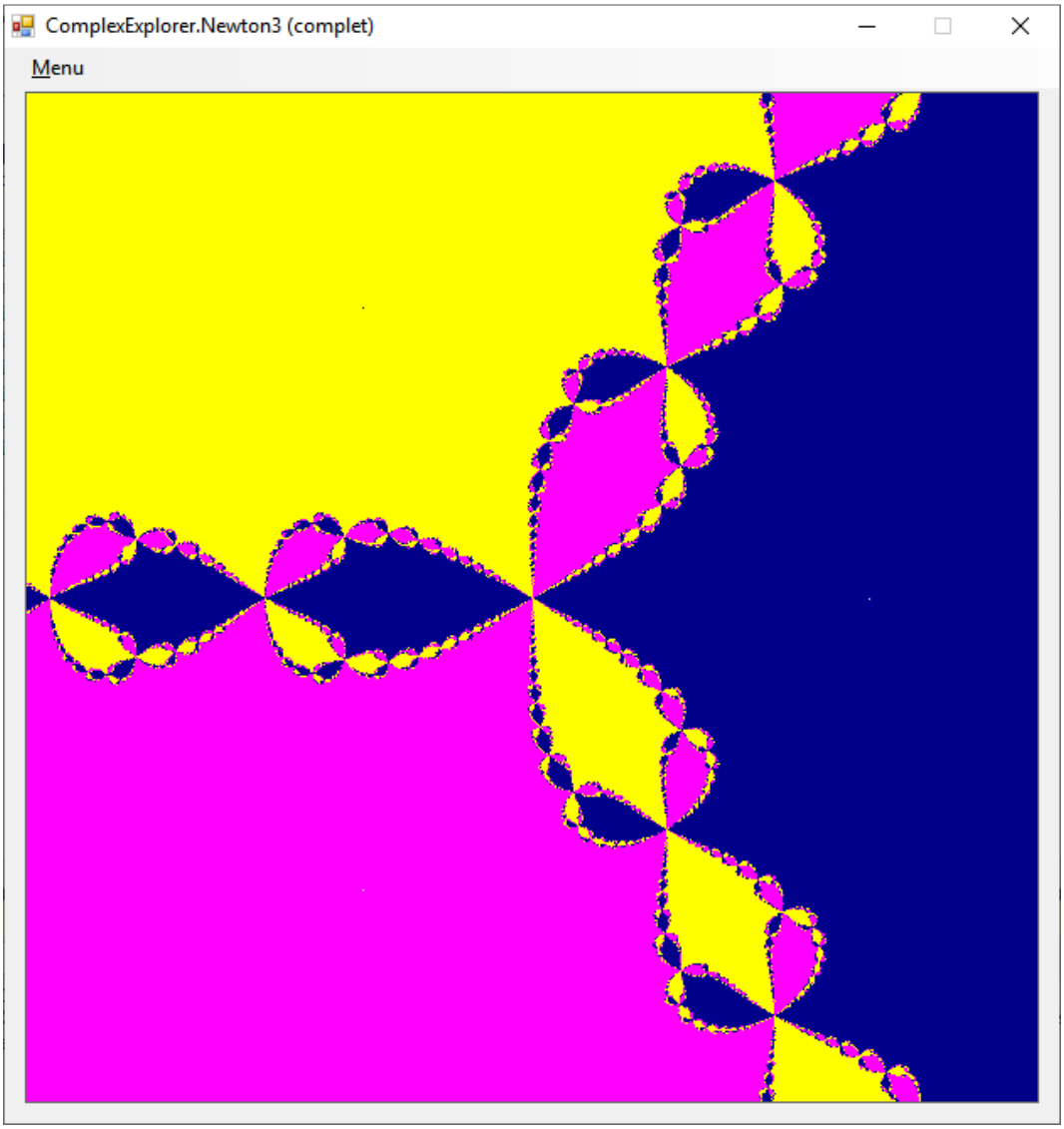
Curs 12

(*plan de curs*)

1. Teorema lui Banach, metoda lui Newton, problema lui Cayley, teorema lui Julia.



2. Desene recurente.

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.Threading;
using System.Collections.Generic;

namespace ComplexExplorer
{
    public class ComplexFormSierpinski : ComplexForm
    {
        public static Complex i = Complex.setReIm(0, 1);
        public struct Patrat
        {   //patratul de centru q si de latura lat
            public Complex q;
            public double lat;
            public Patrat(Complex qq, double llat)
            {
                q = qq;
                lat = llat;
            }
        }
        public void deseneazaPatrat(Patrat P, Color col)
        {
            double r = P.lat / 2;
            int iimin = getI(P.q.Re - r), iimax = getI(P.q.Re + r);
            int jjmin = getJ(P.q.Im - r), jjmax = getI(P.q.Im + r);
            for (int ii = iimin; ii <= iimax; ii++)
                for (int jj = jjmin; jj <= jjmax; jj++)
                {
                    setPixel(ii, jj, col);
                }
            setLine(iimin, jjmin, iimax, jjmin, PenColor);
            setLine(iimin, jjmax, iimax, jjmax, PenColor);
            setLine(iimin, jjmin, iimin, jjmax, PenColor);
            setLine(iimax, jjmin, iimax, jjmax, PenColor);
        }
        public void traseaza(List<Patrat> li, Color col)
        {
            foreach (Patrat p in li) deseneazaPatrat(p, col);
        }

    }

    /*********************************************************************/
    public class SierpinskiMotiveIterate : ComplexFormSierpinski
    {
        void transforma(ref List<Patrat> li)
        {
            List<Patrat> rez = new List<Patrat>();
            foreach (Patrat P in li)
            {
                double x = P.q.Re, y = P.q.Im;
                double lat = P.lat / 3;
                rez.Add(new Patrat(new Complex(x - lat, y - lat), lat));
                rez.Add(new Patrat(new Complex(x - lat, y), lat));
                rez.Add(new Patrat(new Complex(x - lat, y + lat), lat));
                rez.Add(new Patrat(new Complex(x, y - lat), lat));
```

```
            rez.Add(new Patrat(new Complex(x, y + lat), lat));
            rez.Add(new Patrat(new Complex(x + lat, y - lat), lat));
            rez.Add(new Patrat(new Complex(x + lat, y), lat));
            rez.Add(new Patrat(new Complex(x + lat, y + lat), lat));
        }
        li = rez;
    }
    public override void makeImage()
    {
        setXminXmaxYminYmax(-0.1, 1.1, -0.1, 1.1);
        ScreenColor = Color.White;
        List<Patrat> fig = new List<Patrat>();
        fig.Add(new Patrat((1 + i) / 2, 1.0));
        traseaza(fig, getColor(650));
        for (int k = 1; k < 5; k++)
        {
            //initScreen();
            transforma(ref fig);
            traseaza(fig, getColor(650 + 50 * k));
            if (!resetScreen()) return;
        }
    }
}
/*****************************************/

public class SierpinskiRecursiv : ComplexFormSierpinski
{
    void deseneazaRec(Patrat P, int niv)
    {
        if (niv <= 0) return;
        deseneazaPatrat(P, getColor(900 - 50 * niv));
        if (!resetScreen()) return;

        niv--;
        double x = P.q.Re, y = P.q.Im;
        double lat = P.lat / 3;
        deseneazaRec(new Patrat(new Complex(x - lat, y - lat), lat), niv);
        deseneazaRec(new Patrat(new Complex(x - lat, y), lat), niv);
        deseneazaRec(new Patrat(new Complex(x - lat, y + lat), lat), niv);
        deseneazaRec(new Patrat(new Complex(x, y - lat), lat), niv);

        deseneazaRec(new Patrat(new Complex(x, y + lat), lat), niv);
        deseneazaRec(new Patrat(new Complex(x + lat, y - lat), lat), niv);
        deseneazaRec(new Patrat(new Complex(x + lat, y), lat), niv);
        deseneazaRec(new Patrat(new Complex(x + lat, y + lat), lat), niv);
    }

    public override void makeImage()
    {
        setXminXmaxYminYmax(-0.1, 1.1, -0.1, 1.1);
        ScreenColor = Color.White;
        Patrat P = new Patrat((1 + i) / 2, 1.0);
        deseneazaRec(P, 5);
        resetScreen();


    }
}
```

```
    /*****************************************/

    public class SierpinskiTransformariIterate : ComplexFormSierpinski
    {

        Complex w1 = (1 + 1 * i) / 6;
        Complex w2 = (1 + 3 * i) / 6;
        Complex w3 = (1 + 5 * i) / 6;
        Complex w4 = (3 + 1 * i) / 6;
        Complex w5 = (3 + 3 * i) / 6;
        Complex w6 = (3 + 5 * i) / 6;
        Complex w7 = (5 + 1 * i) / 6;
        Complex w8 = (5 + 3 * i) / 6;
        Complex w9 = (5 + 5 * i) / 6;

        Patrat T1(Patrat P) { return new Patrat(w1 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T2(Patrat P) { return new Patrat(w2 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T3(Patrat P) { return new Patrat(w3 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T4(Patrat P) { return new Patrat(w4 + (P.q - w5) / 3.0, P.lat / 3); }
        //lipseste T5
        Patrat T6(Patrat P) { return new Patrat(w6 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T7(Patrat P) { return new Patrat(w7 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T8(Patrat P) { return new Patrat(w8 + (P.q - w5) / 3.0, P.lat / 3); }
        Patrat T9(Patrat P) { return new Patrat(w9 + (P.q - w5) / 3.0, P.lat / 3); }

        void transforma(ref List<Patrat> li)
        {
            List<Patrat> rez = new List<Patrat>();
            foreach (Patrat P in li) rez.Add(T1(P));
            foreach (Patrat P in li) rez.Add(T2(P));
            foreach (Patrat P in li) rez.Add(T3(P));
            foreach (Patrat P in li) rez.Add(T4(P));
            //lipseste T5
            foreach (Patrat P in li) rez.Add(T6(P));
            foreach (Patrat P in li) rez.Add(T7(P));
            foreach (Patrat P in li) rez.Add(T8(P));
            foreach (Patrat P in li) rez.Add(T9(P));
            li = rez;
        }
        public override void makeImage()
        {
            setXminXmaxYminYmax(-0.1, 1.1, -0.1, 1.1);
            ScreenColor = Color.White;
            List<Patrat> fig = new List<Patrat>();
            fig.Add(new Patrat(w5, 1.0));
            traseaza(fig, getColor(650));

            for (int k = 1; k < 5; k++)
            {
                transforma(ref fig);
                traseaza(fig, getColor(650 + 50 * k));
                if (!resetScreen()) return;
            }
        }
    }
    /*************************************/
}
```