МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА

ІМЕНІ ЮРІЯ КОНДРАТЮКА»

Навчально науковий інститут інформаційних технологій і робототехніки

Кафедра комп’ютерних та інформаційних технологій і систем

**Лабораторна робота № 2**

з навчальної дисципліни

"ОБ’ЄКТНО ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ"

Варіант – 1

**Виконала:**

*студента групи 203-ТК*

*Богданов Ілля Віталійович*

**Перевірив:**

*Демиденко Максим Ігорович*

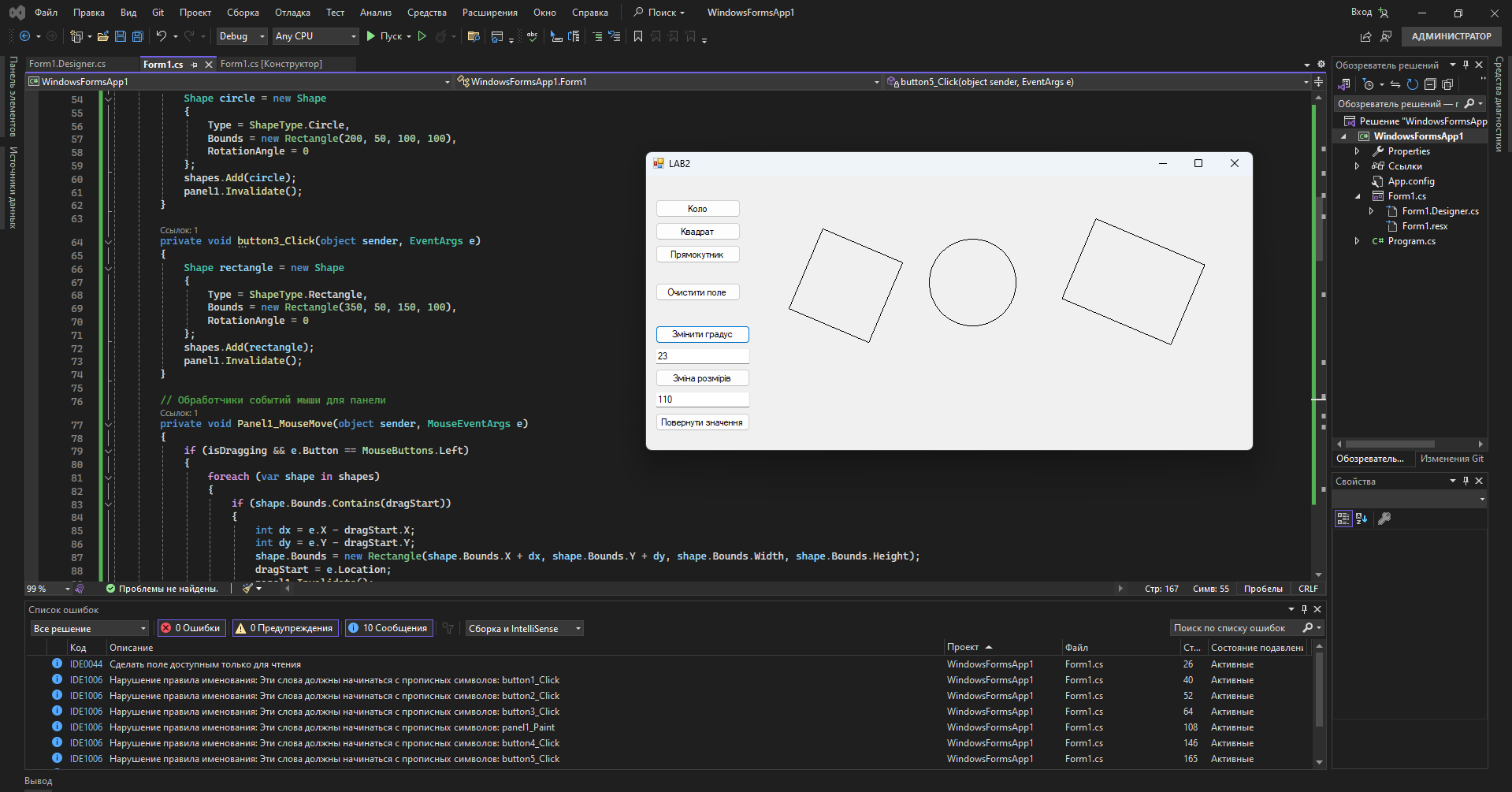
*Полтава 2024*

1. Побудувати систему класів для опису плоских геометричних фігур:

кола, квадрата, прямокутника. Передбачити методи для створення

об’єктів, переміщення на площини, зміни розмірів і повороту на заданий

кут.



using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

// Перечисление для типов фигур

private enum ShapeType

{

Square,

Circle,

Rectangle

}

// Класс для представления фигур

private class Shape

{

public ShapeType Type { get; set; } // Тип фигуры

public Rectangle Bounds { get; set; } // Границы фигуры

public int RotationAngle { get; set; } // Угол поворота фигуры

}

private List<Shape> shapes = new List<Shape>(); // Список для хранения фигур

private Point dragStart; // Точка начала перетаскивания

private bool isDragging = false; // Флаг перетаскивания

public Form1()

{

InitializeComponent();

// Подписываемся на события мыши для панели

panel1.MouseMove += Panel1\_MouseMove;

panel1.MouseDown += Panel1\_MouseDown;

panel1.MouseUp += Panel1\_MouseUp;

}

// Обработчики нажатия на кнопки создания фигур

private void button1\_Click(object sender, EventArgs e)

{

Shape square = new Shape

{

Type = ShapeType.Square,

Bounds = new Rectangle(50, 50, 100, 100),

RotationAngle = 0

};

shapes.Add(square);

panel1.Invalidate();

}

private void button2\_Click(object sender, EventArgs e)

{

Shape circle = new Shape

{

Type = ShapeType.Circle,

Bounds = new Rectangle(200, 50, 100, 100),

RotationAngle = 0

};

shapes.Add(circle);

panel1.Invalidate();

}

private void button3\_Click(object sender, EventArgs e)

{

Shape rectangle = new Shape

{

Type = ShapeType.Rectangle,

Bounds = new Rectangle(350, 50, 150, 100),

RotationAngle = 0

};

shapes.Add(rectangle);

panel1.Invalidate();

}

// Обработчики событий мыши для панели

private void Panel1\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging && e.Button == MouseButtons.Left)

{

foreach (var shape in shapes)

{

if (shape.Bounds.Contains(dragStart))

{

int dx = e.X - dragStart.X;

int dy = e.Y - dragStart.Y;

shape.Bounds = new Rectangle(shape.Bounds.X + dx, shape.Bounds.Y + dy, shape.Bounds.Width, shape.Bounds.Height);

dragStart = e.Location;

panel1.Invalidate();

break;

}

}

}

}

private void Panel1\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

dragStart = e.Location;

}

private void Panel1\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

// Обработчики рисования на панели

private void panel1\_Paint(object sender, PaintEventArgs e)

{

foreach (var shape in shapes)

{

Graphics g = e.Graphics;

switch (shape.Type)

{

case ShapeType.Square:

// Рисуем квадрат

g.TranslateTransform(shape.Bounds.Left + shape.Bounds.Width / 2, shape.Bounds.Top + shape.Bounds.Height / 2);

g.RotateTransform(shape.RotationAngle);

g.TranslateTransform(-(shape.Bounds.Left + shape.Bounds.Width / 2), -(shape.Bounds.Top + shape.Bounds.Height / 2));

g.DrawRectangle(Pens.Black, shape.Bounds);

g.ResetTransform();

break;

case ShapeType.Circle:

// Рисуем круг

g.TranslateTransform(shape.Bounds.Left + shape.Bounds.Width / 2, shape.Bounds.Top + shape.Bounds.Height / 2);

g.RotateTransform(shape.RotationAngle);

g.TranslateTransform(-(shape.Bounds.Left + shape.Bounds.Width / 2), -(shape.Bounds.Top + shape.Bounds.Height / 2));

g.DrawEllipse(Pens.Black, shape.Bounds);

g.ResetTransform();

break;

case ShapeType.Rectangle:

// Рисуем прямоугольник

g.TranslateTransform(shape.Bounds.Left + shape.Bounds.Width / 2, shape.Bounds.Top + shape.Bounds.Height / 2);

g.RotateTransform(shape.RotationAngle);

g.TranslateTransform(-(shape.Bounds.Left + shape.Bounds.Width / 2), -(shape.Bounds.Top + shape.Bounds.Height / 2));

g.DrawRectangle(Pens.Black, shape.Bounds);

g.ResetTransform();

break;

default:

break;

}

}

}

// Обработчик нажатия на кнопку изменения угла поворота

private void button4\_Click(object sender, EventArgs e)

{

// Пытаемся преобразовать текст введенный в текстовое поле в число

if (int.TryParse(textBox1.Text, out int angle))

{

// Устанавливаем угол поворота для всех фигур введенное значение

foreach (var shape in shapes)

{

shape.RotationAngle = angle;

}

panel1.Invalidate(); // Перерисовываем панель

}

else

{

MessageBox.Show("Invalid angle!"); // Выводим сообщение об ошибке, если угол введен некорректно

}

}

// Обработчик нажатия на кнопку сброса угла поворота

private void button5\_Click(object sender, EventArgs e)

{

// Сбрасываем угол поворота для всех фигур

foreach (var shape in shapes)

{

shape.RotationAngle = 0;

}

panel1.Invalidate(); // Перерисовываем панель

}

// Обработчик нажатия на кнопку очистки панели

private void button6\_Click(object sender, EventArgs e)

{

shapes.Clear(); // Очищаем список фигур

panel1.Invalidate(); // Перерисовываем панель

}

private void button7\_Click(object sender, EventArgs e)

{

// Пытаемся преобразовать текст введенный в textBox2 в число

if (int.TryParse(textBox2.Text, out int newSize))

{

// Изменяем размеры всех фигур на новые значения

foreach (var shape in shapes)

{

// Проверяем тип фигуры и устанавливаем новые размеры

switch (shape.Type)

{

case ShapeType.Square:

// Если фигура квадрат, то устанавливаем новый размер

shape.Bounds = new Rectangle(shape.Bounds.Location, new Size(newSize, newSize));

break;

case ShapeType.Circle:

// Если фигура круг, то устанавливаем новый радиус

shape.Bounds = new Rectangle(shape.Bounds.Location, new Size(newSize, newSize));

break;

case ShapeType.Rectangle:

// Если фигура прямоугольник, то изменяем ширину и высоту независимо

shape.Bounds = new Rectangle(shape.Bounds.Location, new Size(shape.Bounds.Width, newSize));

break;

default:

break;

}

}

panel1.Invalidate(); // Перерисовываем панель

}

else

{

MessageBox.Show("Invalid size!"); // Выводим сообщение об ошибке, если размер введен некорректно

}

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

}

}

**Посилання на файли проєкту:**