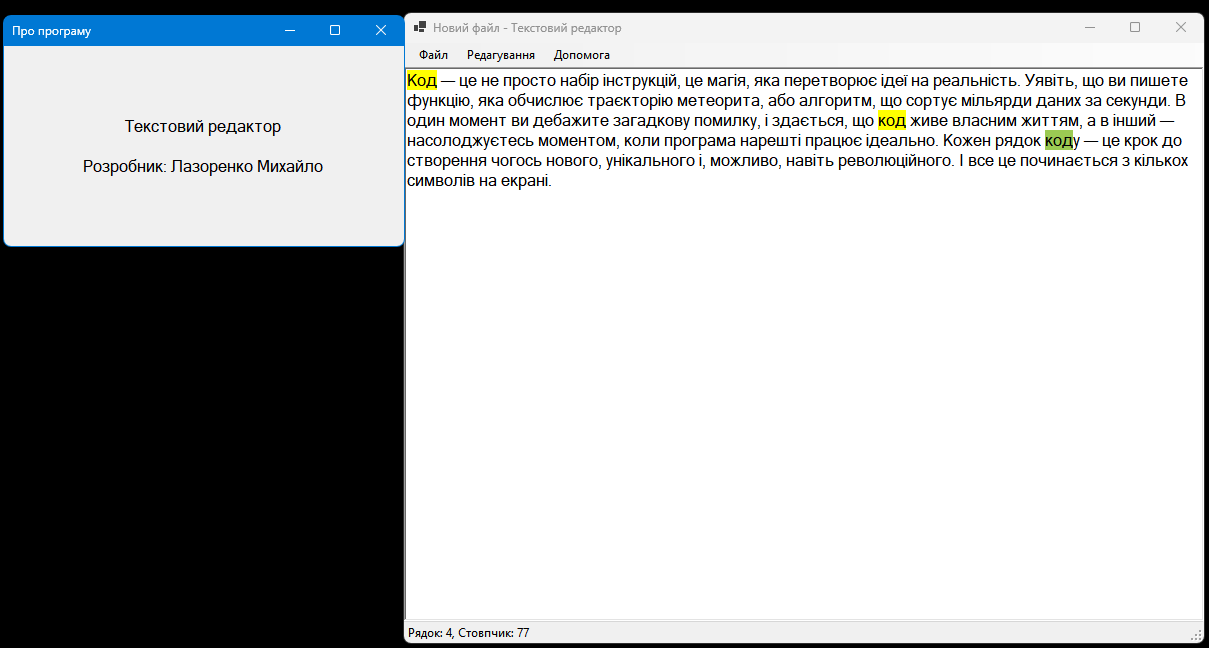
}

}

}

**Результат роботи програми:**





Ця програма є простим текстовим редактором, розробленим на мові C# з використанням технології Windows Forms. Основні можливості програми включають створення, редагування, пошук, збереження текстових файлів, а також роботу з графічним інтерфейсом користувача.

**Основні функціональні можливості:**

1. **Створення нового файлу**:
   * Користувач може створити новий текстовий документ. Перед цим програма запропонує зберегти зміни в поточному документі, якщо такі є.
2. **Відкриття існуючого файлу**:
   * Завантаження текстового файлу з локального диска. Підтримуються текстові файли з розширенням .txt та інші типи файлів.
3. **Збереження файлу**:
   * Можливість збереження змін у поточному файлі. Якщо файл ще не збережений, буде запропоновано вказати його назву та місце розташування.
4. **Збереження файлу як...**:
   * Збереження документа у новому файлі з вибором шляху та імені.
5. **Пошук тексту**:
   * Функція дозволяє знайти певне слово в документі. Знайдені слова виділяються жовтим кольором для кращої видимості.
6. **Відображення позиції курсора**:
   * Програма в режимі реального часу показує номер рядка та стовпчика, в якому знаходиться курсор.
7. **Підтвердження змін перед закриттям**:
   * Якщо користувач намагається створити новий файл, відкрити інший або вийти з програми, перед цим буде запропоновано зберегти зміни.
8. **Про програму**:
   * Окреме вікно "Про програму" надає інформацію про розробника або інші дані.
9. **Графічний інтерфейс**:
   * Головне вікно програми містить текстовий редактор на базі RichTextBox, меню з основними командами (створення, відкриття, збереження, пошук, вихід), а також рядок стану для відображення позиції курсора.

**Висновок**

В рамках лабораторної роботи №12 з теми "Створення багатовіконних застосунків. Частина 1" було виконано завдання, що дозволило набути практичних навичок у розробці застосунків з багатократним використанням вікон.

У процесі виконання лабораторної роботи було засвоєно ключові принципи створення графічного інтерфейсу користувача (GUI) за допомогою Windows Forms, а також управління кількома вікнами в межах одного застосунку. Зокрема, реалізовано основні операції з файлами, таких як відкриття, збереження та пошук у текстових документах, що є важливими функціями для текстових редакторів.

Крім того, було досягнуто розуміння важливості ефективної організації користувацького інтерфейсу, зокрема правильного розташування елементів управління та забезпечення зручності роботи з програмою. Окрему увагу було приділено обробці подій, які виникають у процесі взаємодії користувача з програмою, що є основою для інтерактивних застосунків.

Отже, виконання цієї лабораторної роботи дозволило поглибити розуміння принципів створення багатовіконних застосунків та застосування цих принципів для вирішення практичних завдань. В подальшому це знання можна використовувати для розробки більш складних багатовіконних програм із покращеним користувацьким досвідом.