Державний вищий навчальний заклад

«Прикарпатський національний університет імені Василя Стефаника» Кафедра комп’ютерних наук та інформаційних систем

**ЛАБОРАТОРНА РОБОТА №2**

з предмету «Алгоритми і структури даних»

Виконав:

студент групи КН-2

Гриньків В.І.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2020р.

к.т.н., доц. Никорак Я.Я.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2020р.

Івано-Франківськ – 2020

**1 і 2.** **Записати програму побудови трикутник Серпінського, записати програму для побудови килима Серпінського**

Код програми:

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Serpinski

{

public partial class Form1 : Form

{

private const int Level = 5;

//Висота і ширина

private int \_width;

private int \_height;

// Bitmap для фракталу

private Bitmap \_fractal;

// пікчр бокс

private Graphics \_graph;

public Form1()

{

InitializeComponent();

//ініціалізую висоту і ширину

\_width = FractalPictureBox.Width;

\_height = FractalPictureBox.Height;

}

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

{

//бітмап для трикутника

\_fractal = new Bitmap(\_width, \_height);

// створюю новий об'єкт Graphics з вказаного Bitmap

\_graph = Graphics.FromImage(\_fractal);

//вершини трикутника

PointF topPoint = new PointF(\_width / 2f, 0);

PointF leftPoint = new PointF(0, \_height);

PointF rightPoint = new PointF(\_width, \_height);

//викликаю функцію малювання

DrawTriangle(Level, topPoint, leftPoint, rightPoint);

//виводжу що вийшло

FractalPictureBox.BackgroundImage = \_fractal;

}

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

{

//бітмап для прямокутника

\_fractal = new Bitmap(\_width, \_height);

\_graph = Graphics.FromImage(\_fractal);

//створюю прямокутник и викликаю функція малювання ковра

RectangleF carpet = new RectangleF(0, 0, \_width, \_height);

DrawCarpet(Level, carpet);

//виводжу

FractalPictureBox.BackgroundImage = \_fractal;

}

private void DrawTriangle(int level, PointF top, PointF left, PointF right)

{

//перевіряжмо чи намолювалося

if (level == 0)

{

PointF[] points = new PointF[3]

{

top, right, left

};

//червоний трикутник

\_graph.FillPolygon(Brushes.Red, points);

}

else

{

//вичисляю середню точу

var leftMid = MidPoint(top, left); //ліва сторона

var rightMid = MidPoint(top, right); //права сторона

var topMid = MidPoint(left, right); // основа

//Рекурсивний виклик функції для кожного з трикутників

DrawTriangle(level - 1, top, leftMid, rightMid);

DrawTriangle(level - 1, leftMid, left, topMid);

DrawTriangle(level - 1, rightMid, topMid, right);

}

}

//Рахую координати середньої точки

private PointF MidPoint(PointF p1, PointF p2)

{

return new PointF((p1.X + p2.X) / 2f, (p1.Y + p2.Y) / 2f);

}

private void DrawCarpet(int level, RectangleF carpet)

{

//Якщо домалювало

if (level == 0)

{

//Малюю прямокутник

\_graph.FillRectangle(Brushes.OrangeRed, carpet);

}

else

{

// ділю прямокутник на 9 частин

var width = carpet.Width / 3f;

var height = carpet.Height / 3f;

// (x1, y1) - координати лівої верхньої вершини прямокутника

// від неї відраховую інші частини(координати) маленьких прямокутників

var x1 = carpet.Left;

var x2 = x1 + width;

var x3 = x1 + 2f \* width;

var y1 = carpet.Top;

var y2 = y1 + height;

var y3 = y1 + 2f \* height;

DrawCarpet(level - 1, new RectangleF(x1, y1, width, height)); // лівий 1(верхній)

DrawCarpet(level - 1, new RectangleF(x2, y1, width, height)); // середній 1

DrawCarpet(level - 1, new RectangleF(x3, y1, width, height)); // правий 1

DrawCarpet(level - 1, new RectangleF(x1, y2, width, height)); // лівий 2

DrawCarpet(level - 1, new RectangleF(x3, y2, width, height)); // правий 2

DrawCarpet(level - 1, new RectangleF(x1, y3, width, height)); // лівий 3

DrawCarpet(level - 1, new RectangleF(x2, y3, width, height)); // середній 3

DrawCarpet(level - 1, new RectangleF(x3, y3, width, height)); // правий 3

}

}

}

}

**3. Записати програму побудови дерева Піфагора**

**Код виконання:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace PythagorasTree

{

public partial class Form1 : Form

{

public Graphics g; //Графік

public Bitmap map; //Бітмап

public Pen p; //Ручка

public double angle = Math.PI / 2; //Кут повороту на 90 градусів

public double ang1 = Math.PI / 4; //Кут повороту на 45 градусів

public double ang2 = Math.PI / 6; //Кут повороту на 30 градусів

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

//x и y - координати вершини

//a - фіксує кількість ітерацій в функції

//kut - кут повороту на кожній ітерації

public int DrawTree(double x, double y, double a, double kut)

{

if (a > 2)

{

a \*= 0.7;

//Координати для молодшої вершини

double xnew = Math.Round(x + a \* Math.Cos(kut)),

ynew = Math.Round(y - a \* Math.Sin(kut));

//лінія між вершинами

g.DrawLine(p, (float)x, (float)y, (float)xnew, (float)ynew);

//Переписую координати

x = xnew;

y = ynew;

//Рекурсія для лівої і правої молодших вершин

DrawTree(x, y, a, kut + ang1);

DrawTree(x, y, a, kut - ang2);

}

return 0;

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

}

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

{

map = new Bitmap(pictureBox1.Width, pictureBox1.Height);//Підключаю бітмап

g = Graphics.FromImage(map); //Підключаю графік

g.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.AntiAlias;//Включаем смуз

p = new Pen(Color.Black); //Колір чорний

//Виклик рекурсивної функції яка малює дерево

DrawTree(300, 450, 200, angle);

//З бітмапу на пікчрокс

pictureBox1.BackgroundImage = map;

}

}

}

**4. Записати програму побудови множини Кантора**

**Код виконання:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Lab1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{ }

private void pictureBox1\_Click(object sender, EventArgs e)

{ }

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

{

Graphics g = pictureBox1.CreateGraphics();

//очищує вікно

g.Clear(Color.White);

//рахує довжину відрізка з текстбоксу

int a = Int32.Parse(textBox1.Text);

//рахує координату х

int x = 610 / 2 - a / 2;

//функція малювання

DrawB(x, 10, a);

}

private void DrawB(int x, int y, int width)

{

Graphics g = pictureBox1.CreateGraphics();

SolidBrush Black = new SolidBrush(Color.Black);

Pen myPen = new Pen(Color.Black, 1);

//умова виклику і малювання

if (width >= 3)

{

//Відрізки малюємо в формі прямокутників

g.DrawRectangle(myPen, x, y, width, 12);

g.FillRectangle(Black, x, y, width, 12);

y = y + 40; //Зсув вниз

//Функція для двох отриманих відрізків

DrawB(x + width \* 2 / 3, y, width / 3);

DrawB(x, y, width / 3);

}

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{ }

private void label1\_Click(object sender, EventArgs e)

{ }

}

}

**6. Записати програму побудови сніжинки Коха.**

**Код виконання:**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace koh

{

public partial class Form1 : Form

{

static Pen pen1;

static Graphics g;

static Pen pen2;

public Form1()

{

InitializeComponent();

}

private void Draw(object sender, EventArgs e)

{

//Беру два кольори

pen1 = new Pen(Color.Green, 1);

pen2 = new Pen(Color.Blue, 1);

//Роблю графік

g = CreateGraphics();

g.Clear(Color.Black);//Замальовую екран чорним

//Стартові позиції трикутника

var point1 = new PointF(200, 200);

var point2 = new PointF(500, 200);

var point3 = new PointF(350, 400);

//Замалюєм трикутник

g.DrawLine(pen1, point1, point2);

g.DrawLine(pen1, point2, point3);

g.DrawLine(pen1, point3, point1);

//Малюємо три криві коха на стронах трикутника

Fractal(point1, point2, point3, 5);

Fractal(point2, point3, point1, 5);

Fractal(point3, point1, point2, 5);

}

static int Fractal(PointF p1, PointF p2, PointF p3, int iter)

{

if (iter > 0) //условие выхода из рекурсии

{

//Середня третина відрізку

var p4 = new PointF((p2.X + 2 \* p1.X) / 3, (p2.Y + 2 \* p1.Y) / 3);

var p5 = new PointF((2 \* p2.X + p1.X) / 3, (p1.Y + 2 \* p2.Y) / 3);

//Координати вершини кута

var ps = new PointF((p2.X + p1.X) / 2, (p2.Y + p1.Y) / 2);

var pn = new PointF((4 \* ps.X - p3.X) / 3, (4 \* ps.Y - p3.Y) / 3);

g.DrawLine(pen1, p4, pn);

g.DrawLine(pen1, p5, pn);

g.DrawLine(pen2, p4, p5);

//Рекурсивний виклик функції

Fractal(p4, pn, p5, iter - 1);

Fractal(pn, p5, p4, iter - 1);

Fractal(p1, p4, new PointF((2 \* p1.X + p3.X) / 3, (2 \* p1.Y + p3.Y) / 3), iter - 1);

Fractal(p5, p2, new PointF((2 \* p2.X + p3.X) / 3, (2 \* p2.Y + p3.Y) / 3), iter - 1);

}

return iter;

}

}

}