**Лабораторна робота №13. Використання засобів графіки Windows Forms.**

**Виконавець:** Лазоренко М.Д.

**Група виконавця:** КН-922в

**Мета роботи:** набути навички у створенні застосунків із використанням графічних можливостей фреймворку Windows Forms.

**Індивідуальні завдання**

**Варіант №9**

**Завдання 1**

Використовуючи компонент PictureBox та його графічні можливості, намалювати пpaпop України в місці компоненту, завданому власним варіантом (для оріснтацїі треба встановити компонентний контур)

|  |  |
| --- | --- |
| № варіанту | Відносна локація розташування пpaпopy на Picture Box |
| 1 | 2 |
| 9. |  |

**Зауваження:** рисунок мас бути створений за допомогою графічних примітивів, використання зовнішніх зображень чи фото ЗАБОРОНЯЄТЬСЯ

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task01

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void pictureBox1\_Paint(object sender, PaintEventArgs e)

{

// Отримуємо графічний контекст

Graphics g = e.Graphics;

// Визначаємо розміри PictureBox

int width = pictureBox1.ClientSize.Width;

int height = pictureBox1.ClientSize.Height;

// Розміри прапора

int flagWidth = width / 4;

int flagHeight = height / 6;

// Координати для лівого нижнього кута

int x = 10;

int y = height - flagHeight - 10;

// Малюємо синю частину прапора

g.FillRectangle(Brushes.Blue, x, y, flagWidth, flagHeight / 2);

// Малюємо жовту частину прапора

g.FillRectangle(Brushes.Yellow, x, y + flagHeight / 2, flagWidth, flagHeight / 2);

}

}

}

***Результат роботи програми:***

******

Програма реалізує відображення графічного об’єкта — прапора України в елементі PictureBox форми Windows Forms. Вона складається з класу Form1, який є головним класом програми. Ключові функції програми описані нижче:

1. **Ініціалізація форми:**
   * Під час створення об'єкта Form1 відбувається ініціалізація інтерфейсу користувача, зокрема елемента PictureBox, у якому буде відображатися прапор.
2. **Обробка події Paint для елемента PictureBox:**
   * Подія Paint обробляється методом pictureBox1\_Paint.
   * При виклику цього методу:
     + Отримується графічний контекст (Graphics) для малювання в області PictureBox.
     + Визначаються розміри області PictureBox, щоб розрахувати пропорції прапора.
3. **Розрахунок параметрів прапора:**
   * Розмір прапора визначається пропорційно до розмірів PictureBox:
     + Ширина прапора становить чверть ширини PictureBox.
     + Висота прапора становить одну шосту висоти PictureBox.
   * Координати верхнього лівого кута прапора визначаються так, щоб прапор був розташований у лівому нижньому куті PictureBox із відступами.
4. **Малювання прапора:**
   * Верхня половина прапора зафарбовується синім кольором (Brushes.Blue).
   * Нижня половина прапора зафарбовується жовтим кольором (Brushes.Yellow).
   * Для малювання використовується метод FillRectangle.
5. **Візуалізація:**
   * Після виконання коду вікно програми показує прапор України в лівому нижньому куті елемента PictureBox.

**Завдання 2**

Модифікувати минуле завдання так, щоб під час наведения курсору на компонент Picture Box поруч (ліворуч a6o праворуч в залежності від локації i бажання) із прапором України з'являвся пpaпop країни згідно до свого варіанту, а після видалення курсору— пpaпop даної країни зникав би.

|  |  |
| --- | --- |
| № варіанту | Відносна локація розташування пpaпopy на Picture Box |
| 1 | 2 |
| 9. |  |

**Зауваження:** рисунки мають бути створені за допомогою графічних примітивів, використання зовнішніх зображень чи фото ЗАБОРОНЯЄТЬСЯ.

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task02

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void pictureBoxUkraine\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малювання прапора України

int width = pictureBoxUkraine.ClientSize.Width;

int height = pictureBoxUkraine.ClientSize.Height;

g.FillRectangle(Brushes.Blue, 0, 0, width, height / 2);

g.FillRectangle(Brushes.Yellow, 0, height / 2, width, height / 2);

}

private void pictureBoxBulgaria\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малювання прапора Болгарії

int width = pictureBoxBulgaria.ClientSize.Width;

int height = pictureBoxBulgaria.ClientSize.Height;

g.FillRectangle(Brushes.White, 0, 0, width, height / 3);

g.FillRectangle(Brushes.Green, 0, height / 3, width, height / 3);

g.FillRectangle(Brushes.Red, 0, 2 \* height / 3, width, height / 3);

}

private void pictureBoxUkraine\_MouseEnter(object sender, EventArgs e)

{

// Показати PictureBox із прапором Болгарії

pictureBoxBulgaria.Visible = true;

}

private void pictureBoxUkraine\_MouseLeave(object sender, EventArgs e)

{

// Сховати PictureBox із прапором Болгарії

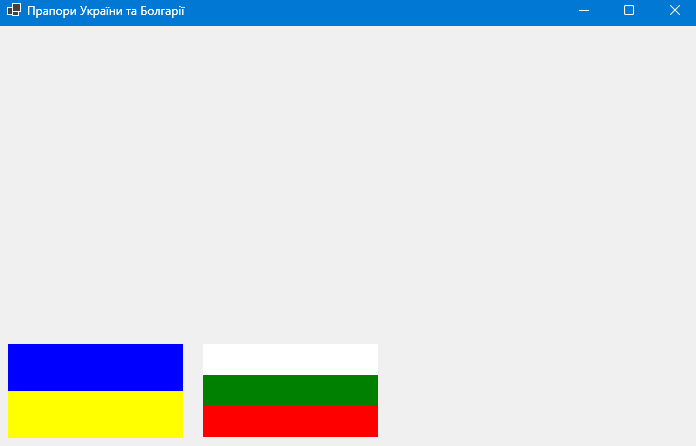
pictureBoxBulgaria.Visible = false;

}

}

}

***Результат роботи програми:***



Програма реалізує графічне відображення прапорів України та Болгарії з використанням елементів PictureBox у Windows Forms, а також взаємодію з користувачем через події миші. Програма складається з класу Form1, який є основним класом форми. Ключові функції програми описані нижче:

1. **Ініціалізація форми:**
   * Під час створення об'єкта Form1 ініціалізуються компоненти форми, включаючи два елементи PictureBox для відображення прапорів України та Болгарії.
2. **Малювання прапора України (pictureBoxUkraine\_Paint):**
   * У події Paint для елемента pictureBoxUkraine виконується малювання прапора України.
   * Прапор розділений на дві рівні частини:
     + Верхня частина зафарбовується синім кольором (Brushes.Blue).
     + Нижня частина зафарбовується жовтим кольором (Brushes.Yellow).
3. **Малювання прапора Болгарії (pictureBoxBulgaria\_Paint):**
   * У події Paint для елемента pictureBoxBulgaria виконується малювання прапора Болгарії.
   * Прапор складається з трьох рівних горизонтальних смуг:
     + Верхня смуга зафарбовується білим кольором (Brushes.White).
     + Середня смуга зафарбовується зеленим кольором (Brushes.Green).
     + Нижня смуга зафарбовується червоним кольором (Brushes.Red).
4. **Реакція на події миші (MouseEnter і MouseLeave):**
   * Подія MouseEnter для pictureBoxUkraine:
     + При наведенні курсора миші на pictureBoxUkraine, прапор Болгарії (pictureBoxBulgaria) стає видимим.
   * Подія MouseLeave для pictureBoxUkraine:
     + При виході курсора миші з області pictureBoxUkraine, прапор Болгарії стає невидимим.
5. **Візуалізація:**
   * У стартовому стані відображається лише прапор України.
   * При наведенні курсора на прапор України з'являється прапор Болгарії.

**Завдання 3**

Модифікувати минуле завдання, замінивши подію наведения мишачого курсору на натискання лівої кнопки миші на компоненті, а подію видалення курсору на відпускання кнопки миші. Локацію ліворуч чи праворуч треба замінити на зверху чи знизу (в залежності від локації похідного пpaпopa) від пpaпopa України.

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task03

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void pictureBoxUkraine\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малюємо прапор України

int width = pictureBoxUkraine.ClientSize.Width;

int height = pictureBoxUkraine.ClientSize.Height;

g.FillRectangle(Brushes.Blue, 0, 0, width, height / 2);

g.FillRectangle(Brushes.Yellow, 0, height / 2, width, height / 2);

}

private void pictureBoxBulgaria\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малюємо прапор Болгарії

int width = pictureBoxBulgaria.ClientSize.Width;

int height = pictureBoxBulgaria.ClientSize.Height;

g.FillRectangle(Brushes.White, 0, 0, width, height / 3);

g.FillRectangle(Brushes.Green, 0, height / 3, width, height / 3);

g.FillRectangle(Brushes.Red, 0, 2 \* height / 3, width, height / 3);

}

private void pictureBoxUkraine\_MouseDown(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

// Показати PictureBox із прапором Болгарії при натисканні

pictureBoxBulgaria.Visible = true;

}

}

private void pictureBoxUkraine\_MouseUp(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

// Сховати PictureBox із прапором Болгарії при відпусканні

pictureBoxBulgaria.Visible = false;

}

}

}

}

***Результат роботи програми:***



Програма реалізує графічне відображення прапорів України та Болгарії з використанням елементів PictureBox у Windows Forms, а також взаємодію з користувачем через події миші. Програма складається з класу Form1, який є основним класом форми. Ключові функції програми описані нижче:

1. **Ініціалізація форми:**
   * Під час створення об'єкта Form1 ініціалізуються компоненти форми, включаючи два елементи PictureBox для відображення прапорів України та Болгарії.
2. **Малювання прапора України (pictureBoxUkraine\_Paint):**
   * У події Paint для елемента pictureBoxUkraine виконується малювання прапора України.
   * Прапор розділений на дві рівні горизонтальні частини:
     + Верхня частина зафарбовується синім кольором (Brushes.Blue).
     + Нижня частина зафарбовується жовтим кольором (Brushes.Yellow).
3. **Малювання прапора Болгарії (pictureBoxBulgaria\_Paint):**
   * У події Paint для елемента pictureBoxBulgaria виконується малювання прапора Болгарії.
   * Прапор складається з трьох рівних горизонтальних смуг:
     + Верхня смуга зафарбовується білим кольором (Brushes.White).
     + Середня смуга зафарбовується зеленим кольором (Brushes.Green).
     + Нижня смуга зафарбовується червоним кольором (Brushes.Red).
4. **Реакція на події миші (MouseDown і MouseUp):**
   * Подія MouseDown для pictureBoxUkraine:
     + При натисканні лівої кнопки миші на pictureBoxUkraine, прапор Болгарії (pictureBoxBulgaria) стає видимим.
   * Подія MouseUp для pictureBoxUkraine:
     + При відпусканні лівої кнопки миші з області pictureBoxUkraine, прапор Болгарії стає невидимим.
5. **Візуалізація:**
   * У стартовому стані відображається лише прапор України.
   * При натисканні лівої кнопки миші на прапор України з'являється прапор Болгарії, а при відпусканні кнопки — прапор Болгарії зникає.

**Завдання 4**

Створити застосунок, який комбінував би роботу трьох попередніх завдань, змінюючи локацію пpaпopy іншої країни відносно пpaпopy України згідно з локаціями цих завдань під час спрацьовування таймеру, який треба використати в процесі розробки застосунку. Тобто застосунок мас перемикати вміст **PictureBox** між станами: пpaпop України в однині, праворуч чи ліворуч пpaпopy України з’являеться пpaпop іншої країни, пpaпop іншої країни переміщусться до гори чи до низу. Інтервал спрацьовування таймеру можна вибрати за бажанням.

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task04

{

public partial class Form1 : Form

{

private int state = 0; // Поточний стан

private System.Windows.Forms.Timer timer; // Таймер для перемикання станів

public Form1()

{

InitializeComponent();

// Ініціалізація таймера

timer = new System.Windows.Forms.Timer();

timer.Interval = 2000; // Інтервал 2 секунди

timer.Tick += Timer\_Tick;

timer.Start();

}

private void Timer\_Tick(object sender, EventArgs e)

{

state = (state + 1) % 3; // Зміна стану (0, 1, 2)

UpdateFlagLocations();

}

private void UpdateFlagLocations()

{

switch (state)

{

case 0:

// Стан 0: Прапор України в однині

pictureBoxBulgaria.Visible = false;

break;

case 1:

// Стан 1: Прапор Болгарії праворуч від України

pictureBoxBulgaria.Location = new Point(pictureBoxUkraine.Right + 10, pictureBoxUkraine.Top);

pictureBoxBulgaria.Visible = true;

break;

case 2:

// Стан 2: Прапор Болгарії зверху над Україною

pictureBoxBulgaria.Location = new Point(pictureBoxUkraine.Left, pictureBoxUkraine.Top - pictureBoxBulgaria.Height - 10);

pictureBoxBulgaria.Visible = true;

break;

}

}

private void pictureBoxUkraine\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малюємо прапор України

int width = pictureBoxUkraine.ClientSize.Width;

int height = pictureBoxUkraine.ClientSize.Height;

g.FillRectangle(Brushes.Blue, 0, 0, width, height / 2);

g.FillRectangle(Brushes.Yellow, 0, height / 2, width, height / 2);

}

private void pictureBoxBulgaria\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Малюємо прапор Болгарії

int width = pictureBoxBulgaria.ClientSize.Width;

int height = pictureBoxBulgaria.ClientSize.Height;

g.FillRectangle(Brushes.White, 0, 0, width, height / 3);

g.FillRectangle(Brushes.Green, 0, height / 3, width, height / 3);

g.FillRectangle(Brushes.Red, 0, 2 \* height / 3, width, height / 3);

}

}

}

***Результат роботи програми:***



******

Програма реалізує графічне відображення прапорів України та Болгарії у Windows Forms, а також динамічну зміну їх розташування за допомогою таймера. Основний клас Form1 містить логіку для періодичної зміни станів, що впливають на вигляд і позицію прапорів. Нижче наведено детальний опис функціоналу програми.

**Основні компоненти програми:**

1. **Таймер для зміни станів (System.Windows.Forms.Timer):**
   * Використовується для автоматичного перемикання між трьома станами через кожні 2 секунди.
   * У події Timer\_Tick змінюється значення змінної state, яка визначає поточний стан розташування прапорів.
2. **Зміна розташування прапорів (UpdateFlagLocations):**
   * Відповідно до значення state, прапори відображаються або приховуються, а їх позиція змінюється:
     + **Стан 0:** Відображається лише прапор України. Прапор Болгарії приховано.
     + **Стан 1:** Прапор Болгарії відображається праворуч від прапора України, із проміжком у 10 пікселів.
     + **Стан 2:** Прапор Болгарії розташовується над прапором України, із відступом у 10 пікселів.
3. **Малювання прапора України (pictureBoxUkraine\_Paint):**
   * У події Paint для pictureBoxUkraine створюється прапор України:
     + Верхня половина заповнюється синім кольором (Brushes.Blue).
     + Нижня половина заповнюється жовтим кольором (Brushes.Yellow).
4. **Малювання прапора Болгарії (pictureBoxBulgaria\_Paint):**
   * У події Paint для pictureBoxBulgaria створюється прапор Болгарії:
     + Верхня третина заповнюється білим кольором (Brushes.White).
     + Середня третина заповнюється зеленим кольором (Brushes.Green).
     + Нижня третина заповнюється червоним кольором (Brushes.Red).

**Завдання 5**

Розробити програму, форма якої має фонове зображення електричного умовного позначення елемента за наступним зразком

Изображение выглядит как снимок экрана, текст, Шрифт, линия

Автоматически созданное описание

та власним варіантом

|  |  |  |  |
| --- | --- | --- | --- |
| № варіанту | Елемент | Опис | Колір |
| 1 | 2 | 3 | 4 |
| 9. |  | телефон | Блакитний |

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task05

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Фоновий прямокутник

Brush grayBrush = new SolidBrush(Color.White);

int rectWidth = 300;

int rectHeight = 200;

int rectX = (this.ClientSize.Width - rectWidth) / 2;

int rectY = (this.ClientSize.Height - rectHeight) / 2;

g.FillRectangle(grayBrush, rectX, rectY, rectWidth, rectHeight);

// Параметри малювання телефону

Pen bluePen = new Pen(Color.Blue, 5);

int squareSize = 100;

int wireLength = 80; // Довжина дротів ліворуч

int lineExtension = 40; // Подовження вертикальної лінії праворуч

// Центр світло-сірого прямокутника

int centerX = rectX + rectWidth / 2;

int centerY = rectY + rectHeight / 2;

// Малювання прямокутника (корпус телефону)

Rectangle phoneRect = new Rectangle(centerX - squareSize / 2, centerY - squareSize / 2, squareSize, squareSize);

g.DrawRectangle(bluePen, phoneRect);

// Малювання двох подовжених ліній (дроти ліворуч)

g.DrawLine(bluePen, centerX - squareSize / 2 - wireLength, centerY - squareSize / 4, centerX - squareSize / 2, centerY - squareSize / 4);

g.DrawLine(bluePen, centerX - squareSize / 2 - wireLength, centerY + squareSize / 4, centerX - squareSize / 2, centerY + squareSize / 4);

// Малювання подовженої вертикальної лінії справа

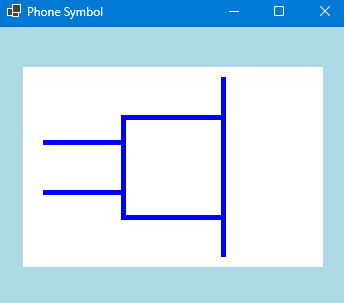
g.DrawLine(bluePen, centerX + squareSize / 2, centerY - squareSize / 2 - lineExtension, centerX + squareSize / 2, centerY + squareSize / 2 + lineExtension);

}

}

}

***Результат роботи програми:***



Програма реалізує графічне відображення схематичного зображення телефону у Windows Forms за допомогою обробника події Paint. Основний клас Form1 відповідає за малювання елементів інтерфейсу та зображення об'єктів, використовуючи методи класу Graphics. Нижче наведено детальний опис функціоналу програми.

**Основні компоненти програми:**

1. **Фоновий прямокутник:**
   * Прямокутник білого кольору (Brush Color.White) відображається у центрі вікна форми.
   * Його розміри становлять 300 пікселів по ширині та 200 пікселів по висоті. Він створює базову область для малювання інших елементів.
2. **Корпус телефону:**
   * Малюється у вигляді синього прямокутника (Pen Color.Blue, товщина 5 пікселів), розташованого в центрі фонового прямокутника.
   * Розмір прямокутника: 100x100 пікселів.
3. **Дроти телефону (ліворуч):**
   * Малюються дві горизонтальні лінії, які з’єднуються з лівою стороною корпусу телефону.
   * Відстань між кінцем дротів і корпусом становить 80 пікселів, а самі лінії розташовані на рівні верхньої та нижньої частини корпусу телефону.
4. **Подовжена вертикальна лінія (праворуч):**
   * Малюється вертикальна лінія, яка проходить уздовж правого боку корпусу телефону.
   * Лінія виходить за межі корпусу на 40 пікселів зверху і знизу, створюючи схематичне зображення антени чи іншого елемента.

**Завдання 6**

Розробити програму, форма якої містить графік коливання курсу національної валюти за наступним зразком

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

Виміри курсу повинні мати випадковий характер. Кількість валютних вимірів (обумовлює крок графіку) за бажанням, але не менш ніж l0. Стиль та колір лінії графіку визначаеться власним варіантом

|  |  |  |
| --- | --- | --- |
| № варіанту | Стиль лінії графіку | Колір лінії графіку |
| 1 | 2 | 3 |
| 9. | DashDot | Блакитний |

Додати до форми кнопку із написом оновити, яка перебудовує графік за випадковими даними.

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task06

{

public partial class Form1 : Form

{

private Random random = new Random();

private float[] currencyValues;

public Form1()

{

InitializeComponent();

GenerateRandomData();

}

private void GenerateRandomData()

{

int pointsCount = 10; // Кількість вимірів

currencyValues = new float[pointsCount];

for (int i = 0; i < pointsCount; i++)

{

// Генеруємо випадкові значення курсу (наприклад, у діапазоні від 30 до 32)

currencyValues[i] = (float)(30 + random.NextDouble() \* 2);

}

// Перемальовуємо графік

Invalidate();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

Pen pen = new Pen(Color.CadetBlue, 2) { DashStyle = System.Drawing.Drawing2D.DashStyle.DashDot };

int margin = 50; // Відступи з боків

int width = ClientSize.Width - 2 \* margin;

int height = ClientSize.Height - 2 \* margin;

// Малювання осей

g.DrawLine(Pens.Black, margin, margin, margin, height + margin); // Y-axis

g.DrawLine(Pens.Black, margin, height + margin, width + margin, height + margin); // X-axis

if (currencyValues == null || currencyValues.Length < 2)

return;

// Масштабування

float maxY = 32; // Верхнє значення діапазону

float minY = 30; // Нижнє значення діапазону

float scaleX = (float)width / (currencyValues.Length - 1);

float scaleY = (float)height / (maxY - minY);

// Малювання графіка

PointF[] points = new PointF[currencyValues.Length];

for (int i = 0; i < currencyValues.Length; i++)

{

float x = margin + i \* scaleX;

float y = height + margin - (currencyValues[i] - minY) \* scaleY;

points[i] = new PointF(x, y);

// Підписуємо значення

string label = currencyValues[i].ToString("0.00");

g.DrawString(label, new Font("Arial", 8), Brushes.Black, x - 10, y - 20);

}

// Малюємо лінію графіка

g.DrawLines(pen, points);

// Малюємо точки графіка

foreach (var point in points)

{

g.FillRectangle(Brushes.Black, point.X - 2, point.Y - 2, 4, 4);

}

}

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

{

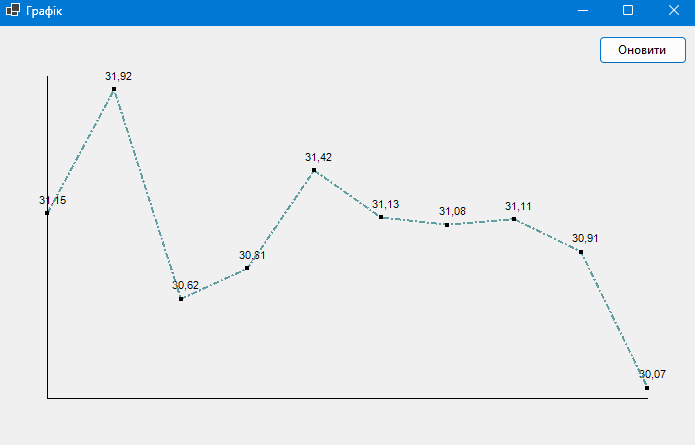
GenerateRandomData(); // Генеруємо нові дані

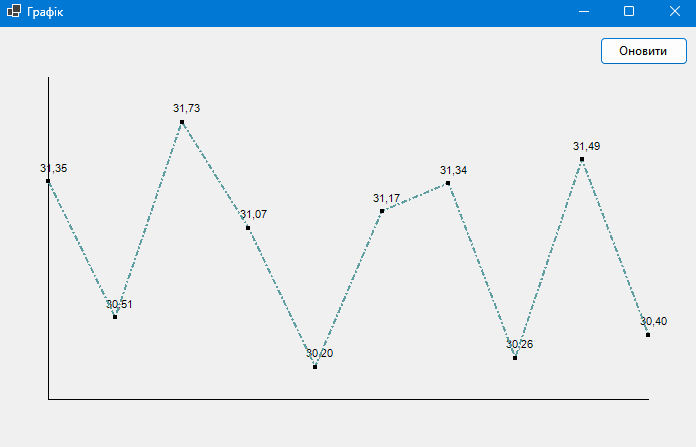
}

}

}

***Результат роботи програми:***





Програма реалізує графічне відображення динамічного лінійного графіка, що демонструє зміну умовного валютного курсу на основі випадкових значень. Графік будується у вікні Windows Forms і автоматично оновлюється при натисканні кнопки. Основний клас Form1 відповідає за генерацію даних, масштабування та відображення графіка.

**Основні компоненти програми:**

1. **Генерація даних:**
   * У методі GenerateRandomData() створюється масив currencyValues, що містить 10 випадкових значень курсу у діапазоні від 30 до 32.
   * Метод викликається при запуску програми та натисканні кнопки оновлення.
2. **Малювання осей:**
   * На формі відображаються осі координат:
     + Вертикальна вісь (Y) — розташована зліва від графіка.
     + Горизонтальна вісь (X) — розташована внизу графіка.
3. **Побудова графіка:**
   * Масштабування виконується з урахуванням:
     + Діапазону значень (від 30 до 32).
     + Розмірів вікна форми та відступів.
   * Точки графіка формуються на основі розрахованих координат і відображаються як чорні квадрати.
4. **Лінії графіка:**
   * Лінії, що з’єднують точки, малюються з використанням пера (Pen) кольору CadetBlue зі стилем лінії DashDot.
5. **Позначення значень:**
   * Кожна точка графіка підписується відповідним значенням курсу (з точністю до двох знаків після коми).
6. **Оновлення графіка:**
   * Натискання кнопки викликає метод GenerateRandomData(), що генерує нові значення для графіка та виконує його повторне відображення за допомогою Invalidate().

**Завдання 7**

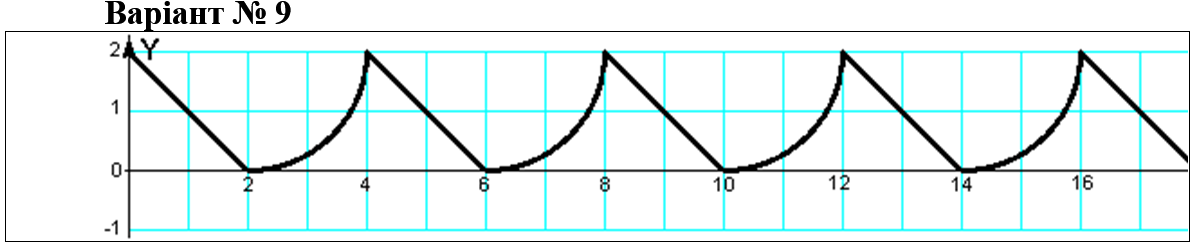
Розробити програму, форма якої містить графік завдання 9 лабораторної роботи № 1 з курсу «Основи мови С#».

**Завдання 9 лабораторної роботи №1 з курсу «Основи мови С#».**

Використовуючи завдання власного варіанта до лабораторної роботи 7 з електронних методичних вказівок з основ програмування та алгоритмічних мов створити проект з реалізації цих завдань засобами мови C#.

***Лабораторна робота №7***

Для функції ***y = f***(***x***), графік якої наведений у виданому індивідуальному завданні, вивести на екран значення ***y*** для ***x =*** 0, 0.25, 0.5 ... 19.75. Необов’язкова для виконання додаткова вимога – отримати на екрані графік функції з використанням засобів текстового режиму.



**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task07

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Параметри області малювання

int margin = 50; // Відступи

int width = ClientSize.Width - 2 \* margin;

int height = ClientSize.Height - 2 \* margin;

// Параметри осей

Pen axisPen = new Pen(Color.Black, 2);

// Малюємо осі

g.DrawLine(axisPen, margin, height + margin, margin, margin); // Y-axis

g.DrawLine(axisPen, margin, height + margin, width + margin, height + margin); // X-axis

// Малювання сітки

Pen gridPen = new Pen(Color.LightBlue, 1);

for (int i = 1; i <= 10; i++)

{

// Вертикальні лінії

int x = margin + i \* (width / 10);

g.DrawLine(gridPen, x, margin, x, height + margin);

// Горизонтальні лінії

int y = margin + i \* (height / 10);

g.DrawLine(gridPen, margin, y, width + margin, y);

}

// Параметри функції

double minX = 0;

double maxX = 20;

double step = 0.25;

// Масштабування

float scaleX = (float)width / (float)(maxX - minX);

float scaleY = (float)height / 4.0f; // Діапазон Y: від -2 до 2

// Координати для побудови графіка

PointF[] points = new PointF[(int)((maxX - minX) / step) + 1];

int index = 0;

for (double x = minX; x <= maxX; x += step)

{

double y = SawtoothWave(x);

// Перетворення координат в область малювання

float graphX = margin + (float)((x - minX) \* scaleX);

float graphY = height + margin - (float)((y + 2) \* scaleY); // Y зміщений на 2

points[index++] = new PointF(graphX, graphY);

}

// Малювання графіка

Pen graphPen = new Pen(Color.Black, 2);

g.DrawLines(graphPen, points);

}

// Обчислення пилкоподібної хвилі

private double SawtoothWave(double x)

{

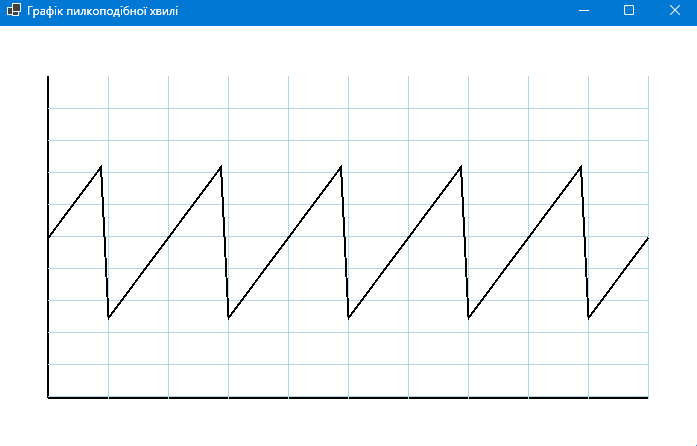
return 2 \* (x / 4.0 - Math.Floor(x / 4.0 + 0.5));

}

}

}

***Результат:***



Програма створює графічне відображення пилкоподібної хвилі у вікні Windows Forms. Графік будується з урахуванням сітки та осей координат, а також автоматичного масштабування. Основний клас Form1 відповідає за обчислення координат і відображення графіка.

**Основні компоненти програми:**

1. **Малювання осей координат:**
   * Вертикальна вісь (Y) малюється чорним кольором зліва від області графіка.
   * Горизонтальна вісь (X) розташована внизу графіка.
2. **Малювання сітки:**
   * Сітка складається з 10 вертикальних і 10 горизонтальних ліній.
   * Для малювання сітки використовується світло-блакитний колір.
3. **Генерація графіка пилкоподібної хвилі:**
   * Значення функції обчислюються у методі SawtoothWave, що реалізує формулу:
   * Значення xxx змінюється у діапазоні від 0 до 20 з кроком 0.25.
4. **Масштабування координат:**
   * Горизонтальна шкала (XXX) масштабується на ширину області графіка.
   * Вертикальна шкала (YYY) масштабується так, щоб відобразити значення від -2 до 2 з урахуванням відступів і розміру вікна.
5. **Малювання графіка:**
   * Побудовані точки об'єднуються у вигляді ламаної лінії чорного кольору товщиною 2 пікселі.

**Завдання 8**

Розробити програму, форма якої містить кругову діаграму, використовуючи наступний зразок

Изображение выглядит как текст, снимок экрана, Операционная система, программное обеспечение

Автоматически созданное описание

згідно свого варіанту

|  |  |
| --- | --- |
| № варіанту | Завдання на побудову діаграми |
| 1 | 2 |
| 9. |  |

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task08

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

// Дані для діаграми

string[] labels = { "Вода - 60%", "Білки - 20%", "Жири - 15%", "Вуглеводи та інші речовини - 5%" };

float[] values = { 3f / 5f, 1f / 5f, 3f / 20f, 1f / 20f };

// Кольори для кожного сегмента

Brush[] brushes = {

Brushes.LightBlue, // Вода

Brushes.LightGreen, // Білки

Brushes.Orange, // Жири

Brushes.Purple // Вуглеводи та інші

};

// Центр і радіус діаграми

int centerX = ClientSize.Width / 3;

int centerY = ClientSize.Height / 2;

int radius = 100;

// Початковий кут

float startAngle = 0;

// Малювання кругової діаграми

for (int i = 0; i < values.Length; i++)

{

// Кут для поточного сегмента

float sweepAngle = values[i] \* 360;

// Малюємо сегмент

g.FillPie(brushes[i], centerX - radius, centerY - radius, 2 \* radius, 2 \* radius, startAngle, sweepAngle);

// Переходимо до наступного сегмента

startAngle += sweepAngle;

}

// Легенда

int legendX = centerX + radius + 50;

int legendY = centerY - radius;

for (int i = 0; i < labels.Length; i++)

{

// Малюємо прямокутник кольору сегмента

g.FillRectangle(brushes[i], legendX, legendY + i \* 30, 20, 20);

// Підпис для сегмента

g.DrawString(labels[i], new Font("Arial", 10), Brushes.Black, legendX + 30, legendY + i \* 30);

}

// Заголовок

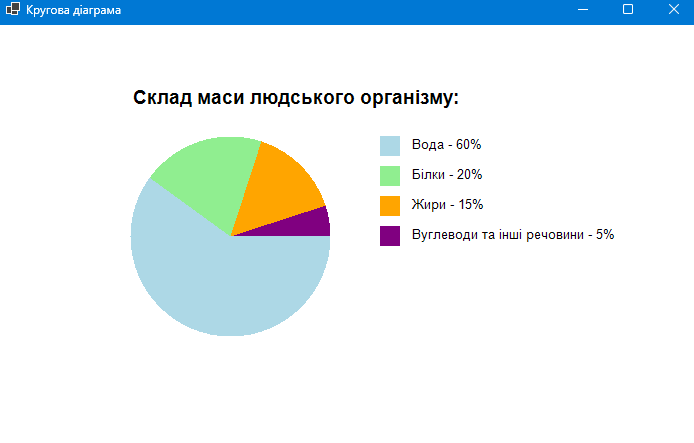
g.DrawString("Склад маси людського організму:", new Font("Arial", 14, FontStyle.Bold), Brushes.Black, centerX - radius, centerY - radius - 50);

}

}

}

***Результат роботи:***



Програма створює кругову діаграму, яка відображає склад маси людського організму у відсотках. Діаграма містить легенду для пояснення кожного сегмента та заголовок, що описує зміст діаграми. Основний клас Form1 відповідає за малювання кругової діаграми, легенди та заголовка.

**Основні компоненти програми:**

1. **Дані для діаграми:**
   * Програма використовує масиви:
     + **labels** — назви сегментів діаграми із зазначенням їхньої частки у відсотках:  
       "Вода - 60%", "Білки - 20%", "Жири - 15%", "Вуглеводи та інші речовини - 5%".
     + **values** — числові значення, які визначають частку кожного сегмента (у частках від 1):  
       3/5, 1/5, 3/20, 1/20.
   * **brushes** — кольори для кожного сегмента:
     + Вода — світло-блакитний.
     + Білки — світло-зелений.
     + Жири — помаранчевий.
     + Вуглеводи та інші — фіолетовий.
2. **Малювання кругової діаграми:**
   * Діаграма малюється з центром у точці, що знаходиться на 1/3 ширини форми та половині її висоти.
   * Радіус діаграми становить 100 пікселів.
   * Кожен сегмент малюється методом FillPie, який задає сектор круга відповідно до пропорційного значення.
3. **Додавання легенди:**
   * Легенда розташована праворуч від діаграми.
   * Для кожного сегмента малюється кольоровий прямокутник і текстова підпис, що пояснює його.
4. **Заголовок:**
   * Над діаграмою розміщується текст заголовка "Склад маси людського організму:" великим шрифтом жирного стилю.

**Завдання 9**

Використовуючи засоби графіки та компонент timer створити динамічний застосунок, який реалізуе pyx двовимірних кульок різного діаметру та кольору у прямокутній області. Під час ружу мас бути реалізована імітація стикання кульок між собою та із сторонами прямокутної області. Форма реалізації застосунку може бути створена, наприклад, за таким зразком

Изображение выглядит как снимок экрана, Операционная система, мультимедиа, программное обеспечение

Автоматически созданное описание

Параметри кульок визначаються власним варіантом

|  |  |  |  |
| --- | --- | --- | --- |
| № варіанту | Кількість кульок | Особливості діаметрів кульок | Кольорові параметри |
| 1 | 2 | 3 | 4 |
| 9. | 5 | Діаметри однакові | Усі кольори різні |

**Текст програми:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace task09

{

public partial class Form1 : Form

{

private class Ball

{

public int X { get; set; }

public int Y { get; set; }

public int Diameter { get; set; }

public Color Color { get; set; }

public int VelocityX { get; set; }

public int VelocityY { get; set; }

}

private readonly Ball[] balls;

private readonly System.Windows.Forms.Timer timer;

private const int BallDiameter = 40;

public Form1()

{

InitializeComponent();

// Увімкнути подвійну буферизацію для усунення миготіння

this.DoubleBuffered = true;

// Ініціалізація кульок

balls = new Ball[5];

Color[] colors = { Color.Red, Color.Blue, Color.Green, Color.Yellow, Color.Purple };

Random random = new Random();

for (int i = 0; i < balls.Length; i++)

{

bool overlapping;

do

{

overlapping = false;

balls[i] = new Ball

{

X = random.Next(50, ClientSize.Width - BallDiameter - 50),

Y = random.Next(50, ClientSize.Height - BallDiameter - 50),

Diameter = BallDiameter,

Color = colors[i],

VelocityX = random.Next(4, 8) \* (random.Next(0, 2) == 0 ? 1 : -1),

VelocityY = random.Next(4, 8) \* (random.Next(0, 2) == 0 ? 1 : -1)

};

// Перевірка на перекриття з попередніми кульками

for (int j = 0; j < i; j++)

{

if (IsColliding(balls[i], balls[j]))

{

overlapping = true;

break;

}

}

}

while (overlapping);

}

// Ініціалізація таймера

timer = new System.Windows.Forms.Timer

{

Interval = 15 // Менший інтервал для збільшення швидкості

};

timer.Tick += Timer\_Tick;

timer.Start();

}

private void Timer\_Tick(object sender, EventArgs e)

{

for (int i = 0; i < balls.Length; i++)

{

Ball ball = balls[i];

// Рух кульки

ball.X += ball.VelocityX;

ball.Y += ball.VelocityY;

// Перевірка зіткнень зі стінками

if (ball.X <= 0)

{

ball.X = 0;

ball.VelocityX = -ball.VelocityX;

}

else if (ball.X + ball.Diameter >= ClientSize.Width)

{

ball.X = ClientSize.Width - ball.Diameter;

ball.VelocityX = -ball.VelocityX;

}

if (ball.Y <= 0)

{

ball.Y = 0;

ball.VelocityY = -ball.VelocityY;

}

else if (ball.Y + ball.Diameter >= ClientSize.Height)

{

ball.Y = ClientSize.Height - ball.Diameter;

ball.VelocityY = -ball.VelocityY;

}

// Перевірка зіткнень з іншими кульками

for (int j = i + 1; j < balls.Length; j++)

{

Ball otherBall = balls[j];

if (IsColliding(ball, otherBall))

{

// Обчислення вектора зіткнення

int dx = otherBall.X - ball.X;

int dy = otherBall.Y - ball.Y;

double distance = Math.Sqrt(dx \* dx + dy \* dy);

if (distance == 0) distance = 1; // Запобігання ділення на нуль

// Нормалізація вектора

double nx = dx / distance;

double ny = dy / distance;

// Переміщення кульок, щоб уникнути перекриття

double overlap = 0.5 \* (distance - ball.Diameter);

ball.X += (int)(overlap \* nx);

ball.Y += (int)(overlap \* ny);

otherBall.X -= (int)(overlap \* nx);

otherBall.Y -= (int)(overlap \* ny);

// Обмін швидкостями для простоти

int tempVx = ball.VelocityX;

int tempVy = ball.VelocityY;

ball.VelocityX = otherBall.VelocityX;

ball.VelocityY = otherBall.VelocityY;

otherBall.VelocityX = tempVx;

otherBall.VelocityY = tempVy;

}

}

}

// Перемальовуємо форму

Invalidate();

}

private bool IsColliding(Ball ball1, Ball ball2)

{

int dx = ball1.X - ball2.X;

int dy = ball1.Y - ball2.Y;

double distance = Math.Sqrt(dx \* dx + dy \* dy);

return distance < ball1.Diameter;

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

foreach (Ball ball in balls)

{

using (Brush brush = new SolidBrush(ball.Color))

{

g.FillEllipse(brush, ball.X, ball.Y, ball.Diameter, ball.Diameter);

}

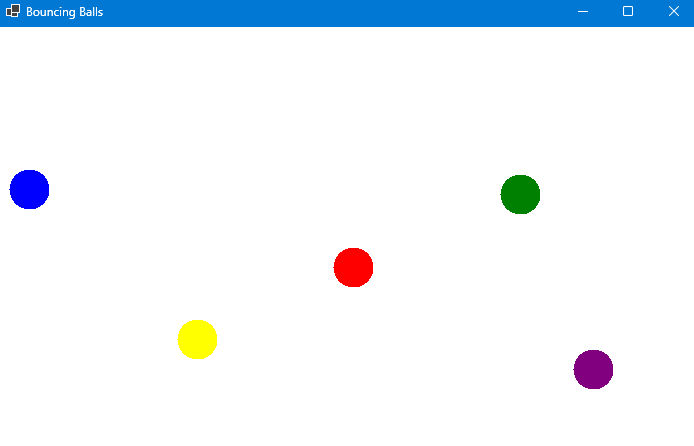
}

}

}

}

***Результат роботи:***



Програма створює анімацію з п'яти кульок, які рухаються в межах вікна форми. Кульки стикаються зі стінками форми та одна з одною, змінюючи напрямок руху при зіткненнях. Програма демонструє основи роботи з графікою, обробки фізичних взаємодій і використання таймерів у середовищі .NET.

**Ключові аспекти реалізації**

1. **Створення класу Ball**
   * Описує кульку, її координати (X, Y), діаметр (Diameter), колір (Color), а також швидкість по осях X та Y (VelocityX, VelocityY).
   * Забезпечує можливість зберігання та зміни властивостей кожної кульки.
2. **Ініціалізація кульок**
   * Створюється масив кульок (balls), кожна з яких отримує випадкові початкові координати, колір, та швидкість.
   * Перевіряється, щоб кульки не перекривалися на початковому етапі.
3. **Обробка руху та зіткнень**
   * Використовується таймер (Timer) для регулярного оновлення координат кульок.
   * Кульки змінюють напрямок при зіткненнях зі стінками форми.
   * Реалізовано алгоритм перевірки зіткнень між кульками з урахуванням перекриття та обміну швидкостями.
4. **Малювання кульок**
   * Для кожної кульки викликається метод FillEllipse, що малює її на формі.
   * Використовується подія Paint для відображення графіки.

**Завдання 10**

Модифікувати застосунок минулого завдання таким чином, щоб створити можливість налаштувань в окремій формі такі параметри кульок:

* кількість;
* кольори;
* діаметри;
* швидкість

**Текст програми:**

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

using System.Drawing;

using System.Windows.Forms;

namespace task09

{

public partial class Form1 : Form

{

private class Ball

{

public int X { get; set; }

public int Y { get; set; }

public int Diameter { get; set; }

public Color Color { get; set; }

public int VelocityX { get; set; }

public int VelocityY { get; set; }

}

private readonly Ball[] balls;

private readonly System.Windows.Forms.Timer timer;

private const int BallDiameter = 40;

public Form1()

{

InitializeComponent();

// Увімкнути подвійну буферизацію для усунення миготіння

this.DoubleBuffered = true;

// Ініціалізація кульок

balls = new Ball[5];

Color[] colors = { Color.Red, Color.Blue, Color.Green, Color.Yellow, Color.Purple };

Random random = new Random();

for (int i = 0; i < balls.Length; i++)

{

bool overlapping;

do

{

overlapping = false;

balls[i] = new Ball

{

X = random.Next(50, ClientSize.Width - BallDiameter - 50),

Y = random.Next(50, ClientSize.Height - BallDiameter - 50),

Diameter = BallDiameter,

Color = colors[i],

VelocityX = random.Next(4, 8) \* (random.Next(0, 2) == 0 ? 1 : -1),

VelocityY = random.Next(4, 8) \* (random.Next(0, 2) == 0 ? 1 : -1)

};

// Перевірка на перекриття з попередніми кульками

for (int j = 0; j < i; j++)

{

if (IsColliding(balls[i], balls[j]))

{

overlapping = true;

break;

}

}

}

while (overlapping);

}

// Ініціалізація таймера

timer = new System.Windows.Forms.Timer

{

Interval = 15 // Менший інтервал для збільшення швидкості

};

timer.Tick += Timer\_Tick;

timer.Start();

}

private void Timer\_Tick(object sender, EventArgs e)

{

for (int i = 0; i < balls.Length; i++)

{

Ball ball = balls[i];

// Рух кульки

ball.X += ball.VelocityX;

ball.Y += ball.VelocityY;

// Перевірка зіткнень зі стінками

if (ball.X <= 0)

{

ball.X = 0;

ball.VelocityX = -ball.VelocityX;

}

else if (ball.X + ball.Diameter >= ClientSize.Width)

{

ball.X = ClientSize.Width - ball.Diameter;

ball.VelocityX = -ball.VelocityX;

}

if (ball.Y <= 0)

{

ball.Y = 0;

ball.VelocityY = -ball.VelocityY;

}

else if (ball.Y + ball.Diameter >= ClientSize.Height)

{

ball.Y = ClientSize.Height - ball.Diameter;

ball.VelocityY = -ball.VelocityY;

}

// Перевірка зіткнень з іншими кульками

for (int j = i + 1; j < balls.Length; j++)

{

Ball otherBall = balls[j];

if (IsColliding(ball, otherBall))

{

// Обчислення вектора зіткнення

int dx = otherBall.X - ball.X;

int dy = otherBall.Y - ball.Y;

double distance = Math.Sqrt(dx \* dx + dy \* dy);

if (distance == 0) distance = 1; // Запобігання ділення на нуль

// Нормалізація вектора

double nx = dx / distance;

double ny = dy / distance;

// Переміщення кульок, щоб уникнути перекриття

double overlap = 0.5 \* (distance - ball.Diameter);

ball.X += (int)(overlap \* nx);

ball.Y += (int)(overlap \* ny);

otherBall.X -= (int)(overlap \* nx);

otherBall.Y -= (int)(overlap \* ny);

// Обмін швидкостями для простоти

int tempVx = ball.VelocityX;

int tempVy = ball.VelocityY;

ball.VelocityX = otherBall.VelocityX;

ball.VelocityY = otherBall.VelocityY;

otherBall.VelocityX = tempVx;

otherBall.VelocityY = tempVy;

}

}

}

// Перемальовуємо форму

Invalidate();

}

private bool IsColliding(Ball ball1, Ball ball2)

{

int dx = ball1.X - ball2.X;

int dy = ball1.Y - ball2.Y;

double distance = Math.Sqrt(dx \* dx + dy \* dy);

return distance < ball1.Diameter;

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

foreach (Ball ball in balls)

{

using (Brush brush = new SolidBrush(ball.Color))

{

g.FillEllipse(brush, ball.X, ball.Y, ball.Diameter, ball.Diameter);

}

}

}

}

}

***ConfigurationForm.cs:***

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Linq;

using System.Windows.Forms;

namespace task10

{

public partial class SettingsForm : Form

{

public Form1.BallSettings UpdatedSettings { get; private set; }

public SettingsForm(Form1.BallSettings currentSettings)

{

InitializeComponent();

// Клонування поточних налаштувань для редагування

UpdatedSettings = new Form1.BallSettings

{

NumberOfBalls = currentSettings.NumberOfBalls,

Colors = new List<Color>(currentSettings.Colors),

Diameter = currentSettings.Diameter,

Speed = currentSettings.Speed

};

// Ініціалізація елементів управління

numericUpDownNumber.Value = UpdatedSettings.NumberOfBalls;

numericUpDownDiameter.Value = UpdatedSettings.Diameter;

numericUpDownSpeed.Value = (decimal)UpdatedSettings.Speed;

// Ініціалізація списку кольорів

listViewColors.Clear();

listViewColors.View = View.Details;

listViewColors.Columns.Add("Кольори", -2, HorizontalAlignment.Left);

listViewColors.FullRowSelect = true;

foreach (Color color in UpdatedSettings.Colors)

{

ListViewItem item = new ListViewItem

{

BackColor = color,

Text = color.IsNamedColor ? color.Name : $"RGB({color.R},{color.G},{color.B})"

};

listViewColors.Items.Add(item);

}

}

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

{

using (ColorDialog colorDialog = new ColorDialog())

{

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

{

UpdatedSettings.Colors.Add(colorDialog.Color);

ListViewItem item = new ListViewItem

{

BackColor = colorDialog.Color,

Text = colorDialog.Color.IsNamedColor ? colorDialog.Color.Name : $"RGB({colorDialog.Color.R},{colorDialog.Color.G},{colorDialog.Color.B})"

};

listViewColors.Items.Add(item);

}

}

}

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

{

if (listViewColors.SelectedItems.Count > 0)

{

int index = listViewColors.SelectedItems[0].Index;

listViewColors.Items.RemoveAt(index);

UpdatedSettings.Colors.RemoveAt(index);

}

}

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

{

// Валідація введених даних

if (UpdatedSettings.Colors.Count == 0)

{

MessageBox.Show("Будь ласка, додайте хоча б один колір.", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (numericUpDownSpeed.Value <= 0)

{

MessageBox.Show("Швидкість повинна бути більше нуля.", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// Оновлення налаштувань

UpdatedSettings.NumberOfBalls = (int)numericUpDownNumber.Value;

UpdatedSettings.Diameter = (int)numericUpDownDiameter.Value;

UpdatedSettings.Speed = (double)numericUpDownSpeed.Value;

this.DialogResult = DialogResult.OK;

this.Close();

}

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

{

this.DialogResult = DialogResult.Cancel;

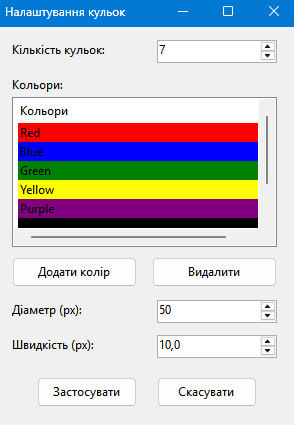
this.Close();

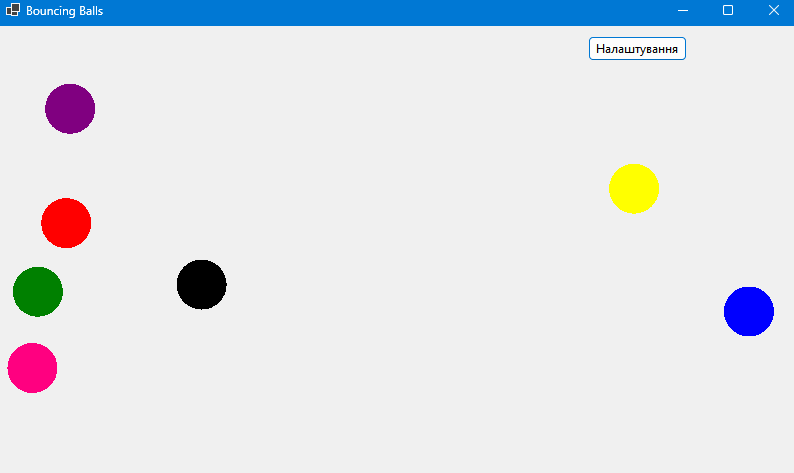
}

}

}

***Результат роботи:***





Програма створює інтерактивне вікно, в якому кілька кульок різних кольорів і розмірів рухаються, відбиваючись одна від одної та від стінок. Користувач може змінювати налаштування кульок, такі як кількість, кольори, діаметр і швидкість їхнього руху, через окреме вікно конфігурації.

**Основна функціональність**

1. **Анімація руху кульок:**
   * Кульки випадковим чином розташовуються у вікні при запуску програми.
   * Рух кульок моделюється з урахуванням їхньої швидкості, напрямку та перевірки на зіткнення зі стінками або іншими кульками.
   * При зіткненні зі стінкою кулька змінює напрямок руху на протилежний.
   * При зіткненні кульки обмінюються швидкостями та коригуються для уникнення накладання.
2. **Налаштування параметрів:**
   * Кількість кульок, кольори, діаметр і швидкість руху можна змінювати через конфігураційне вікно.
   * Усі зміни в налаштуваннях застосовуються до поточної анімації після підтвердження.
3. **Управління та інтерфейс:**
   * Графічний інтерфейс користувача розроблений на основі технології Windows Forms.
   * Використовується таймер для оновлення позицій кульок і їхнього повторного відображення з інтервалом у 15 мс, що забезпечує плавність анімації.
   * Подвійна буферизація використовується для запобігання миготінню під час перемальовування.

**Висновок**

Виконуючи лабораторну роботу №13 були набуті навички створення програми з використанням графічних можливостей Windows Forms. У результаті було створено застосунок, де кульки рухаються, стикаються одна з одною і з краями вікна. Це допомогло зрозуміти, як працювати з графікою, обробляти події, налаштовувати анімацію та взаємодію об'єктів у програмі.

Робота була цікавою і дала корисний досвід, який допоможе в майбутньому створювати подібні програми чи складніші проєкти з графічним інтерфейсом.