**Виконав: Мельник Б. В.; КН-922в**

**Лабораторна робота №5**

**Тема:** Використання конструктора форм для створення та налаштування інтерфейсу віконних застосунків. Частина 1.

**Мета роботи:** вивчення властивостей та використання компонентів конструктора форм для створення i налаштування віконного інтерфейсу застосунків.

**Індивідуальні завдання**

**Варіант №12**

**Завдання 1**

Використовуючи компонент Label та його відповідні властивості, створити застосунок, вікно якого містить три рядки будь-якого тексту, які відрізняються один від одного за розміром, кольором, типом вирівнювання та формою тексту згідно із параметрами свого варіанту

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № варіанту | № рядка | Розмір | Кольори тексту/фону | Тип вирівнювання | Форма |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 12 | 1 | 16pt | Рожевий/ Зелений | Ліве | Жирний, курсив |
| 2 | 14 t | Синій/ Червоний | Праве | Курсив, з підкресленням |
| 3 | 17pt | Блакитний, Жовтий | Центр | Жирний, курсив, з підкресленням |

**Текст програми:**

using System; // Підключення бібліотеки для базових функцій C#

using System.Drawing; // Підключення бібліотеки для роботи з графікою

using System.Windows.Forms; // Підключення бібліотеки для створення віконних додатків

public class Task\_1 : Form // Оголошення класу Task\_1, який успадковує клас Form (вікно додатку)

{

public Task\_1() // Конструктор класу Task\_1

{

// Налаштування першого рядка

System.Windows.Forms.Label label1 = new System.Windows.Forms.Label(); // Створення нового елементу Label (мітка)

label1.Text = "Перший рядок"; // Встановлення тексту для першого рядка

label1.Font = new System.Drawing.Font("Arial", 16, FontStyle.Bold | FontStyle.Italic); // Задання шрифту, його розміру і стилю (жирний, курсив)

label1.ForeColor = Color.Pink; // Колір тексту - рожевий

label1.BackColor = Color.Green; // Фоновий колір - зелений

label1.TextAlign = ContentAlignment.MiddleLeft; // Вирівнювання тексту по лівому краю

label1.Dock = DockStyle.Top; // Розташування мітки зверху вікна

// Налаштування другого рядка

System.Windows.Forms.Label label2 = new System.Windows.Forms.Label(); // Створення другого елементу Label

label2.Text = "Другий рядок"; // Встановлення тексту для другого рядка

label2.Font = new System.Drawing.Font("Arial", 14, FontStyle.Italic | FontStyle.Underline); // Задання шрифту зі стилем курсиву та підкреслення

label2.ForeColor = Color.Blue; // Колір тексту - синій

label2.BackColor = Color.Red; // Фоновий колір - червоний

label2.TextAlign = ContentAlignment.MiddleRight; // Вирівнювання тексту по правому краю

label2.Dock = DockStyle.Top; // Розташування мітки зверху (під попередньою міткою)

// Налаштування третього рядка

System.Windows.Forms.Label label3 = new System.Windows.Forms.Label(); // Створення третього елементу Label

label3.Text = "Третій рядок"; // Встановлення тексту для третього рядка

label3.Font = new System.Drawing.Font("Arial", 17, FontStyle.Bold | FontStyle.Italic | FontStyle.Underline); // Шрифт із трьома стилями: жирний, курсив, підкреслення

label3.ForeColor = Color.LightBlue; // Колір тексту - світло-блакитний

label3.BackColor = Color.Yellow; // Фоновий колір - жовтий

label3.TextAlign = ContentAlignment.MiddleCenter; // Вирівнювання тексту по центру

label3.Dock = DockStyle.Top; // Розташування мітки зверху (під другим рядком)

// Додавання рядків до форми

this.Controls.Add(label3); // Додавання третього рядка на форму

this.Controls.Add(label2); // Додавання другого рядка на форму

this.Controls.Add(label1); // Додавання першого рядка на форму

// Налаштування форми

this.Text = "Task\_1"; // Встановлення назви форми

this.Size = new Size(400, 300); // Встановлення розміру форми (ширина 400, висота 300)

}

[STAThread] // Вказує на те, що головний потік є однопоточним (Single Threaded Apartment)

public static void Main() // Головна функція програми

{

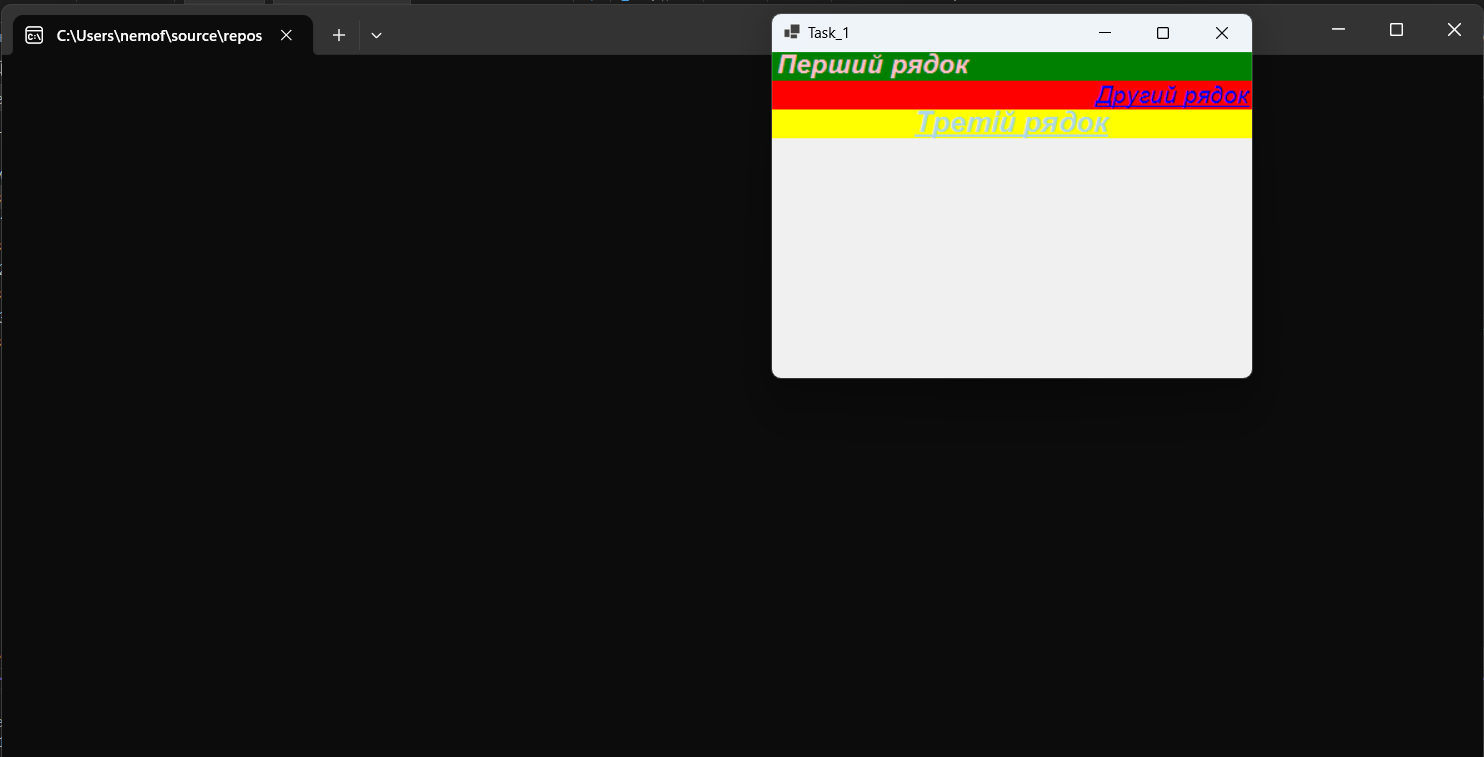
Application.EnableVisualStyles(); // Увімкнення візуальних стилів для форми

System.Windows.Forms.Application.Run(new Task\_1()); // Запуск додатку з головною формою Task\_1

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Оголошення форми**: Створюється клас Task\_1, що успадковує клас Form. Це дозволяє використовувати віконні елементи та керувати вікном програми.
2. **Створення елементів Label**: Для кожного рядка тексту створюється окремий елемент Label. Кожен рядок налаштовується за такими параметрами:
   * **Текст**: Встановлюється текст для кожного рядка.
   * **Шрифт і стиль**: Визначаються шрифт, розмір та стиль (жирний, курсив, підкреслення).
   * **Колір тексту та фон**: Встановлюються різні кольори для тексту та фону кожного рядка.
   * **Вирівнювання тексту**: Налаштовується вирівнювання по лівому, правому або центральному краю.
   * **Розташування**: Використовується властивість Dock, щоб розташувати рядки один під одним.
3. **Додавання рядків до форми**: Кожен елемент Label додається до форми через метод Controls.Add.
4. **Налаштування форми**: Визначається заголовок вікна та його розмір.
5. **Запуск програми**: Викликається метод Application.Run, який запускає вікно програми з налаштованими елементами.

**Функціональність:**

Програма створює вікно, в якому відображаються три рядки тексту:

* Перший рядок: рожевий текст на зеленому фоні, вирівняний по лівому краю, із жирним курсивом.
* Другий рядок: синій текст на червоному фоні, вирівняний по правому краю, курсив із підкресленням.
* Третій рядок: світло-блакитний текст на жовтому фоні, вирівняний по центру, жирний курсив із підкресленням.

**Висновок:**

Застосунок демонструє можливості компонента Label для форматування тексту у віконних додатках. Було успішно реалізовано відмінності між текстовими рядками за допомогою різних стилів шрифту, кольорів та вирівнювань, що відповідає умовам завдання.

**Завдання 2\***

Виконати минуле завдання додавши рядки тексту програмним шляхом, тобто під час завантаження вікна застосунку.

**Текст програми:**

using System; // Підключення простору імен System для основних функцій

using System.Drawing; // Підключення простору імен для роботи з графікою та кольорами

using System.Windows.Forms; // Підключення простору імен для створення Windows Forms додатків

public class Task\_2 : Form // Оголошення класу Task\_2, який наслідує клас Form (створює вікно)

{

public Task\_2() // Конструктор класу Task\_2

{

// Налаштування заголовку форми

this.Text = "Task\_2";

// Встановлення розміру форми

this.Size = new Size(400, 300);

// Додаємо обробник події завантаження форми

this.Load += new EventHandler(Form\_Load);

}

private void Form\_Load(object sender, EventArgs e) // Метод, що викликається при завантаженні форми

{

// Створюємо перший рядок

System.Windows.Forms.Label label1 = new System.Windows.Forms.Label();

label1.Text = "Перший рядок"; // Встановлюємо текст для першого рядка

label1.Font = new System.Drawing.Font("Arial", 16, FontStyle.Bold | FontStyle.Italic); // Встановлюємо шрифт, стиль жирний + курсив

label1.ForeColor = Color.Pink; // Встановлюємо колір тексту рожевим

label1.BackColor = Color.Green; // Встановлюємо зелений колір фону

label1.TextAlign = ContentAlignment.MiddleLeft; // Вирівнювання тексту зліва

label1.Dock = DockStyle.Top; // Розташування елемента зверху форми

// Створюємо другий рядок

System.Windows.Forms.Label label2 = new System.Windows.Forms.Label();

label2.Text = "Другий рядок"; // Встановлюємо текст для другого рядка

label2.Font = new System.Drawing.Font("Arial", 14, FontStyle.Italic | FontStyle.Underline); // Шрифт курсив + підкреслення

label2.ForeColor = Color.Blue; // Встановлюємо синій колір тексту

label2.BackColor = Color.Red; // Встановлюємо червоний колір фону

label2.TextAlign = ContentAlignment.MiddleRight; // Вирівнювання тексту справа

label2.Dock = DockStyle.Top; // Розташування елемента зверху форми

// Створюємо третій рядок

System.Windows.Forms.Label label3 = new System.Windows.Forms.Label();

label3.Text = "Третій рядок"; // Встановлюємо текст для третього рядка

label3.Font = new System.Drawing.Font("Arial", 17, FontStyle.Bold | FontStyle.Italic | FontStyle.Underline); // Стиль жирний + курсив + підкреслення

label3.ForeColor = Color.LightBlue; // Встановлюємо світло-блакитний колір тексту

label3.BackColor = Color.Yellow; // Встановлюємо жовтий колір фону

label3.TextAlign = ContentAlignment.MiddleCenter; // Вирівнювання тексту по центру

label3.Dock = DockStyle.Top; // Розташування елемента зверху форми

// Додаємо рядки до форми у зворотньому порядку для правильного відображення

this.Controls.Add(label3); // Додаємо третій рядок

this.Controls.Add(label2); // Додаємо другий рядок

this.Controls.Add(label1); // Додаємо перший рядок

}

[STAThread] // Атрибут для запуску програми в однопоточному режимі

public static void Main() // Основний метод програми

{

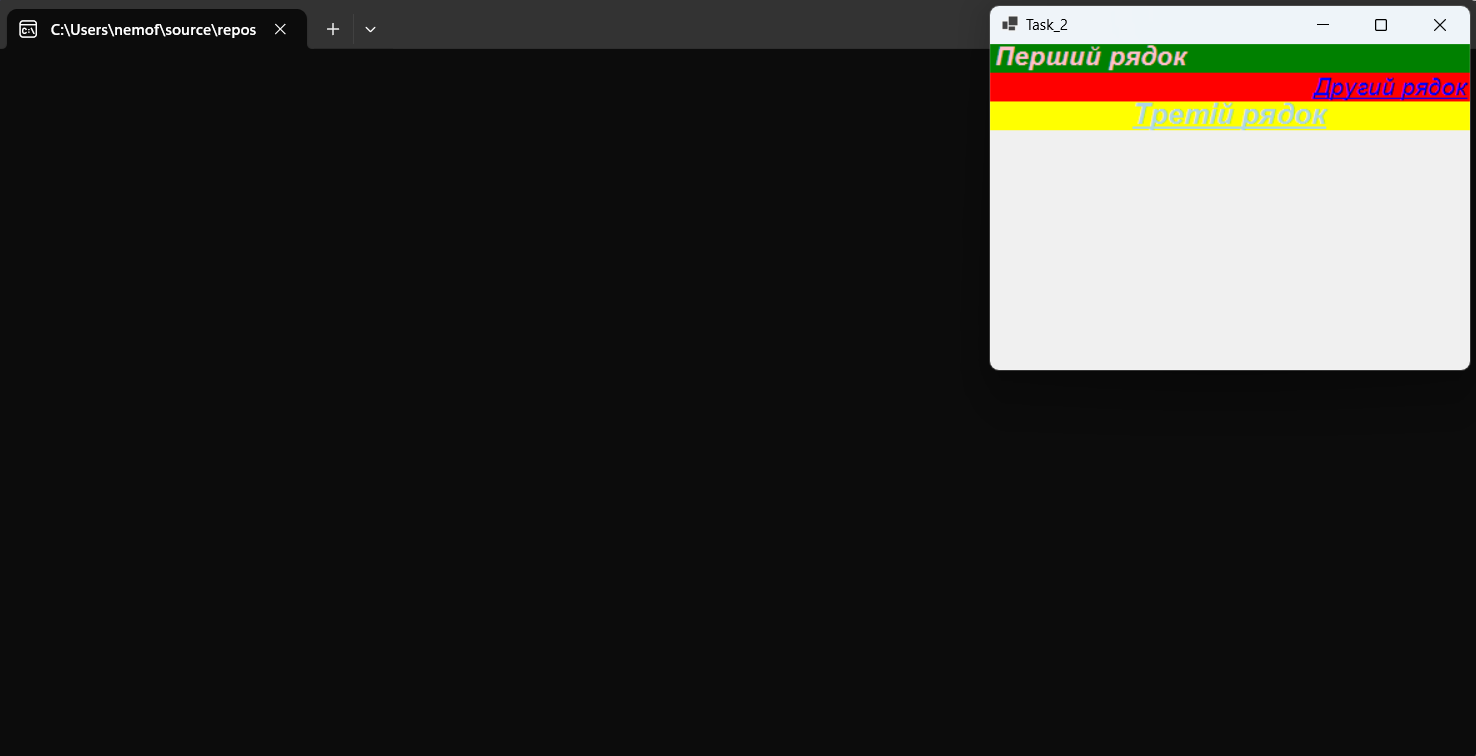
Application.EnableVisualStyles(); // Включає візуальні стилі для елементів

System.Windows.Forms.Application.Run(new Task\_2()); // Запускає форму Task\_2

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Ініціалізація форми**:
   * Створюється клас Task\_2, який наслідує клас Form. У конструкторі форми встановлюється її заголовок і розмір.
   * Використовується обробник події завантаження форми (Form\_Load), який додає елементи управління (три рядки тексту).
2. **Додавання рядків**:
   * У методі Form\_Load створюються три об'єкти класу Label, кожен з яких відповідає за свій рядок тексту.
   * Налаштовуються наступні властивості для кожного рядка:
     + Текст.
     + Шрифт та стиль (жирний, курсив, підкреслення).
     + Колір тексту та фону.
     + Вирівнювання тексту (по лівому, правому краю або по центру).
     + Розташування елемента у формі за допомогою властивості DockStyle.
3. **Додавання елементів у форму**:
   * Три рядки тексту додаються у форму з використанням методу this.Controls.Add, причому у зворотному порядку, щоб вони правильно відображалися зверху вниз.
4. **Запуск програми**:
   * Програма запускається з використанням методу Application.Run(), який створює вікно з назвою Task\_2.

**Функціональність:**

Програма створює вікно з трьома рядками тексту, кожен з яких має різний стиль:

1. Перший рядок тексту ("Перший рядок") має рожевий текст на зеленому фоні, вирівняний по лівому краю, жирний курсивний шрифт.
2. Другий рядок тексту ("Другий рядок") має синій текст на червоному фоні, вирівняний по правому краю, курсивний шрифт із підкресленням.
3. Третій рядок тексту ("Третій рядок") має світло-блакитний текст на жовтому фоні, вирівняний по центру, жирний курсивний шрифт із підкресленням.

**Висновок:**

Даний застосунок продемонстрував можливість динамічного додавання елементів управління до форми програмним шляхом під час її завантаження. Це дозволяє налаштовувати вміст форми залежно від потреб програми. Користувач може змінювати властивості тексту, стилю та вирівнювання, що робить даний підхід гнучким і масштабованим для розробки інтерфейсів з різноманітними налаштуваннями.

**Завдання 3**

Виконати перше завдання, використавши замість компоненту Label компонент **TextBox**.

**Текст програми:**

using System; // Підключення простору імен для базових класів

using System.Drawing; // Підключення простору імен для графіки та роботи з кольорами

using System.Windows.Forms; // Підключення простору імен для створення віконних додатків

// Оголошення класу Task\_3, який наслідується від класу Form

public class Task\_3 : Form

{

// Конструктор класу Task\_3

public Task\_3()

{

// Встановлення заголовка форми

this.Text = "Task\_3";

// Встановлення розміру форми

this.Size = new Size(400, 300);

// Прив'язка обробника події завантаження форми

this.Load += new EventHandler(Form\_Load);

}

// Метод, який викликається під час завантаження форми

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

{

// Створення першого текстового поля

TextBox textBox1 = new TextBox();

// Встановлення тексту в першому рядку

textBox1.Text = "Перший рядок";

// Налаштування шрифту: Arial, 16 розмір, жирний і курсив

textBox1.Font = new Font("Arial", 16, FontStyle.Bold | FontStyle.Italic);

// Колір тексту - рожевий

textBox1.ForeColor = Color.Pink;

// Колір фону - зелений

textBox1.BackColor = Color.Green;

// Вирівнювання тексту ліворуч

textBox1.TextAlign = HorizontalAlignment.Left;

// Прив'язка елемента до верхнього краю форми

textBox1.Dock = DockStyle.Top;

// Встановлення поля лише для читання

textBox1.ReadOnly = true;

// Створення другого текстового поля

TextBox textBox2 = new TextBox();

// Встановлення тексту у другому рядку

textBox2.Text = "Другий рядок";

// Налаштування шрифту: Arial, 14 розмір, курсив і підкреслений

textBox2.Font = new Font("Arial", 14, FontStyle.Italic | FontStyle.Underline);

// Колір тексту - синій

textBox2.ForeColor = Color.Blue;

// Колір фону - червоний

textBox2.BackColor = Color.Red;

// Вирівнювання тексту праворуч

textBox2.TextAlign = HorizontalAlignment.Right;

// Прив'язка елемента до верхнього краю форми

textBox2.Dock = DockStyle.Top;

// Встановлення поля лише для читання

textBox2.ReadOnly = true;

// Створення третього текстового поля

TextBox textBox3 = new TextBox();

// Встановлення тексту в третьому рядку

textBox3.Text = "Третій рядок";

// Налаштування шрифту: Arial, 17 розмір, жирний, курсив, підкреслений

textBox3.Font = new Font("Arial", 17, FontStyle.Bold | FontStyle.Italic | FontStyle.Underline);

// Колір тексту - світло-блакитний

textBox3.ForeColor = Color.LightBlue;

// Колір фону - жовтий

textBox3.BackColor = Color.Yellow;

// Вирівнювання тексту по центру

textBox3.TextAlign = HorizontalAlignment.Center;

// Прив'язка елемента до верхнього краю форми

textBox3.Dock = DockStyle.Top;

// Встановлення поля лише для читання

textBox3.ReadOnly = true;

// Додавання третього текстового поля до форми

this.Controls.Add(textBox3);

// Додавання другого текстового поля до форми

this.Controls.Add(textBox2);

// Додавання першого текстового поля до форми

this.Controls.Add(textBox1);

}

// Головний метод для запуску програми

[STAThread]

public static void Main()

{

// Увімкнення візуальних стилів

Application.EnableVisualStyles();

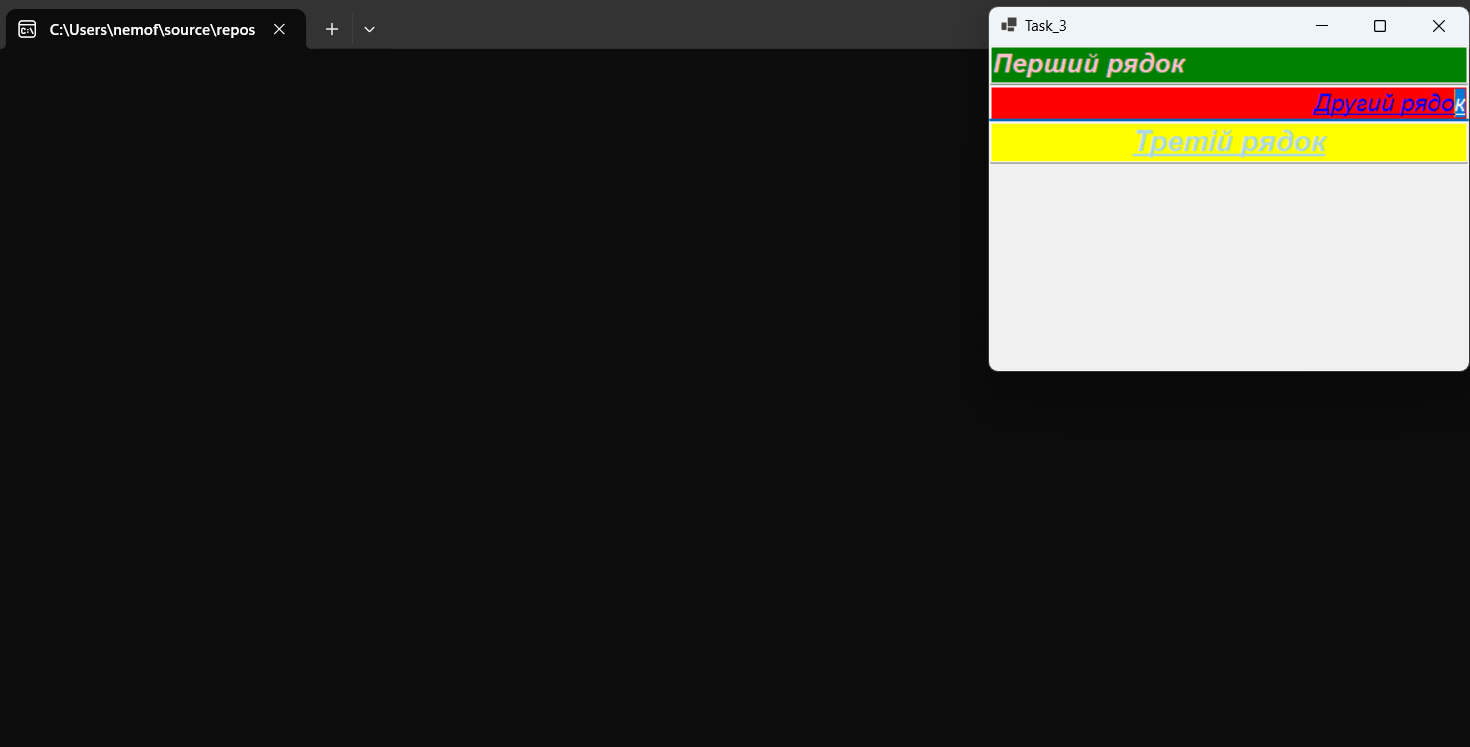
// Запуск програми з головною формою Task\_3

Application.Run(new Task\_3());

}

}

**Результат роботи програми:**



**Алгоритм побудови коду**

1. **Створення форми**:
   * Використовуємо клас Form для створення віконної форми.
   * Заголовок форми та її розмір встановлюються через властивості Text та Size відповідно.
   * Додається обробник події завантаження форми (Form\_Load), який містить логіку для додавання компонентів на форму.
2. **Створення текстових полів**:
   * Використовуємо компонент TextBox для кожного текстового рядка.
   * Налаштовуємо текст, стиль шрифту, вирівнювання, колір тексту та фону.
   * Поля встановлюються лише для читання через властивість ReadOnly.
   * Елементи TextBox додаються до форми, використовуючи метод Controls.Add().
3. **Стилізація компонентів**:
   * Для кожного текстового поля встановлюються індивідуальні стилі шрифту: розмір, жирність, курсив, підкреслення.
   * Колір тексту та фону для кожного текстового поля налаштовується через відповідні властивості (ForeColor, BackColor).
   * Вирівнювання тексту (ліворуч, по центру, праворуч) реалізується через властивість TextAlign.
4. **Головний метод**:
   * Головний метод Main() запускає програму з увімкненими візуальними стилями через Application.EnableVisualStyles() та створює екземпляр форми Task\_3.

**Функціональність**

Код дозволяє створити форму з трьома текстовими полями (TextBox), в яких відображаються три рядки тексту з різними стилями:

* Перший рядок тексту: шрифт розміром 16, рожевий текст на зеленому фоні, жирний і курсивний стиль, вирівняний ліворуч.
* Другий рядок тексту: шрифт розміром 14, синій текст на червоному фоні, курсив і підкреслення, вирівняний праворуч.
* Третій рядок тексту: шрифт розміром 17, світло-блакитний текст на жовтому фоні, жирний, курсивний та підкреслений, вирівняний по центру.

Текстові поля налаштовуються так, щоб користувач не міг їх редагувати.

**Висновок**

У цьому завданні компонент Label було замінено на TextBox. Це дозволило додати більше функціональності, такої як можливість прокручування тексту при його перевищенні розміру текстового поля. Задача показала можливість стилізації текстових полів та управління їхніми властивостями для досягнення бажаного зовнішнього вигляду і поведінки.

**Завдання 4\***

Виконати минуле завдання додавши текстові поля програмним шляхом, тобто під час завантаження вікна застосунку.

**Текст програми:**

using System; // Підключення простору імен System для базових функцій

using System.Drawing; // Підключення простору імен для роботи з графікою

using System.Windows.Forms; // Підключення простору імен для створення форми

// Створення класу Task\_4, який наслідує клас Form

public class Task\_4 : Form

{

// Конструктор класу Task\_4

public Task\_4()

{

// Встановлюємо заголовок форми

this.Text = "Task\_4";

// Встановлюємо розмір форми

this.Size = new Size(400, 300);

// Додаємо обробник події завантаження форми

this.Load += new EventHandler(Form\_Load);

}

// Метод, який виконується при завантаженні форми

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

{

// Створюємо текстове поле textBox1

TextBox textBox1 = new TextBox();

// Встановлюємо текст для першого рядка

textBox1.Text = "Перший рядок";

// Встановлюємо шрифт Arial 16, напівжирний і курсивний

textBox1.Font = new Font("Arial", 16, FontStyle.Bold | FontStyle.Italic);

// Встановлюємо колір тексту рожевим

textBox1.ForeColor = Color.Pink;

// Встановлюємо колір фону зеленим

textBox1.BackColor = Color.Green;

// Вирівнюємо текст по лівому краю

textBox1.TextAlign = HorizontalAlignment.Left;

// Закріплюємо поле зверху форми

textBox1.Dock = DockStyle.Top;

// Робимо поле тільки для читання

textBox1.ReadOnly = true;

// Створюємо текстове поле textBox2

TextBox textBox2 = new TextBox();

// Встановлюємо текст для другого рядка

textBox2.Text = "Другий рядок";

// Встановлюємо шрифт Arial 14, курсивний і підкреслений

textBox2.Font = new Font("Arial", 14, FontStyle.Italic | FontStyle.Underline);

// Встановлюємо колір тексту синім

textBox2.ForeColor = Color.Blue;

// Встановлюємо колір фону червоним

textBox2.BackColor = Color.Red;

// Вирівнюємо текст по правому краю

textBox2.TextAlign = HorizontalAlignment.Right;

// Закріплюємо поле зверху форми (над попереднім)

textBox2.Dock = DockStyle.Top;

// Робимо поле тільки для читання

textBox2.ReadOnly = true;

// Створюємо текстове поле textBox3

TextBox textBox3 = new TextBox();

// Встановлюємо текст для третього рядка

textBox3.Text = "Третій рядок";

// Встановлюємо шрифт Arial 17, напівжирний, курсивний і підкреслений

textBox3.Font = new Font("Arial", 17, FontStyle.Bold | FontStyle.Italic | FontStyle.Underline);

// Встановлюємо колір тексту світло-блакитним

textBox3.ForeColor = Color.LightBlue;

// Встановлюємо колір фону жовтим

textBox3.BackColor = Color.Yellow;

// Вирівнюємо текст по центру

textBox3.TextAlign = HorizontalAlignment.Center;

// Закріплюємо поле зверху форми (над попередніми)

textBox3.Dock = DockStyle.Top;

// Робимо поле тільки для читання

textBox3.ReadOnly = true;

// Додаємо всі три поля до форми

this.Controls.Add(textBox3);

this.Controls.Add(textBox2);

this.Controls.Add(textBox1);

}

// Метод Main, який є точкою входу в програму

[STAThread]

public static void Main()

{

// Увімкнення візуальних стилів додатку

Application.EnableVisualStyles();

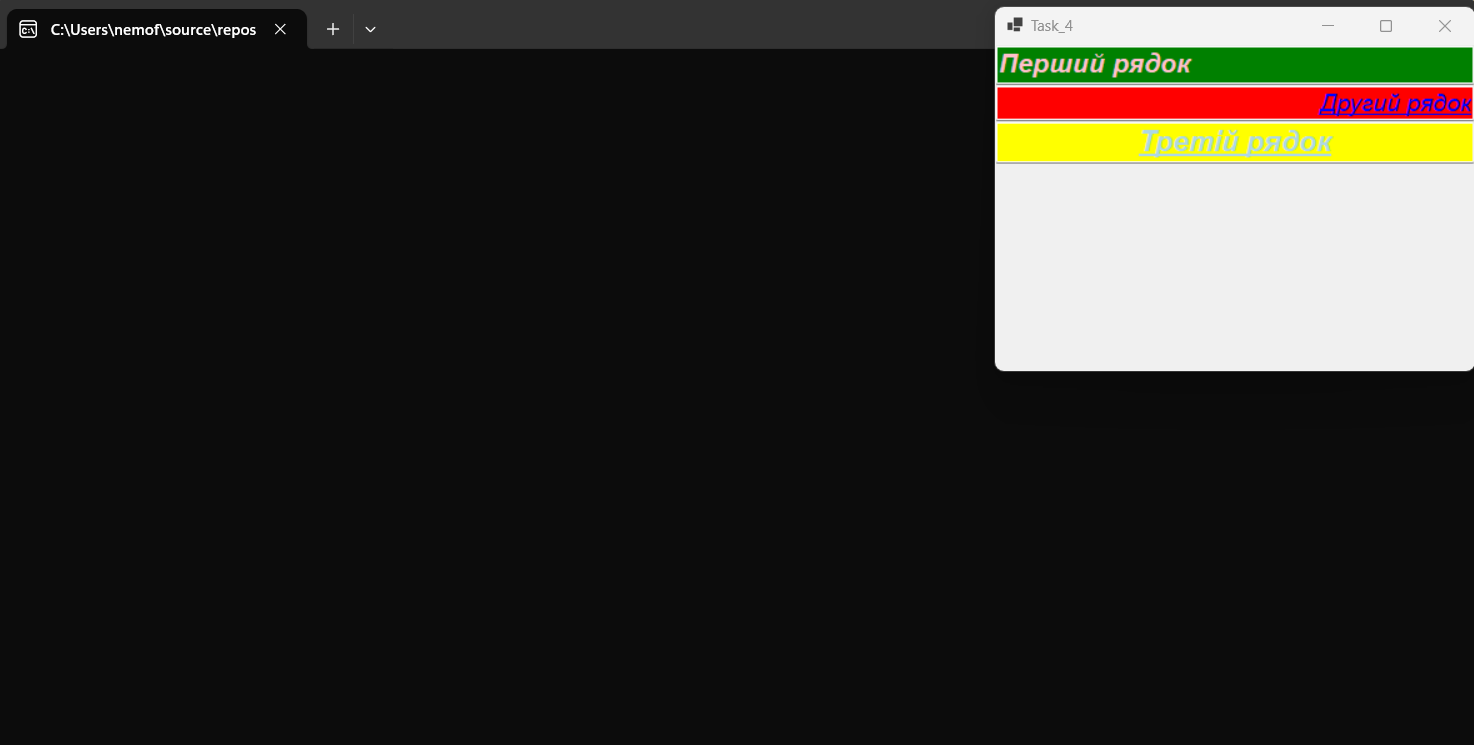
// Запуск форми Task\_4

Application.Run(new Task\_4());

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Створення форми**: Визначається клас Task\_4, який успадковує клас Form, щоб створити вікно застосунку.
2. **Конструктор форми**: У конструкторі задаються заголовок форми та її розмір. Додається обробник події завантаження вікна — метод Form\_Load.
3. **Метод Form\_Load**:
   * У цьому методі створюються три текстові поля (TextBox), які будуть додані до форми під час її завантаження.
   * Для кожного текстового поля встановлюється:
     + Текст.
     + Шрифт із певними параметрами (стиль, розмір).
     + Колір тексту та фону.
     + Вирівнювання тексту.
     + Положення текстового поля на формі (за допомогою DockStyle.Top).
     + Поля робляться доступними лише для читання (ReadOnly).
4. **Додавання текстових полів до форми**: Текстові поля додаються в форму зверху вниз.
5. **Точка входу в програму**: У методі Main увімкнено візуальні стилі застосунку і запускається форма Task\_4.

**Функціональність:**

При завантаженні вікна застосунку автоматично створюються три текстові поля:

* **Перше текстове поле** має рожевий текст на зеленому фоні, шрифт Arial 16 (напівжирний і курсивний), і вирівнювання тексту по лівому краю.
* **Друге текстове поле** має синій текст на червоному фоні, шрифт Arial 14 (курсивний і підкреслений), вирівнювання тексту по правому краю.
* **Третє текстове поле** має світло-блакитний текст на жовтому фоні, шрифт Arial 17 (напівжирний, курсивний і підкреслений), вирівнювання по центру.

Всі поля є тільки для читання.

**Висновок:**

Ця програма демонструє базові можливості роботи з графічним інтерфейсом у C# за допомогою бібліотеки WinForms. Використовуючи програмне створення елементів форми, зручно динамічно змінювати їх властивості та поведінку, що робить цей підхід гнучким для розробки інтерактивних інтерфейсів.

**Завдання 5**

Використовуючи компонент PictureBox та його відповідні властивості створити форму, що містить гpyпy будь-яких зображень за топологією згідно із параметрами свого варіанту (враховувати приблизні співвідношення, а не абсолютні розміри)

|  |  |
| --- | --- |
| №  варіанту | Форма топології |
| 1 | 2 |
| 12 |  |

|  |  |
| --- | --- |
| №  варіанту | Форма топології |
| 1 | 2 |
| 12 |  |

**Текст програми:**

using System; // Підключення простору імен System, що містить базові класи .NET

using System.Drawing; // Підключення простору імен для роботи з графікою та зображеннями

using System.Windows.Forms; // Підключення простору імен для створення та роботи з формами Windows Forms

namespace WinFormsApp1 // Оголошення простору імен для цього додатку

{

public partial class Form1 : Form // Оголошення класу Form1, який є частковим класом і успадковується від класу Form

{

public Form1() // Конструктор класу Form1

{

InitializeComponent(); // Ініціалізація компонентів форми

InitializePictureBoxes(); // Виклик методу для ініціалізації та додавання PictureBox елементів на форму

}

private void InitializePictureBoxes() // Метод для ініціалізації та розміщення PictureBox елементів

{

// Перша топологія (горизонтальна лінія)

PictureBox pictureBox1 = new PictureBox(); // Створення PictureBox для першого зображення

pictureBox1.Image = Image.FromFile(@"C:\1.jpg"); // Завантаження зображення з файлу

pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox1.Location = new Point(10, 60); // Встановлення положення PictureBox на формі

pictureBox1.Size = new Size(100, 50); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox1); // Додавання PictureBox на форму

PictureBox pictureBox2 = new PictureBox(); // Створення PictureBox для другого зображення

pictureBox2.Image = Image.FromFile(@"C:\2.jpg"); // Завантаження другого зображення з файлу

pictureBox2.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox2.Location = new Point(120, 60); // Встановлення положення PictureBox на формі

pictureBox2.Size = new Size(100, 50); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox2); // Додавання PictureBox на форму

PictureBox pictureBox3 = new PictureBox(); // Створення PictureBox для третього зображення

pictureBox3.Image = Image.FromFile(@"C:\3.jpg"); // Завантаження третього зображення з файлу

pictureBox3.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox3.Location = new Point(230, 40); // Встановлення положення PictureBox на формі

pictureBox3.Size = new Size(100, 70); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox3); // Додавання PictureBox на форму

PictureBox pictureBox4 = new PictureBox(); // Створення PictureBox для четвертого зображення

pictureBox4.Image = Image.FromFile(@"C:\4.jpg"); // Завантаження четвертого зображення з файлу

pictureBox4.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox4.Location = new Point(340, 25); // Встановлення положення PictureBox на формі

pictureBox4.Size = new Size(100, 85); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox4); // Додавання PictureBox на форму

PictureBox pictureBox5 = new PictureBox(); // Створення PictureBox для п'ятого зображення

pictureBox5.Image = Image.FromFile(@"C:\5.jpg"); // Завантаження п'ятого зображення з файлу

pictureBox5.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox5.Location = new Point(450, 10); // Встановлення положення PictureBox на формі

pictureBox5.Size = new Size(100, 100); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox5); // Додавання PictureBox на форму

// Друга топологія (різні розміри)

PictureBox pictureBox6 = new PictureBox(); // Створення PictureBox для шостого зображення

pictureBox6.Image = Image.FromFile(@"C:\6.jpg"); // Завантаження шостого зображення з файлу

pictureBox6.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox6.Location = new Point(10, 120); // Встановлення положення PictureBox на формі

pictureBox6.Size = new Size(100, 120); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox6); // Додавання PictureBox на форму

PictureBox pictureBox7 = new PictureBox(); // Створення PictureBox для сьомого зображення

pictureBox7.Image = Image.FromFile(@"C:\7.jpg"); // Завантаження сьомого зображення з файлу

pictureBox7.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox7.Location = new Point(120, 160); // Встановлення положення PictureBox на формі

pictureBox7.Size = new Size(200, 80); // Встановлення розміру PictureBox

this.Controls.Add(pictureBox7); // Додавання PictureBox на форму

PictureBox pictureBox8 = new PictureBox(); // Створення PictureBox для восьмого зображення

pictureBox8.Image = Image.FromFile(@"C:\8.jpg"); // Завантаження восьмого зображення з файлу

pictureBox8.SizeMode = PictureBoxSizeMode.StretchImage; // Встановлення режиму відображення - зображення буде розтягуватись

pictureBox8.Location = new Point(340, 180); // Встановлення положення PictureBox на формі

pictureBox8.Size = new Size(200, 60); // Встановлення розміру PictureBox

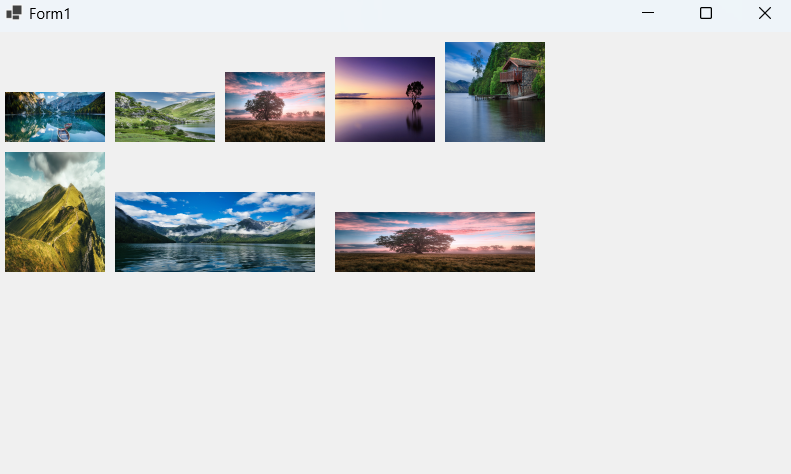
this.Controls.Add(pictureBox8); // Додавання PictureBox на форму

}

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Ініціалізація компонентів:**
   * Створюється новий клас форми Form1, який успадковується від класу Form.
   * В конструкторі форми викликається метод InitializeComponent() для базової ініціалізації форми.
   * Додається виклик методу InitializePictureBoxes(), який містить логіку додавання зображень до форми.
2. **Додавання зображень:**
   * Створюються об'єкти PictureBox для кожного зображення.
   * Використовується метод Image.FromFile(), щоб завантажити зображення з вказаного шляху.
   * Для кожного зображення встановлюється режим відображення StretchImage, щоб зображення масштабувалось до розмірів PictureBox.
   * Вказується розташування (Location) і розмір (Size) кожного елемента на формі.
   * Зображення додаються на форму через this.Controls.Add().
3. **Топології:**
   * Перша група зображень розташовується горизонтально.
   * Друга група зображень має різні розміри та різне розташування на формі.

**Функціональність:**

* Форма відображає групу зображень, організованих у двох топологіях: горизонтальній лінії і зображеннях різного розміру. Кожне зображення завантажується із файлів і розташовується на формі відповідно до встановлених координат і розмірів.
* Зображення автоматично масштабуються до розмірів контейнерів PictureBox, забезпечуючи коректне відображення незалежно від вихідних розмірів зображення.

**Висновок:**

Дане рішення успішно реалізує завдання із створенням групи зображень за допомогою компоненту PictureBox. Код надає гнучкість у розміщенні та налаштуванні зображень, дозволяючи змінювати їхні розміри і позиції на формі.

**Завдання 6\***

Виконати минуле завдання додавши зображення програмним шляхом, тобто під час завантаження вікна застосунку.

**Текст програми:**

using System; // Підключення простору імен System для основних функцій

using System.Drawing; // Підключення простору імен для роботи з графікою (зображення)

using System.Windows.Forms; // Підключення простору імен для створення форм і елементів управління

namespace Task\_6 // Визначення простору імен Task\_6 для унікальності класів

{

public partial class Form1 : Form // Оголошення часткового класу Form1, що успадковує форму (Form)

{

public Form1() // Конструктор класу Form1

{

InitializeComponent(); // Ініціалізація компонентів форми

this.Load += new EventHandler(Form1\_Load); // Додавання обробника події завантаження форми

}

private void Form1\_Load(object sender, EventArgs e) // Метод, що викликається при завантаженні форми

{

InitializePictureBoxes(); // Виклик методу для ініціалізації PictureBox елементів

}

private void InitializePictureBoxes() // Метод для додавання PictureBox елементів на форму

{

// Перша топологія (горизонтальна лінія)

PictureBox pictureBox1 = new PictureBox(); // Створення першого PictureBox

pictureBox1.Image = Image.FromFile(@"C:\1.jpg"); // Завантаження зображення з файлу

pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage; // Режим масштабування зображення до розміру PictureBox

pictureBox1.Location = new Point(10, 60); // Позиція PictureBox на формі

pictureBox1.Size = new Size(100, 50); // Розмір PictureBox

this.Controls.Add(pictureBox1); // Додавання PictureBox до елементів форми

PictureBox pictureBox2 = new PictureBox(); // Створення другого PictureBox

pictureBox2.Image = Image.FromFile(@"C:\2.jpg"); // Завантаження зображення з файлу

pictureBox2.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox2.Location = new Point(120, 60); // Позиція PictureBox

pictureBox2.Size = new Size(100, 50); // Розмір PictureBox

this.Controls.Add(pictureBox2); // Додавання PictureBox до форми

PictureBox pictureBox3 = new PictureBox(); // Створення третього PictureBox

pictureBox3.Image = Image.FromFile(@"C:\3.jpg"); // Завантаження зображення

pictureBox3.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox3.Location = new Point(230, 40); // Позиція PictureBox

pictureBox3.Size = new Size(100, 70); // Розмір PictureBox

this.Controls.Add(pictureBox3); // Додавання PictureBox до форми

PictureBox pictureBox4 = new PictureBox(); // Створення четвертого PictureBox

pictureBox4.Image = Image.FromFile(@"C:\4.jpg"); // Завантаження зображення

pictureBox4.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox4.Location = new Point(340, 25); // Позиція PictureBox

pictureBox4.Size = new Size(100, 85); // Розмір PictureBox

this.Controls.Add(pictureBox4); // Додавання PictureBox до форми

PictureBox pictureBox5 = new PictureBox(); // Створення п'ятого PictureBox

pictureBox5.Image = Image.FromFile(@"C:\5.jpg"); // Завантаження зображення

pictureBox5.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox5.Location = new Point(450, 10); // Позиція PictureBox

pictureBox5.Size = new Size(100, 100); // Розмір PictureBox

this.Controls.Add(pictureBox5); // Додавання PictureBox до форми

// Друга топологія (різні розміри)

PictureBox pictureBox6 = new PictureBox(); // Створення шостого PictureBox

pictureBox6.Image = Image.FromFile(@"C:\6.jpg"); // Завантаження зображення

pictureBox6.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox6.Location = new Point(10, 120); // Позиція PictureBox

pictureBox6.Size = new Size(100, 120); // Розмір PictureBox

this.Controls.Add(pictureBox6); // Додавання PictureBox до форми

PictureBox pictureBox7 = new PictureBox(); // Створення сьомого PictureBox

pictureBox7.Image = Image.FromFile(@"C:\7.jpg"); // Завантаження зображення

pictureBox7.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox7.Location = new Point(120, 160); // Позиція PictureBox

pictureBox7.Size = new Size(200, 80); // Розмір PictureBox

this.Controls.Add(pictureBox7); // Додавання PictureBox до форми

PictureBox pictureBox8 = new PictureBox(); // Створення восьмого PictureBox

pictureBox8.Image = Image.FromFile(@"C:\8.jpg"); // Завантаження зображення

pictureBox8.SizeMode = PictureBoxSizeMode.StretchImage; // Масштабування зображення

pictureBox8.Location = new Point(340, 180); // Позиція PictureBox

pictureBox8.Size = new Size(200, 60); // Розмір PictureBox

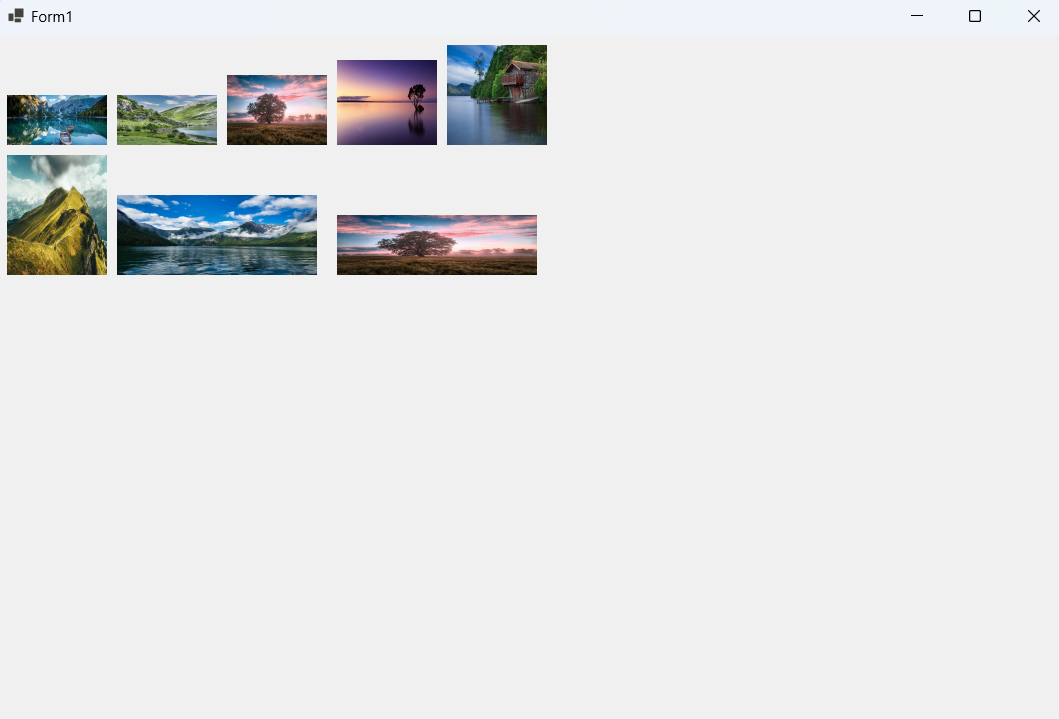
this.Controls.Add(pictureBox8); // Додавання PictureBox до форми

}

}

}

**Результат роботи програми:**



**Алгоритм побудови коду**

1. **Створення форми та підключення необхідних просторів імен**: Для роботи з графікою та елементами керування використовується простір імен System.Drawing та System.Windows.Forms.
2. **Створення конструктора форми**: У конструкторі форми Form1 налаштовується обробник події Form1\_Load, який викликається під час завантаження вікна.
3. **Додавання методу для обробки завантаження форми**: У методі Form1\_Load викликається спеціальний метод InitializePictureBoxes(), який відповідає за створення та розміщення зображень на формі.
4. **Ініціалізація елементів PictureBox**: Метод InitializePictureBoxes створює кілька об'єктів PictureBox, завантажує зображення з файлів і задає властивості для кожного елемента. Зображення масштабуються відповідно до розміру PictureBox, використовуючи властивість SizeMode.
5. **Додавання PictureBox до форми**: Всі створені елементи додаються до колекції керування форми (this.Controls.Add()), щоб вони відображалися під час запуску застосунку.

**Функціональність**

* При завантаженні форми, кожен елемент PictureBox ініціалізується і завантажує зображення з вказаних шляхів.
* Зображення автоматично масштабуються до розміру контейнера та правильно позиціонуються на формі.
* Для різних елементів PictureBox використовуються різні позиції та розміри, що створює дві окремі топології розміщення.

**Висновок**

Реалізація програмного додавання зображень до форми дозволяє гнучко керувати графічними елементами в інтерфейсі користувача. Завантаження зображень під час ініціалізації форми полегшує керування контентом та забезпечує динамічну роботу з графікою в C# застосунках.

**Завдання 7**

Використовуючи компоненти Button, TableLayoutPanel та відповідні властивості створити форму, що містить гpyпy кнопок за топологіею згідно із параметрами свого варіанту

|  |  |
| --- | --- |
| №  варіанту | Форма топології |
| 1 | 2 |
| 12 |  |

**Текст програми:**

using System; // Підключення простору імен для основних функцій

using System.Windows.Forms; // Підключення простору імен для роботи з формами

namespace Task\_7 // Оголошення простору імен Task\_7

{

public partial class Form1 : Form // Оголошення класу Form1, який є частковим та наслідується від класу Form

{

private Button but1; // Оголошення кнопки but1

private Button but2; // Оголошення кнопки but2

private Button but3; // Оголошення кнопки but3

private Button but4; // Оголошення кнопки but4

private Button but5; // Оголошення кнопки but5

private TableLayoutPanel tableLayoutPanel; // Оголошення компонента TableLayoutPanel для розміщення елементів

public Form1() // Конструктор класу Form1

{

InitializeComponent(); // Ініціалізація компонентів форми

// Створення TableLayoutPanel

tableLayoutPanel = new TableLayoutPanel

{

RowCount = 4, // Встановлення кількості рядків у таблиці (4 рядки)

ColumnCount = 3, // Встановлення кількості стовпців у таблиці (3 стовпці)

Dock = DockStyle.Fill // Автоматичне заповнення форми TableLayoutPanel

};

// Додавання рядків та стовпців

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Перший рядок: займає 25% висоти для but5

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Другий рядок: 25% висоти для but1

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Третій рядок: 25% для but3, but2, but4

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Четвертий рядок: порожній, займає 25% висоти

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Лівий стовпець, 33.33% ширини

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Середній стовпець, 33.33% ширини

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Правий стовпець, 33.33% ширини

// Створення кнопок

but1 = new Button { Text = "but1" }; // Ініціалізація кнопки but1 з текстом "but1"

but2 = new Button { Text = "but2" }; // Ініціалізація кнопки but2 з текстом "but2"

but3 = new Button { Text = "but3" }; // Ініціалізація кнопки but3 з текстом "but3"

but4 = new Button { Text = "but4" }; // Ініціалізація кнопки but4 з текстом "but4"

but5 = new Button { Text = "but5" }; // Ініціалізація кнопки but5 з текстом "but5"

// Додавання кнопок до TableLayoutPanel

tableLayoutPanel.Controls.Add(but5, 1, 0); // Додавання кнопки but5 у верхній ряд, середній стовпець

tableLayoutPanel.Controls.Add(but1, 1, 1); // Додавання кнопки but1 у другий ряд, середній стовпець

tableLayoutPanel.Controls.Add(but3, 0, 2); // Додавання кнопки but3 у третій ряд, лівий стовпець

tableLayoutPanel.Controls.Add(but2, 1, 2); // Додавання кнопки but2 у третій ряд, середній стовпець

tableLayoutPanel.Controls.Add(but4, 2, 2); // Додавання кнопки but4 у третій ряд, правий стовпець

// Додавання TableLayoutPanel до форми

this.Controls.Add(tableLayoutPanel); // Додає TableLayoutPanel до колекції елементів управління форми

}

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Імпорт бібліотек**: Спочатку підключаються простори імен System та System.Windows.Forms, які потрібні для роботи з основними функціями та формами у C#.
2. **Оголошення форми**: Створюється клас Form1, який наслідується від базового класу Form. Це забезпечує функціонал для створення віконних додатків.
3. **Створення елементів**:
   * Оголошуються змінні для п'яти кнопок (Button) та таблиці для їх розміщення (TableLayoutPanel).
4. **Ініціалізація компонентів**:
   * У конструкторі класу формується таблиця з 4 рядків та 3 стовпців. Кожен рядок і стовпець займає певний відсоток від доступного простору.
   * Створюються кнопки з текстом, який відображатиметься на кожній з них.
5. **Розміщення кнопок**:
   * Кнопки додаються до таблиці відповідно до заданої топології. Наприклад, кнопка but5 розміщується в верхньому ряді по центру, а but1 — у другому ряді по центру.
6. **Додавання таблиці до форми**: Таблиця з кнопками додається до форми, і вона автоматично заповнює доступний простір.

**Функціональність:**

Цей код створює форму, яка містить групу з 5 кнопок, розташованих в певній конфігурації за допомогою компонента TableLayoutPanel. Кожна кнопка має текстову мітку, яка відображається в інтерфейсі. Кнопки розміщені у сітці, де кожен рядок і стовпець займає фіксований відсоток простору, що дозволяє гнучко налаштовувати інтерфейс при зміні розміру вікна.

**Висновок:**

Завдання дозволило продемонструвати базові навички роботи з графічним інтерфейсом у C#, зокрема, використання компонента TableLayoutPanel для структурованого розміщення елементів. Використання сіткової розмітки дає можливість створювати адаптивний інтерфейс, що автоматично підлаштовується під розмір вікна.

**Завдання 8\***

Виконати минуле завдання додавши кнопки та таблицю розміщення програмним шляхом, тобто під час завантаження вікна застосунку

**Текст програми:**

using System; // Підключення простору імен для загальних функцій

using System.Windows.Forms; // Підключення простору імен для створення форм та елементів керування

namespace Task\_8 // Оголошення простору імен Task\_8 для структурування коду

{

public partial class Form1 : Form // Оголошення класу Form1, який є частиною форми та успадковується від класу Form

{

private Button but1; // Оголошення кнопки but1

private Button but2; // Оголошення кнопки but2

private Button but3; // Оголошення кнопки but3

private Button but4; // Оголошення кнопки but4

private Button but5; // Оголошення кнопки but5

private TableLayoutPanel tableLayoutPanel; // Оголошення панелі TableLayoutPanel

public Form1() // Конструктор форми

{

InitializeComponent(); // Ініціалізація компонентів форми

this.Load += Form1\_Load; // Підписка на подію Load, яка викликається при завантаженні форми

}

private void Form1\_Load(object sender, EventArgs e) // Обробник події завантаження форми

{

// Створення TableLayoutPanel, таблиці для розміщення елементів керування

tableLayoutPanel = new TableLayoutPanel

{

RowCount = 4, // Задання кількості рядків у таблиці (4 рядки)

ColumnCount = 3, // Задання кількості стовпців у таблиці (3 стовпці)

Dock = DockStyle.Fill // Розтягування таблиці на весь доступний простір форми

};

// Додавання стилів для рядків таблиці

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Перший рядок займає 25% висоти для but5

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Другий рядок займає 25% висоти для but1

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Третій рядок займає 25% висоти для but3, but2, but4

tableLayoutPanel.RowStyles.Add(new RowStyle(SizeType.Percent, 25F)); // Четвертий рядок, порожній, займає 25% висоти

// Додавання стилів для стовпців таблиці

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Лівий стовпець займає 33.33% ширини

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Середній стовпець займає 33.33% ширини

tableLayoutPanel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33.33F)); // Правий стовпець займає 33.33% ширини

// Створення кнопок та задання тексту для них

but1 = new Button { Text = "but1" }; // Створення кнопки but1 з текстом "but1"

but2 = new Button { Text = "but2" }; // Створення кнопки but2 з текстом "but2"

but3 = new Button { Text = "but3" }; // Створення кнопки but3 з текстом "but3"

but4 = new Button { Text = "but4" }; // Створення кнопки but4 з текстом "but4"

but5 = new Button { Text = "but5" }; // Створення кнопки but5 з текстом "but5"

// Додавання кнопок до відповідних позицій у таблиці

tableLayoutPanel.Controls.Add(but5, 1, 0); // Додавання but5 у перший рядок, середній стовпець

tableLayoutPanel.Controls.Add(but1, 1, 1); // Додавання but1 у другий рядок, середній стовпець

tableLayoutPanel.Controls.Add(but3, 0, 2); // Додавання but3 у третій рядок, лівий стовпець

tableLayoutPanel.Controls.Add(but2, 1, 2); // Додавання but2 у третій рядок, середній стовпець

tableLayoutPanel.Controls.Add(but4, 2, 2); // Додавання but4 у третій рядок, правий стовпець

// Додавання TableLayoutPanel до форми для відображення на екрані

this.Controls.Add(tableLayoutPanel);

}

}

}

**Результат роботи програми:**



**Алгоритм побудови коду:**

1. **Створення форми**: Спочатку створюється форма з використанням класу Form, який успадковується для подальшої кастомізації.
2. **Оголошення компонентів**: Оголошуються кнопки (but1, but2, but3, but4, but5) і панель для розміщення (TableLayoutPanel).
3. **Подія завантаження форми**: Використовується подія Form1\_Load, яка викликається під час завантаження форми. У цьому обробнику подій програмно створюється таблиця та кнопки.
4. **Налаштування TableLayoutPanel**: Задається кількість рядків і стовпців (4 рядки та 3 стовпці). Кожен рядок і стовпець отримує процентне співвідношення, яке дозволяє рівномірно розподілити простір між елементами.
5. **Створення кнопок**: Програмно створюються кнопки та їм присвоюється текст.
6. **Розміщення кнопок**: Кнопки додаються в різні клітинки таблиці, що забезпечує правильне відображення на формі.
7. **Додавання панелі на форму**: Готовий елемент TableLayoutPanel додається на форму для відображення кнопок у зазначеному порядку.

**Функціональність програми:**

* Під час завантаження форми користувач побачить п’ять кнопок, розміщених у таблиці (TableLayoutPanel) з трьома стовпцями і чотирма рядками.
* Кнопки розташовані наступним чином:
  + but5 займає перший рядок у середньому стовпці.
  + but1 займає другий рядок у середньому стовпці.
  + but3, but2, but4 займають третій рядок у лівому, середньому та правому стовпцях відповідно.
* Структура кнопок і таблиці динамічно налаштовується під час завантаження форми, що дозволяє змінювати їх конфігурацію програмно.

**Висновок:**

Це завдання демонструє можливість динамічного створення та розміщення елементів інтерфейсу користувача (UI) в середовищі WinForms за допомогою C#. Програмне додавання компонентів під час завантаження форми дозволяє забезпечити гнучкість у розміщенні елементів та їх конфігурації, що може бути корисним для створення адаптивних інтерфейсів.

**Загальний Висновок**

Загальний висновок до лабораторної роботи №5 "Використання конструктора форм для створення та налаштування інтерфейсу віконних застосунків. Частина 1" полягає в наступному:

У ході виконання лабораторної роботи було глибоко досліджено і практично застосовано різноманітні компоненти та властивості конструктора форм у середовищі розробки Windows Forms. Це дозволило опанувати ключові аспекти створення та налаштування інтерфейсів віконних застосунків.

Були детально вивчені такі компоненти, як Label, TextBox, PictureBox, Button та TableLayoutPanel. Кожен з них відіграє важливу роль у побудові інтерактивних та зручних інтерфейсів користувача.

* **Label** дозволяє відображати статичний текст, який можна форматувати різними способами.
* **TextBox** надає можливість вводити та редагувати текст, а також налаштовувати його зовнішній вигляд.
* **PictureBox** використовується для відображення зображень, з можливістю їх масштабування та позиціонування.
* **Button** створює інтерактивні кнопки, які реагують на дії користувача.
* **TableLayoutPanel** забезпечує гнучке розміщення елементів у вигляді таблиці, що адаптується до розмірів вікна.

Крім того, було розглянуто різні підходи до додавання компонентів на форму: як безпосередньо у вікні дизайнера, так і програмно під час виконання застосунку. Це надає розробнику більшу гнучкість у керуванні інтерфейсом, дозволяючи динамічно змінювати його в залежності від логіки програми.

В цілому, виконання цієї лабораторної роботи значно розширило розуміння принципів побудови графічних інтерфейсів у Windows Forms. Отримані знання та навички дозволять створювати більш складні та функціональні застосунки з привабливим та зручним інтерфейсом користувача.