ГУАП

КАФЕДРА № 33

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | А.И. Дубинин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| УПРАВЛЕНИЕ ЭФФЕКТИВНОСТЬЮ ИСПОЛЬЗОВАНИЯ СИСТЕМНЫХ РЕСУРСОВ ПРИ ПОСТРОЕНИИ ФРАКТАЛЬНЫХ СТРУКТУР |
| по курсу: Технологии и методы программирования |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3136 |  |  |  | В.А. Панчук |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023 г.

Оглавление

[1. Цель лабораторной работы 3](#_Toc133771499)

[2. Постановка задачи 3](#_Toc133771500)

[3. Часть 1 3](#_Toc133771501)

[3.1. Листинг программы 4](#_Toc133771502)

[3.2. Пример работы программы 9](#_Toc133771503)

[4. Часть 2(С#) 9](#_Toc133771504)

[4.1.1. Листинг алгоритма 11](#_Toc133771505)

[4.1.1. Пример работы программы 14](#_Toc133771506)

[5. Вывод 16](#_Toc133771507)

# Цель лабораторной работы

Рассмотреть понятие рекурсии, научиться проводить анализ рекурсивных алгоритмов и состояние стека при выполнении рекурсий. Изучить свойства и методы классов для растровой графики. Сформировать регулярные фрактальные объекты и сигналы Фурье.

# Постановка задачи

В первой части лабораторной работы на языке C# необходимо разработать программный редактор графических изображений.

В лабораторной работе, задание из второй части (с использованием языка C#) выполнялось под вариантом 4.

Вторая часть лабораторной работы включает в себя разработку программы на языке C#, которая может отрисовывать регулярные фракталы дерева Пифагора и салфетки Серпинского. Программа должна была быть выполнена с использованием интерфейса Windows Forms. Обработанные исключения фиксировать на главной форме проекта и в текстовом файле с указанием даты и времени проявления исключений, а также свойств класса Exception. Также необходимо создать программу, демонстрирующую работу рекурсивного алгоритма в соответствии с вариантом 4.

# Часть 1

Разработать программный редактор графических изображений, способный:

* создавать/открывать/сохранять/редактировать масштабируемое растровое изображение;
* использовать средства выбора параметров отрисовки (ширина/тип линии, способ отрисовки, цвет линии/фона/заливки);
* отображать «живые» сигналы Фурье с прямоугольной/треугольной/пилообразной/ формой огибающей гармоник;
* предусматривать эффективное использование системных ресурсов,
* изображение должно автоматически перерисовываться при изменении размеров форм, их закрытии и свертывании.

## Листинг программы

Основная программа:

public partial class paint : Form

{

private int mode;

private Point movePt;

private Point nullPt = new Point(int.MaxValue, 0);

private SolidBrush brush = new SolidBrush(Color.White);

private Pen pen = new Pen(Color.Black);

private Point startPt;

private Color col = Color.Black;

int k1 = 1, k2 = 0;

int A = 50;

int F = 1;

int nterms = 3;

private makePanel makePanel = new makePanel();

public paint()

{

InitializeComponent();

AddOwnedForm(makePanel);

openFileDialog1.InitialDirectory = saveFileDialog1.InitialDirectory = Directory.GetCurrentDirectory();

makePanel.numericUpDown1.Value = panel1.ClientSize.Width;

makePanel.numericUpDown2.Value = panel1.ClientSize.Height;

makePanel.button1\_Click(this, null);

pen.StartCap = pen.EndCap = LineCap.Round;

pen.Alignment = PenAlignment.Inset;

}

public paint(Form1 f)

{

InitializeComponent();

AddOwnedForm(makePanel);

openFileDialog1.InitialDirectory = saveFileDialog1.InitialDirectory = Directory.GetCurrentDirectory();

makePanel.numericUpDown1.Value = panel1.ClientSize.Width;

makePanel.numericUpDown2.Value = panel1.ClientSize.Height;

makePanel.button1\_Click(this, null);

pen.StartCap = pen.EndCap = LineCap.Round;

pen.Alignment = PenAlignment.Inset;

}

private void ReversibleDraw()

{

Point p1 = pictureBox1.PointToScreen(startPt), p2 = pictureBox1.PointToScreen(movePt);

if (mode == 1)

ControlPaint.DrawReversibleLine(p1, p2, Color.Black);

else

ControlPaint.DrawReversibleFrame(PtToRect(p1, p2), Color.Black, FrameStyle.Thick);

}

private void DrawFigure(Rectangle r, Graphics g)

{

g.FillRectangle(brush, r);

g.DrawRectangle(pen, r);

}

private Rectangle PtToRect(Point p1, Point p2)

{

int x = Math.Min(p1.X, p2.X),

y = Math.Min(p1.Y, p2.Y),

w = Math.Abs(p2.X - p1.X),

h = Math.Abs(p2.Y - p1.Y);

return new Rectangle(x, y, w, h);

}

private void button3\_Click(object sender, EventArgs e)

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

string s = openFileDialog1.FileName;

try

{

Image im = new Bitmap(s);

Graphics g = Graphics.FromImage(im);

g.Dispose();

if (pictureBox1.Image != null)

pictureBox1.Image.Dispose();

pictureBox1.Image = im;

}

catch

{

MessageBox.Show("File " + s + " has a wrong format.", "Error");

return;

}

Text = "Image Editor - " + s;

saveFileDialog1.FileName = Path.ChangeExtension(s, "png");

openFileDialog1.FileName = "";

}

}

private void label6\_Click(object sender, EventArgs e)

{

colorDialog1.Color = label6.BackColor;

if (colorDialog1.ShowDialog() == DialogResult.OK)

{

label6.BackColor = colorDialog1.Color;

col = colorDialog1.Color;

}

}

private void label6\_BackColorChanged(object sender, EventArgs e)

{

pen.Color = label6.BackColor;

}

private void label7\_BackColorChanged(object sender, EventArgs e)

{

//заливка

brush.Color = label7.BackColor;

}

private void label7\_Click(object sender, EventArgs e)

{

Label lb = sender as Label;

colorDialog1.Color = lb.BackColor;

if (colorDialog1.ShowDialog() == DialogResult.OK)

lb.BackColor = colorDialog1.Color;

}

private void button2\_Click(object sender, EventArgs e)

{

makePanel.ActiveControl = makePanel.numericUpDown1;

if (makePanel.ShowDialog() == DialogResult.OK)

{

saveFileDialog1.FileName = "";

Text = "Image Editor";

}

}

private void button4\_Click(object sender, EventArgs e)

{

//startPt = nullPt;

string s0 = saveFileDialog1.FileName;

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

string s = saveFileDialog1.FileName;

if (s.ToUpper() == s0.ToUpper())

{

s0 = Path.GetDirectoryName(s0) + "\\($$##$$).png";

pictureBox1.Image.Save(s0);

pictureBox1.Image.Dispose();

File.Delete(s);

File.Move(s0, s);

pictureBox1.Image = new Bitmap(s);

}

else

pictureBox1.Image.Save(s);

Text = "Image Editor - " + s;

}

}

private void pictureBox1\_MouseDown(object sender, MouseEventArgs e)

{

movePt = startPt = e.Location;

}

private void pictureBox1\_MouseMove(object sender, MouseEventArgs e)

{

//label1.Text = string.Format("X,Y: {0},{1}", e.X, e.Y);

if (startPt == nullPt)

return;

if (e.Button == MouseButtons.Left)

switch (mode)

{

case 0:

Graphics g = Graphics.FromImage(pictureBox1.Image);

g.DrawLine(pen, startPt, e.Location);

g.Dispose();

startPt = e.Location;

pictureBox1.Invalidate();

pictureBox1.Update();

break;

case 1:

ReversibleDraw();

movePt = e.Location;

ReversibleDraw();

break;

case 2:

ReversibleDraw();

movePt = e.Location;

ReversibleDraw();

break;

}

}

private void pictureBox1\_MouseUp(object sender, MouseEventArgs e)

{

if (startPt == nullPt)

return;

if (mode >= 1)

{

ReversibleDraw();

Graphics g = Graphics.FromImage(pictureBox1.Image);

switch (mode)

{

case 0:

g.DrawLine(pen, startPt, movePt);

break;

case 1:

g.DrawLine(pen, startPt, movePt);

break;

case 2:

DrawFigure(PtToRect(startPt, movePt), g);

this.Invalidate();

break;

}

g.Dispose();

pictureBox1.Invalidate();

}

}

private void button5\_Click(object sender, EventArgs e)

{

using (Graphics g = Graphics.FromImage(pictureBox1.Image))

g.Clear(Color.White);

pictureBox1.Invalidate();

}

private void numericUpDown4\_ValueChanged(object sender, EventArgs e)

{

pen.Width = (int)numericUpDown4.Value;

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

RadioButton rb = sender as RadioButton;

if (!rb.Checked)

return;

mode = rb.TabIndex;

}

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

{

redrawFourier();

}

private void redrawFourier()

{

Graphics g = Graphics.FromImage(pictureBox1.Image);

g.Clear(Color.White);

g.DrawLine(pen, 0, pictureBox1.Height / 2, pictureBox1.Width, pictureBox1.Height / 2);

pen.Width = Convert.ToInt32(numericUpDown4.Value);

pen.Color = col;

int Interval = pictureBox1.Width;

double yp = 0, yy1 = 0, yy2 = 0;

int angle = 0;

int xtemp = 0;

int ytemp = pictureBox1.Height / 2;

for(int i = 0; i < Interval; i++)

{

for(int j = 1; j < nterms; j++)

{

yy1 = A / ((k1 \* j) + k2);

double arg = ((j \* k1) + k2) \* F \* 0.01397 \* angle;

yy2 = Math.Sin(arg);

yp = yp + yy1 \* yy2;

}

xtemp = i;

g.DrawLine(pen, xtemp, ytemp, i, pictureBox1.Height / 2 + (int)Math.Truncate(yp));

yp = 0;

angle++;

}

pen.Color = Color.Black;

g.Dispose();

pictureBox1.Invalidate();

}

private void numericUpDown1\_ValueChanged(object sender, EventArgs e)

{

nterms = (int)numericUpDown1.Value;

redrawFourier();

}

private void numericUpDown2\_ValueChanged(object sender, EventArgs e)

{

A = (int)numericUpDown2.Value;

redrawFourier();

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

switch ((int)comboBox1.SelectedIndex)

{

case 0:

k1 = 1;

k2 = 0;

break;

case 1:

k1 = 2;

k2 = 1;

break;

case 2:

k1 = 2;

k2 = -1;

break;

}

}

private void numericUpDown3\_ValueChanged(object sender, EventArgs e)

{

F = (int)numericUpDown3.Value;

redrawFourier();

}

}

Дополнительная панель для изменения высоты и ширины окна:

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

{

int w = (int)numericUpDown1.Value,

h = (int)numericUpDown2.Value;

Image im = new Bitmap(w, h);

Graphics g = Graphics.FromImage(im);

g.Clear(Color.White);

g.Dispose();

PictureBox p = Owner.Controls["panel1"].Controls["pictureBox1"] as PictureBox;

if (p.Image != null)

p.Image.Dispose();

p.Image = im;

}

## Пример работы программы

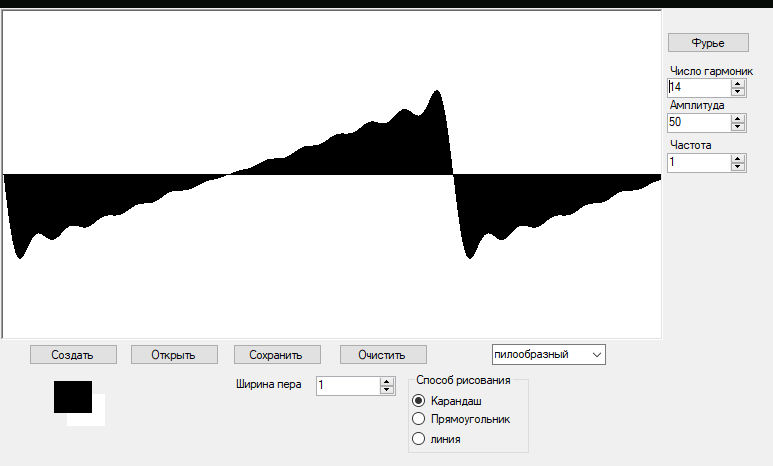


Рисунок − Отображение живого сигнала Фурье с пилообразной формой огибающей гармоник

# Часть 2(С#)

1) Создать программу, отрисовывающую регулярные фракталы дерева Пифагора и салфетки Серпинского. Число рекурсивных вызовов изменяется пользователем и автоматически перерисовывает фрактал. Заливка фигуры должна меняться градиентно на каждой рекурсивной итерации (Использовать асинхронный режим работы).

2) Создать программу, демонстрирующую работу рекурсивного алгоритма в соответствии с 4 вариантом.

3) Программа должна представить результат в графическом виде (для текстовых рекурсий в том числе).

4) Для координатных осей (в вариантах с графиками функций) должны отражаться засечки отсчетов с указанием значений.

Дополнительное задание: сделать дерево утолщенным.

Алгоритм построения дерева Пифагора и ковра Серпинского представлен на рисунке 2 и 3 соответственно.

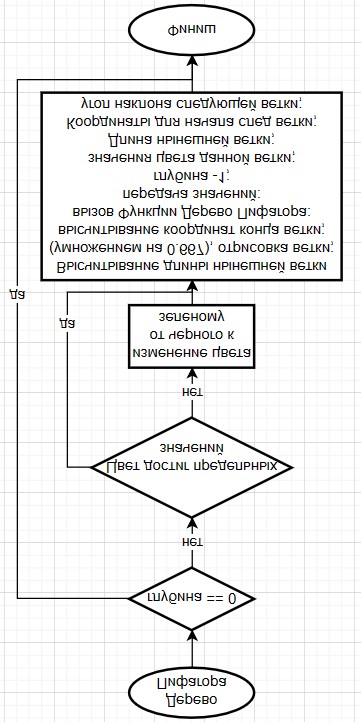


Рисунок − Алгоритм для дерева Пифагора

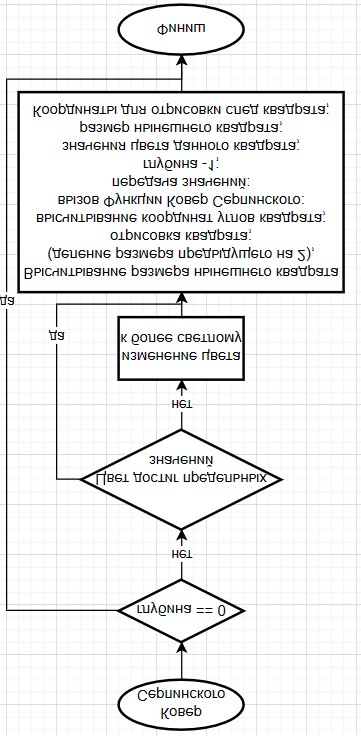


Рисунок − Алгоритм для ковра Серпинского

## Листинг алгоритма

public partial class fractal : Form

{

Form1 fm1;

Graphics g;

int[] array = new int[5];

public fractal()

{

InitializeComponent();

}

public fractal(Form1 f)

{

InitializeComponent();

fm1 = f;

}

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

{

trackBar4.BackColor = Color.LightBlue;

trackBar3.BackColor = Color.LightGreen;

trackBar2.BackColor = Color.LightPink;

}

private double find\_degree(double a, int i)

{

if (i>0)

{

a \*= find\_degree(a, i - 1);

}

return a;

}

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

{

g = pictureBox1.CreateGraphics();

switch (comboBox2.SelectedIndex)

{

case 0:

tree(pictureBox1.Width / 2, pictureBox1.Height, Math.PI / 2, ((pictureBox1.Height / 6) + (pictureBox1.Width / 6)) / 2, Convert.ToInt32(numericUpDown2.Value), trackBar2.Value\*25, trackBar3.Value\*25, trackBar4.Value\*25);

break;

case 1:

carpet(pictureBox1.Width / 2, pictureBox1.Height / 2, ((pictureBox1.Height / 6) + (pictureBox1.Width / 6)) / 2, Convert.ToInt32(numericUpDown2.Value),trackBar2.Value\*25, trackBar3.Value\*25, trackBar4.Value\*25);

break;

}

}

private void Form4\_Resize(object sender, EventArgs e)

{

pictureBox1.Refresh();

this.button1\_Click(sender, e);

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

pictureBox1.Refresh();

this.button1\_Click(sender, e);

}

private void carpet(int x, int y, int size, int step, int R, int G, int B)

{

if (step != 0)

{

int x1 = x - size / 2;

int x2 = x + size / 2;

int y1 = y - size / 2;

int y2 = y + size / 2;

if (R > 255)

R = 255;

if (G > 255)

G = 255;

if (B > 255)

B = 255;

Pen my\_pen\_temp = new Pen(Color.FromArgb(R, G, B));

g.DrawRectangle(my\_pen\_temp, x - size / 2, y - size / 2, size, size);

carpet(x1, y1, size / 2, step - 1, R + 10, G + 30, B + 30);

carpet(x1, y2, size / 2, step - 1, R + 10, G + 30, B + 30);

carpet(x2, y1, size / 2, step - 1, R + 10, G + 30, B + 30);

carpet(x2, y2, size / 2, step - 1, R + 10, G + 30, B + 30);

}

}

private void tree(double x0, double y0, double a, double L, int N, int R, int G, int B)

{

const double k = 0.667;

double x1, y1;

if (N > 0)

{

if (R > 255)

R = 255;

if (G > 255)

G = 255;

if (B > 255)

B = 255;

Pen my\_pen\_temp = new Pen(Color.FromArgb(R, G, B), (float)(L/10));

x1 = x0 + L \* Math.Cos(a);

y1 = y0 - L \* Math.Sin(a);

g.DrawLine(my\_pen\_temp, (float)x0, (float)y0, (float)x1, (float)y1);

tree(x1, y1, a + Math.PI / 3, L \* k, N - 1, R + 10, G + 30, B + 30);

tree(x1, y1, a + Math.PI / 7, L \* k, N - 1, R + 10, G + 30, B + 30);

tree(x1, y1, a - Math.PI / 2, L \* k, N - 1, R + 10, G + 30, B + 30);

tree(x1, y1, a - Math.PI / 5, L \* k, N - 1, R + 10, G + 30, B + 30);

}

}

private void button2\_Click(object sender, EventArgs e)

{

pictureBox1.Invalidate();

}

private void button3\_Click(object sender, EventArgs e)

{

try

{

g = pictureBox1.CreateGraphics();

g.Clear(Color.White);

int intValue;

double number;

bool succes1 = Double.TryParse(textBox1.Text, out number);

bool succes2 = Int32.TryParse(textBox2.Text, out intValue);

if (!succes1 || !succes2 || intValue > 100 || intValue < 0) throw new Exception("Введенные данные не соответствуют требованиям");

number = find\_degree(number, intValue);

System.Drawing.Font drawFont = new System.Drawing.Font("Arial", 16);

System.Drawing.SolidBrush drawBrush = new System.Drawing.SolidBrush(System.Drawing.Color.Black);

float x = 150.0F;

float y = 50.0F;

System.Drawing.StringFormat drawFormat = new System.Drawing.StringFormat();

g.DrawString(Convert.ToString(number), drawFont, drawBrush, x, y, drawFormat);

drawFont.Dispose();

drawBrush.Dispose();

g.Dispose();

}

catch (Exception error)

{

fm1.textBox1.Text = error.Message + '\n';

fm1.textBox1.Text += error.StackTrace;

}

}

## Пример работы программы

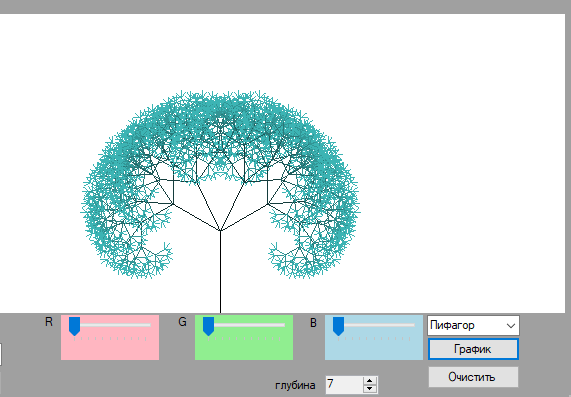


Рисунок − Дерево Пифагора с глубиной 7

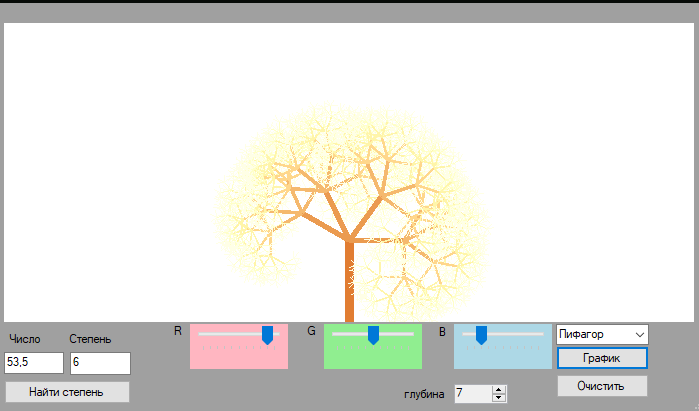


Рисунок − Дерево Пифагора с глубиной 7 с выполненным дополнительным заданием

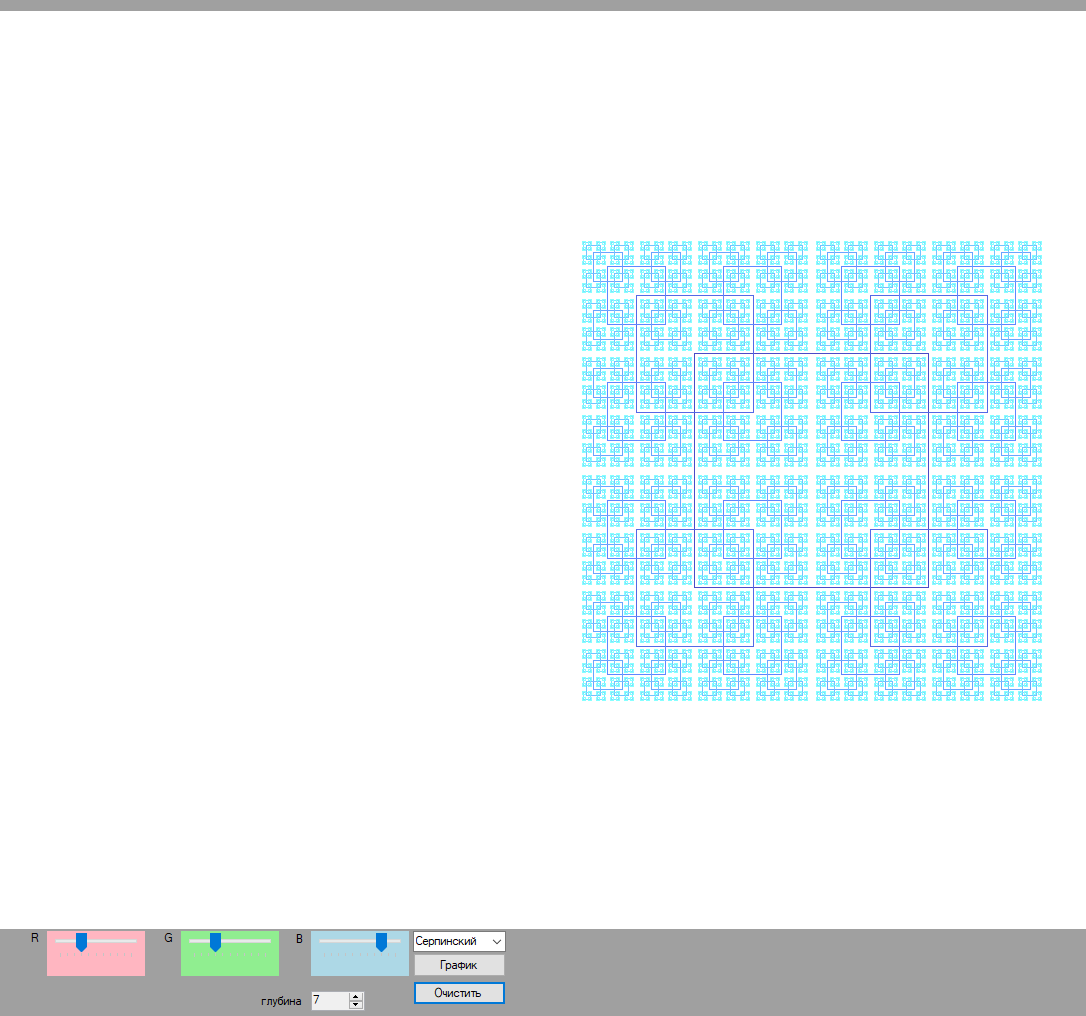


Рисунок − Салфетка Серпинского с глубиной 7

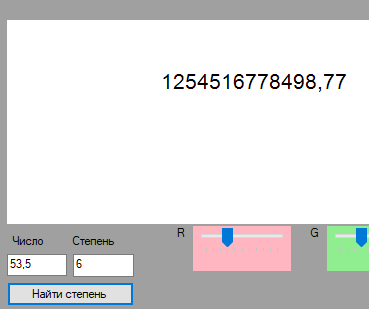


Рисунок − Задание в соответствии с вариантом 4

# Вывод

Рассмотрели понятие рекурсии, научились проводить анализ рекурсивных алгоритмов и состояние стека при выполнении рекурсий. Изучили свойства и методы классов для растровой графики. Сформировали регулярные фрактальные объекты и сигналы Фурье.