Лабораторна робота № 23

Тема. Використання інтерфейсів для обробки споріднених об’єктів.

Мета. Формування вмінь і навиків створення інтерфейсів засобами C#. Закріплення вмінь і навиків наслідування звичайних та абстрактних класів, використання об'єктів, підпрограм, елементів керування. Застосування вмінь і навиків програмування алгоритмів лінійної, розгалуженої та циклічної структури.

Контрольні запитання.

1. Що спільного між абстрактними класами та інтерфейсами? Чим вони відрізняються?
2. Як описуються інтерфейси?
3. Чому в інтерфейсах не вказуються специфікатори доступу?
4. Від скількох інтерфейсів може наслідуватися клас? Де при цьому вказується базовий клас, якщо від наявний?
5. Як використовується наслідування різних класів від одного інтерфейсу?

Завдання.

1. **Відкрийте розроблений раніше додаток-форму з описом базового та породжених класів згідно варіанту і методами *draw3D()* для їх зображення. Опишіть у ній клас кола, породжений від абстрактного класу двовимірної фігури та реалізуйте у цьому класі абстрактні методи батьківського класу (знаходження прощі, периметру та інформування про об’єкт).**

Цей клас може бути, наприклад, таким:

```
class Circle : Figure2D
{
    private double r;

    public double R
    {get { return r; }
     set {if (value >= 0) r = value; }}

    public Circle(double r)
    { this.r = r; }

    public sealed override double area()
    { return Math.PI * R * R; }

    public sealed override double perimeter()
    { return 2 * Math.PI * R; }

    public override void Info()
    { MessageBox.Show("Дані кола:\nрадіус: " + R.ToString() + " од.;\nплоща: " + area().ToString() +
       " кв. од.;\nпериметр: " + perimeter().ToString() + " од.",
       "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}
```

Для чого тут використано інструкцію *sealed*?

2. **Опишіть інтерфейс з властивостями кольору, висоти, процедурою для зображення та функціями для визначення об'єму, площі основи та периметра основи. Забезпечте наслідування абстрактного класу тривимірної фігури не лише від базового класу, а й від розробленого інтерфейсу.**

Наприклад, опис інтерфейсу та породженого класу тривимірної фігури може бути таким:

```
public interface IDrawFigure3D
{   byte CR { get; set;}
    byte CG { get; set;}
    byte CB { get; set;}
    double H { get; set; }
    void Draw3D(int startX, int startY, PaintEventArgs e);
    void Info();
    double Volume();
    double AreaBase();
    double PerimeterBase(); }
```

```
abstract class DrawFigure3DBaseRectangle : Rectangle, IDrawFigure3D
{
    private byte cr, cg, cb;
    public byte CR
    {   get { return cr; }
        set { if (value >= 0 && value <= 255) cr = value; }
    }

    public byte CG
    {   get { return cg; }
        set { if (value >= 0 && value <= 255) cg = value; }
    }

    public byte CB
    {   get { return cb; }
        set { if (value >= 0 && value <= 255) cb = value; }
    }

    public abstract void Draw3D(int startX, int startY, PaintEventArgs e);

    private double h;

    public double H
    {   get { return h; }
        set { if (value >= 0) h = value; }
    }

    public DrawFigure3DBaseRectangle(double a, double b, double h) : base(a, b)
    {   H = h;
    }
    public DrawFigure3DBaseRectangle(double a, double b, double h, byte cr, byte cg, byte cb) : base(a, b)
    {   H = h;
        CR = cr; CG = cg; CB = cb;
    }

    public double AreaBase()
    { return base.Area(); }

    public double PerimeterBase()
    { return base.Perimeter(); }

    public abstract double Volume();
}
```

Навіщо в цьому класі оголошуються абстрактні методи, якщо вони вже є в інтерфейсі?

3. **Опишіть у формі клас кругового конуса, породженого від кола і інтерфейсу тривимірної фігури. Реалізуйте у цьому класі абстрактні методи батьківського інтерфейсу (зображення, знаходження об'єму, прощі та периметру основи).**

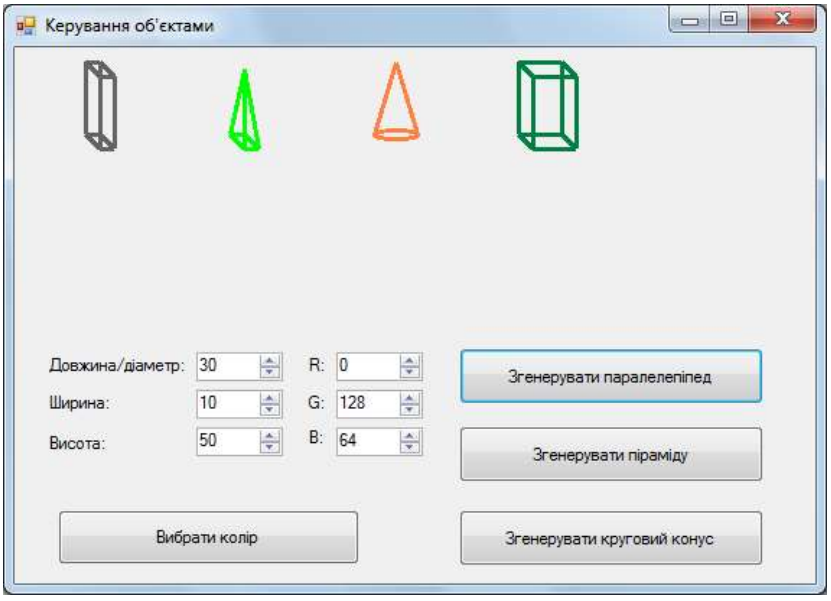
Цей клас може бути, наприклад, таким:

```
class DrawConeCircular : Circle, IDrawFigure3D
{   private byte cr, cg, cb;
    public byte CR
    {   get { return cr; }
        set { if (value >= 0 && value <= 255) cr = value; }
    }

    public byte CG
    {   get { return cg; }
        set { if (value >= 0 && value <= 255) cg = value; }
    }

    public byte CB
    {   get { return cb; }
        set { if (value >= 0 && value <= 255) cb = value; }
    }
    private double h;

    public double H
    { get { return h; }
      set { if (value >= 0) h = value; }}

    public DrawConeCircular(double r, double h) : base(r)
    { this.h = h; }

    public DrawConeCircular(double r, double h, byte cr, byte cg, byte cb) : base(r)
    {   this.h = h;
        CR = cr; CG = cg; CB = cb; }
```

```
    public double AreaBase()
    { return base.Area(); }

    public double PerimeterBase()
    { return Perimeter(); }

    public double Volume()
    { return 1.0/3*R*H; }

    public void Draw3D(int startX, int startY, PaintEventArgs e)
    { Pen pen = new Pen(Color.FromArgb(255, CR, CG, CB), 3);
      int intR = (int)R, intH = (int)H;
      e.Graphics.DrawLine(pen, startX+intR, startY, startX, startY + intH);
      e.Graphics.DrawLine(pen, startX+intR, startY, startX+2*intR, startY + intH);
      e.Graphics.DrawArc(pen, startX, startY+intH-3, 2*intR, 6, 0, 360); }

    public override void Info()
    { MessageBox.Show("Дані конуса:\nрадіус основи: " + R.ToString() + " од., висота: " + H.ToString() +
       " од.;\nплоща основи: " + AreaBase().ToString() + " кв. од.\nоб'єм: " + Volume().ToString() +
       " куб. од.", "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}
```

4. За допомогою списку інтерфейсу забезпечте створення та переміщення з головної кнопкової форми декількох зображень породжених об'єктів для різних класів.

Наприклад, головна кнопкова форма з додатковими елементами керування, нумераторами і кнопками для створення геометричних фігур може бути така:



А її код буде, наприклад, таким:

```
public partial class Form1 : Form
{
    List<IDrawFigure3D> ListDrawFigure3D = new List<IDrawFigure3D>();

    public Form1()
    {   InitializeComponent();}

    private void button3_Click(object sender, EventArgs e)
    {   var pd = new DrawParallelepiped((double)A.Value, (double)B.Value, (double)H.Value, (byte)CR.Value,
                                        (byte)CG.Value, (byte)CB.Value);
        pd.Info();
        ListDrawFigure3D.Add(pd);
        Refresh(); }

    private void button2_Click(object sender, EventArgs e)
    {   var pa = new DrawPiramida((double)A.Value, (double)B.Value, (double)H.Value, (byte)CR.Value,
                                  (byte)CG.Value, (byte)CB.Value);
        pa.Info();
        ListDrawFigure3D.Add(pa);
        Refresh(); }

    private void button4_Click(object sender, EventArgs e)
    {   var c = new DrawConeCircular((double)A.Value/2, (double)H.Value, (byte)CR.Value, (byte)CG.Value,
                                     (byte)CB.Value);
        c.Info();
        ListDrawFigure3D.Add(c);
        Refresh();
    }
```

```
    private void button4_Click(object sender, EventArgs e)
    {   var c = new DrawConeCircular((double)R.Value, (double)H.Value, (byte)CR.Value, (byte)CG.Value,
                                     (byte)CB.Value);
        c.Info();
        ListDrawFigure3D.Add(c);
        Refresh();
    }

    private void Form1_Paint(object sender, PaintEventArgs e)
    {   int pozX = 10;
        foreach (IDrawFigure3D Figure3D in ListDrawFigure3D)
        {Figure3D.Draw3D(DateTime.Now.Second + pozX, 10, e);
         pozX += 100;
        } }

    private void button1_Click(object sender, EventArgs e)
    { colorDialog.Color=Color.FromArgb((int)CR.Value, (int)CG.Value, (int)CB.Value);
      if (colorDialog.ShowDialog()==DialogResult.OK)
      {CR.Value = colorDialog.Color.R;
       CG.Value = colorDialog.Color.G;
       CB.Value = colorDialog.Color.B;
      } }

    private void timer_Tick(object sender, EventArgs e)
    { Refresh(); }
}
```