Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Технології та засоби розробки комп'ютерної графіки та мультимедіа

Лабораторна робота №3

Виконав:
студент групи
ІО-25
Льоскін І. В.
Перевірила:
Хмелюк М. С.

Київ – 2024

Тема: Візуалізація фракталів

Завдання: Побудувати сніжинку Коха

Хід роботи:

Аналогічно попереднім лабораторним роботам створюємо новий проект. Також створюємо нову іконку, встановлюємо її, додаємо задній фон формі, змінюємо ім’я форми та її текст. Після чого приступаємо до написання коду. Для цього перейдемо в файл SnowflakeForm.cs:

```
 public partial class SnowFlakeForm : Form
 {
     public SnowFlakeForm()
     {
         InitializeComponent();
         this.ClientSize = new Size(700, 700);
         this.Paint += new PaintEventHandler(this.OnPaint);

     }

     private const int Depth = 4; // depth of recursion
     private const float P = 1 / 3f; // k for determination of intermediate
points on the side of the triangle



     private void OnPaint(object sender, PaintEventArgs e)
     {
         Graphics g = e.Graphics;

         // base triangle which have equal sides length
         PointF p1 = new PointF(300, 50);   // top point
         PointF p2 = new PointF(50, 500);   // bottom left point
         PointF p3 = new PointF(550, 500);  // bottom right point

         // draw Koch's snowflake
         DrawKochSnowflake(g, p1, p2, Depth);
         DrawKochSnowflake(g, p2, p3, Depth);
         DrawKochSnowflake(g, p3, p1, Depth);
     }


     // function for drawing Loch's snowflake at one side
     private void DrawKochSnowflake(Graphics g, PointF p1, PointF p2, int
depth)
     {
         if (depth == 0)
         {
             g.DrawLine(Pens.Black, p1, p2);
         }
         else
         {
             // calculate three intermadiate points
```

```
            PointF a = new PointF(p1.X + (p2.X - p1.X) * P, p1.Y + (p2.Y -
p1.Y) * P);
            PointF b = new PointF(p1.X + (p2.X - p1.X) * 2 * P, p1.Y + (p2.Y
- p1.Y) * 2 * P);

            // calculate coordinates of new triangle
            float angle = (float)(Math.PI / 3); // 60 degrees
PointF c = new PointF(
                (float)(a.X + (b.X - a.X) * Math.Cos(angle) - (b.Y - a.Y) *
Math.Sin(angle)),
                (float)(a.Y + (b.X - a.X) * Math.Sin(angle) + (b.Y - a.Y) *
Math.Cos(angle))
            );

            // recursively draw a Koch's snowflake for each part
            DrawKochSnowflake(g, p1, a, depth - 1);
            DrawKochSnowflake(g, a, c, depth - 1);
            DrawKochSnowflake(g, c, b, depth - 1);
            DrawKochSnowflake(g, b, p2, depth - 1);
        }
    }



    private void Form1_Load(object sender, EventArgs e)
    {

    }
}
```

Пояснення коду:

Для початку ініціалізуємо нашу форму, вкажемо її розмір на підписуємось методом Paint на подію OnPaint

```
public SnowFlakeForm()
{
    InitializeComponent();
    this.ClientSize = new Size(700, 700);
    this.Paint += new PaintEventHandler(this.OnPaint);

}
```

створюємо константні зміння такі як глибина рекурсії та коефіцієнт для визначення проміжкових точок сторони трикутника

```
private const int Depth = 4; // depth of recursion
private const float P = 1 / 3f; // k for determination of intermediate points on the side of the triangle
```

Метод OnPaint в якому створено об’єкт для малювання на формі, три вершини рівностороннього трикутника, Викликається метод DrawKochSnowflake для кожної сторони трикутника з вказаною глибиною рекурсії

```
private void OnPaint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    // base triangle which have equal sides length
    PointF p1 = new PointF(300, 50);   // top point
    PointF p2 = new PointF(50, 500);   // bottom left point
    PointF p3 = new PointF(550, 500);  // bottom right point

    // draw Koch's snowflake
    DrawKochSnowflake(g, p1, p2, Depth);
    DrawKochSnowflake(g, p2, p3, Depth);
    DrawKochSnowflake(g, p3, p1, Depth);
}
```

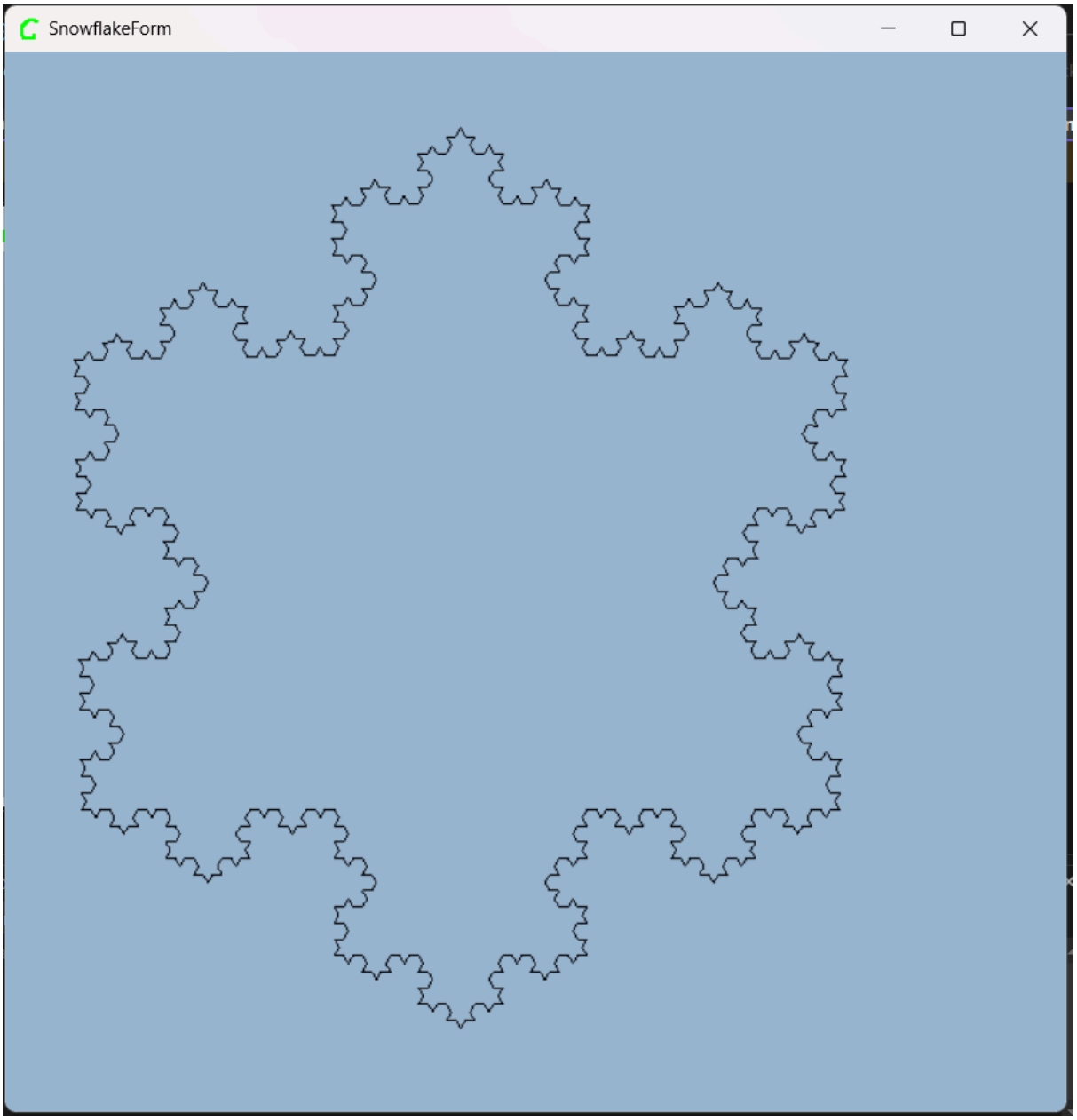
Якщо глибина рівна 0, тоді малюємо просту лінію між точками p1 та p2, інакше створюємо проміжні точки:
a - точка на 1/3 від відрізку між p1 та p2
b – точка на 2/3 від відрізку між p1 та p2
c – це вершина нового трикутника, яка обчислюється поворотом вектора на 60 градусів
Рекурсивно викликається функція для чотирьох нових відрізків. Таким чином, кожна лінія "розбивається" на 4 менші лінії, і цей процес повторюється до досягнення глибини

```
// function for drawing Loch's snowflake at one side
7 references
private void DrawKochSnowflake(Graphics g, PointF p1, PointF p2, int depth)
{
    if (depth == 0)
    {
        g.DrawLine(Pens.Black, p1, p2);
    }
    else
    {
        // calculate three intermadiate points
        PointF a = new PointF(p1.X + (p2.X - p1.X) * P, p1.Y + (p2.Y - p1.Y) * P);
        PointF b = new PointF(p1.X + (p2.X - p1.X) * 2 * P, p1.Y + (p2.Y - p1.Y) * 2 * P);

        // calculate coordinates of new triangle
        float angle = (float)(Math.PI / 3); // 60 degrees
        PointF c = new PointF(
            (float)(a.X + (b.X - a.X) * Math.Cos(angle) - (b.Y - a.Y) * Math.Sin(angle)),
            (float)(a.Y + (b.X - a.X) * Math.Sin(angle) + (b.Y - a.Y) * Math.Cos(angle))
        );

        // recursively draw a Koch's snowflake for each part
        DrawKochSnowflake(g, p1, a, depth - 1);
        DrawKochSnowflake(g, a, c, depth - 1);
        DrawKochSnowflake(g, c, b, depth - 1);
        DrawKochSnowflake(g, b, p2, depth - 1);
    }
}
```

Результат:



Висновок:

Виконуючи цю лабораторну роботу я ознайомився з фракталами, доволі складною геометричною фігурою сніжинкою Коха, і з тим як не важкими ітераційними кроками в залежності від глибини рекурсії можна побудувати дуже гарний візерунок.