Curs 02

(*plan de curs*)

1. De ce (C, +, .) ?

- polinoame → serii de puteri → funcţii analitice: $e^z = 1 + \frac{z}{1!} + \frac{z^2}{2!} + \frac{z^3}{3!} + \cdots$

-operatori liniari, vectori proprii → matrice, polinom caracteristic

2. De ce C# ?

- operator overloading: C++, ~~Java~~, C#:

```
Complex Exp(Complex z)
{
    Complex s = 0, p = 1;
    for (int i = 1; i < 100; i++)
    {
        s += p;
        p *= z / i;
    }
    return s;
}
```

3) Windows Forms Application:

- windows, controls, dialogs, forms
- Object → Application: `Application.Run(new MyForm());`
- event-driven application: `Application.DoEvents();`
- programare în mod vizual:
  - etapa automată (design, `MyForm.Designer.cs`),
  - etapa manuală (codul handler-lor de evenimente, `MyForm.cs`)

```
public class ComplexForm : Form
{
        public static AboutForm aboutForm = new AboutForm();
        private Graphics dc;
        private Bitmap bmp;
        private SolidBrush sdBrush;
       ...
}
```

4) Utilizare Complex explorer `public class Sablon : ComplexForm    {...}`

```
public void setXminXmaxYminYmax(double xm, double XM, double ym, double YM)

public void setPixel(int i, int j, Color c)

public void setPixel(double x, double y, Color c)

public void setPixel(Complex z, Color c)

public Color getColor(int k)     //Atentie: k trebuie sa fie pozitiv
```

```
public void setLine(int i0, int j0, int i1, int j1, Color c)

public void setLine(double x0, double y0, double x1, double y1, Color c)

public void setLine(Complex z0, Complex z1, Color c)

public bool resetScreen()
{
      picBox.Refresh();            //pentru re-pictarea ferestrei
      Application.DoEvents();      //pentru procesarea mesajelor
      return doWork;               //pentru semnalizarea opririi desenarii
}

public virtual void makeImage()     //trebuie suprascrisa
{
      setText("Metoda makeImage()","trebuie  suprascris\u0103!");
}
```

5) Spirala logaritmică $z(t)=e^{\lambda t}=e^{(a+ib)t}=e^{at}(\cos bt+i\sin bt)$

Fişierul `MyToolbox.cs`

```
using System;
using System.Drawing;
using System.Collections.Generic;


namespace ComplexExplorer
{
    public class SpiralaLogaritmica : ComplexForm
    {
        Complex i = new Complex(0, 1);
        Complex Exp(Complex z)
        {
            Complex s = 0, p = 1;
            for (int i = 1; i < 100; i++)
            {
                s += p;
                p *= z / i;
            }
            return s;
        }
        public override void makeImage()
        {
            double r = 1.1;
            setXminXmaxYminYmax(-r, r, -r, r);
            ScreenColor = Color.Yellow;
            PenColor = Color.Navy;
            initScreen();
            setAxis();
```

```
            const double a = -0.1;
            const double b = 1.0;
            Complex lambda = a + i * b;

            for (double t = 0; t < 1000; t += 0.01)
            {

                double x = Math.Exp(a * t) * Math.Cos(b * t);
                double y = Math.Exp(a * t) * Math.Sin(b * t);
                setPixel(x, y, Color.Red);

                Complex z = Complex.setRoTheta(Math.Exp(a * t), b * t);
                setPixel(z, Color.Blue);

                setPixel(Exp(lambda * t), Color.Green);
                if (!resetScreen()) return;
            }
        }
    }
}
```

Fişierul `Program.cs`

```
using System;
using System.Windows.Forms;


namespace ComplexExplorer
{
    static class Program
    {
        [STAThread]
        public static void Main()
        {
            Application.Run(new SpiralaLogaritmica());
        }
    }
}
```