Министерство науки и высшего образования Российской Федерации

**Муромский институт (филиал)**

федерального государственного бюджетного образовательного учреждения высшего образования

**«Владимирский государственный университет   
имени Александра Григорьевича и Николая Григорьевича Столетовых»**

(МИВлГУ)

Факультет НАЗВАНИЕ ФАКУЛЬТЕТА



Кафедра НАЗВАНИЕ КАФЕДРЫ

КУРСОВАЯ

РАБОТА

по ООП

(наименование дисциплины)

Тема Генератор фрактала «Снежинка Коха»

Руководитель

ФИО РУКОВОДИТЕЛЯ

(оценка) (фамилия, инициалы)

Члены комиссии

(подпись) (дата)

Студент НАЗВАНИЕ ГРУППЫ

(подпись) (Ф.И.О.) (группа)

Буканова В. Д.

(подпись) (Ф.И.О.) (фамилия, инициалы)

(подпись) (дата)

Муром 2024

В курсовой работе согласно задания разработан генератор фрактала «Снежинка Коха». В ходе работы сделан анализ технического задания, сформированы требования к фракталу, выбраны средства реализации, а также язык программирования. Фрактал реализован на языке C#, среда разработки Visual Studio 2022. Приложение визуально демонстрирует генерацию «Снежинки Коха» с различным количеством итераций.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc156396770)

[1. Анализ технического задания 5](#_Toc156396771)

[1.1 Анализ предметной области 5](#_Toc156396772)

[1.2 Формирование требований к приложению. 7](#_Toc156396773)

[2. Разработка алгоритма 8](#_Toc156396774)

[3. Реализация программы 11](#_Toc156396775)

[4. Руководство по программному продукту 13](#_Toc156396776)

[4.1 Руководство пользователя 13](#_Toc156396777)

[4.2 Руководство администратора 14](#_Toc156396778)

[5. Тестирование программного продукта 15](#_Toc156396779)

[ЗАКЛЮЧЕНИЕ 16](#_Toc156396780)

[Список литературы 17](#_Toc156396781)

ВВЕДЕНИЕ

Компьютерная обработка и распознавание изображений представляет собой быстро развивающуюся самостоятельную дисциплину. Компьютерная обработка изображений предполагает обработку цифровых изображений с помощью компьютеров или специализированных устройств, построенных на цифровых сигнальных процессорах.

При этом под обработкой изображений понимается не только улучшение зрительного восприятия изображений, но и классификация объектов, выполняемая при анализе изображений.

Методы цифровой обработки широко применяются в промышленности, искусстве, медицине, космосе. Они применяются при управлении процессами, автоматизации обнаружения и сопровождения объектов, распознавании образов и во многих других приложениях.

Компьютерная обработка изображений применяется в задачах экспертизы живописи неразрушающими методами. Для восстановления старых фильмов применяются методы автоматической компенсации дефектов видеоматериала, полученного после преобразования киноизображения в видео. Сегодня трудно представить область деятельности, в которой можно обойтись без компьютерной обработки изображений.

Применяя методы объектно-ориентированного программирования в цифровой обработке изображений, позволяет заметно повысить скорость и качество в работе алгоритмов, а также упрощается реализация сложных алгоритмов. ООП позволяет применять визуальные инструменты, а также визуальную демонстрацию результатов работы алгоритмов.

Темой данной курсовой работы является проектирование и разработка фрактала «Снежинка Коха».

Целью курсовой работы является изучение технологии ООП, а также приобретение практических навыков в цифровой обработке изображений.

Основными задачами работы являются: выбор средств разработки фрактала «Снежинка Коха», проектирование приложения, реализация фрактала «Снежинка Коха»;

1. Анализ технического задания

1.1 Анализ предметной области

В данном курсовой работе необходимо разработать приложение для генерации фрактала «Снежинка Коха».

Шведский математик Хельге Фон Кох в 1904 году описал кривую, воспользовавшись треугольником и методом самоподобия, в результате чего получилась фрактальная снежинка.

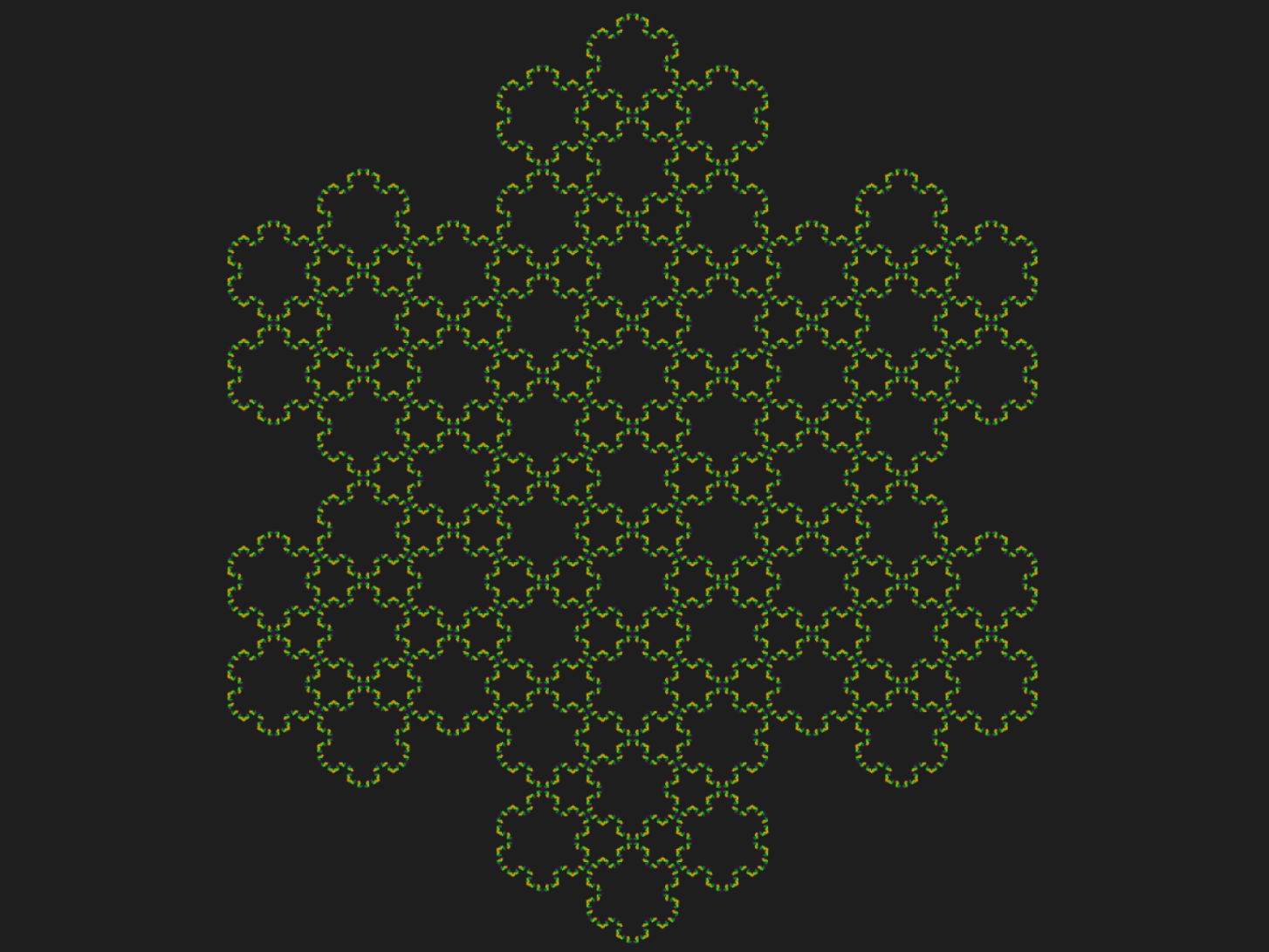


Рисунок 1 – Визуальное представление «Снежинки Коха»

На рисунке 2 показаны четыре итерации построения такой фигуры. Слева изображены исходные кривые, а справа — получившаяся из этих кривых снежинка. Нетрудно заметить, что в снежинке идеально вписывается как равносторонний треугольник, так и сама кривая.

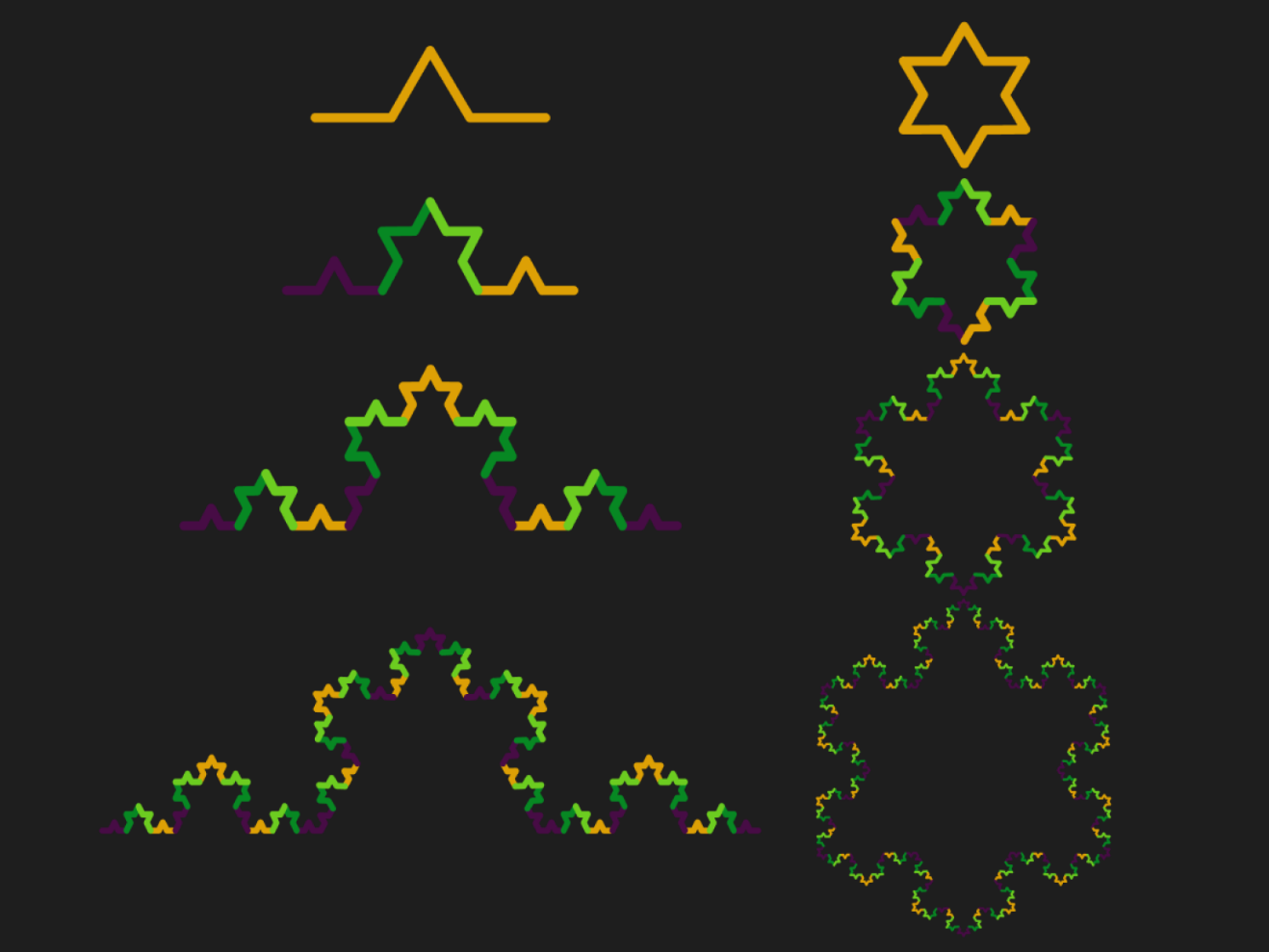


Рисунок 2- Ход построения «Снежинки Коха»

На какой бы итерации мы ни увеличили масштаб изображения, мы всегда сможем увидеть знакомый паттерн, как и с множеством Кантора. Посчитать периметр такой снежинки невозможно, потому что она может разрастаться всё дальше и дальше… Это ещё одно свойство фракталов — **бесконечность**.

Фракталы нашли применение в физике (моделирование сложных процессов и материалов), биологии (моделирование популяций, описание сложных ветвящихся структур), технике (фрактальные антенны), экономике. Существуют алгоритмы сжатия изображений с помощью фракталов. В компьютерной графике фракталы используются для построения изображений природных объектов – растений, ландшафтов, поверхности морей и т. д.

1.2 Формирование требований к приложению.

Исходя из приведенного выше анализа предметной области, сформулированы следующие основные требования к разрабатываемому приложению для генерации «Снежинки Коха»:

- программа должна быть реализована на языке C#

- программа не должна использовать сторонних зависимостей;

- в программе необходимо предусмотреть ввод пользователем количество итераций;

- реализация графического (динамического) вывода «Снежинки Коха»

2. Разработка алгоритма

Шаг 1:

Нарисуем равносторонний треугольник. Лучше всего, если длина сторон будет кратна 3, из-за природы этого фрактала.

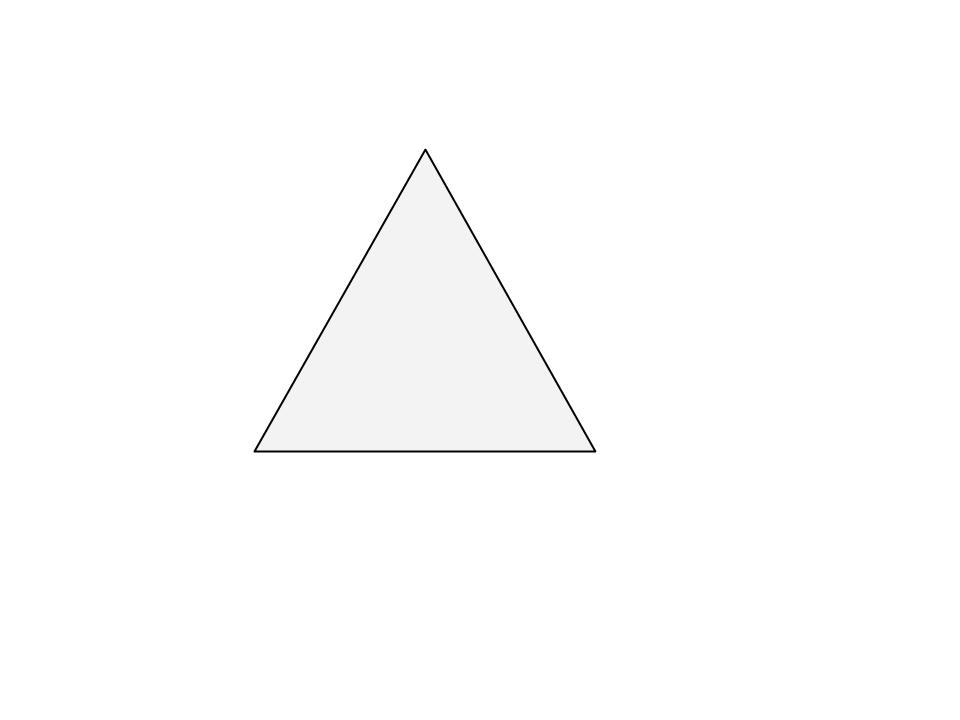


Рисунок 3 – Равносторонний треугольник

Шаг 2:

Разделим каждую сторону на три равные части.

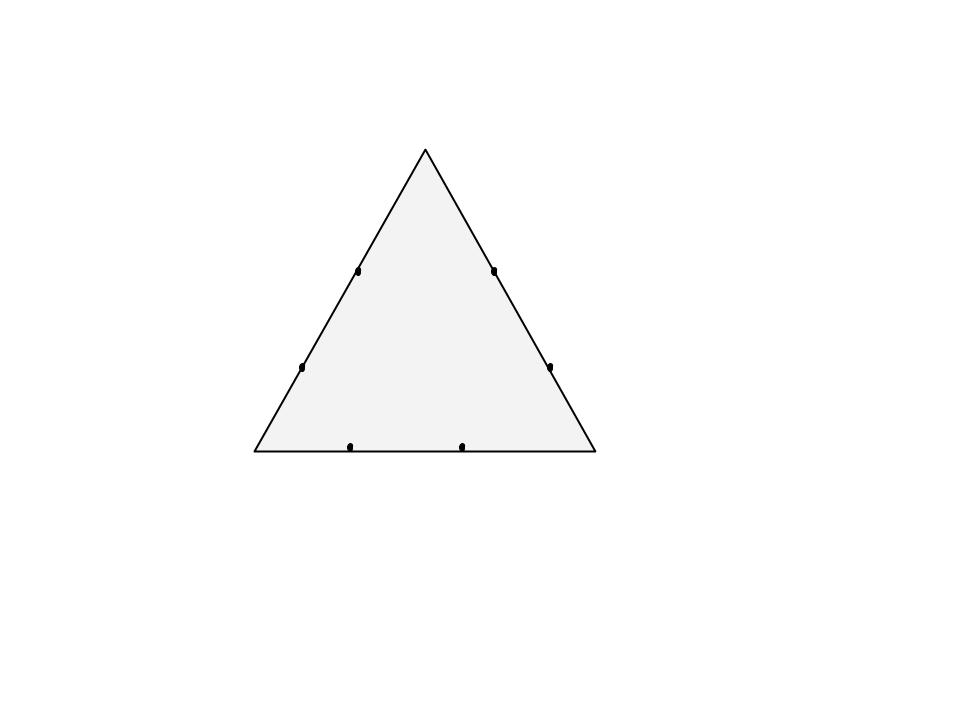


Рисунок 4 – Стороны треугольника разделены на 3 части

Шаг 3:

Нарисуем равносторонний треугольник на каждой средней части. Измерим длину средней трети, чтобы узнать длину сторон этих новых треугольников.

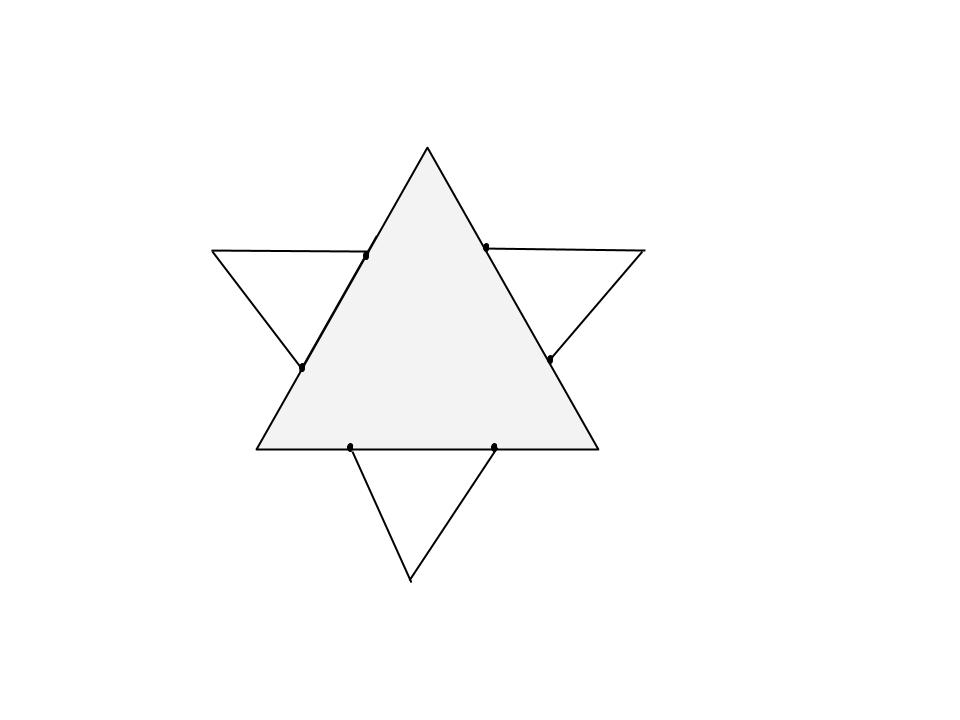


Рисунок 5 – Нарисован 2-й треугольник

Шаг 4:

Разделим каждую внешнюю сторону на трети. Можно видеть, что треугольники 2-го поколения частично покрывают первое. Эти три отрезка не следует разделять пополам.

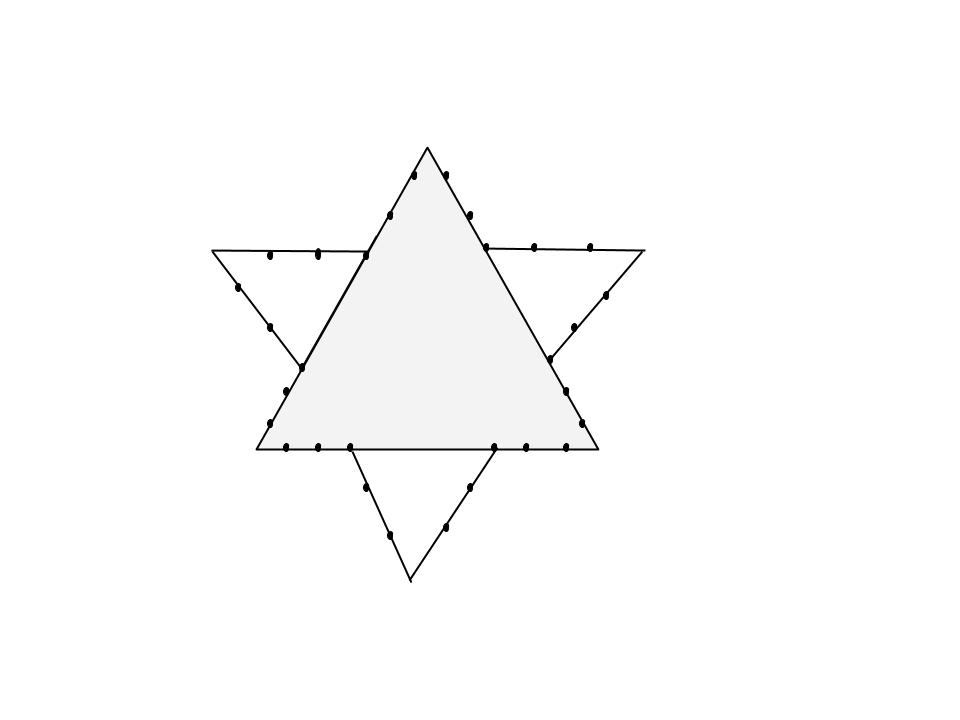


Рисунок 6 – Каждая вершина разделена на трети

Шаг 5:

Нарисуем равносторонний треугольник на каждой средней части.

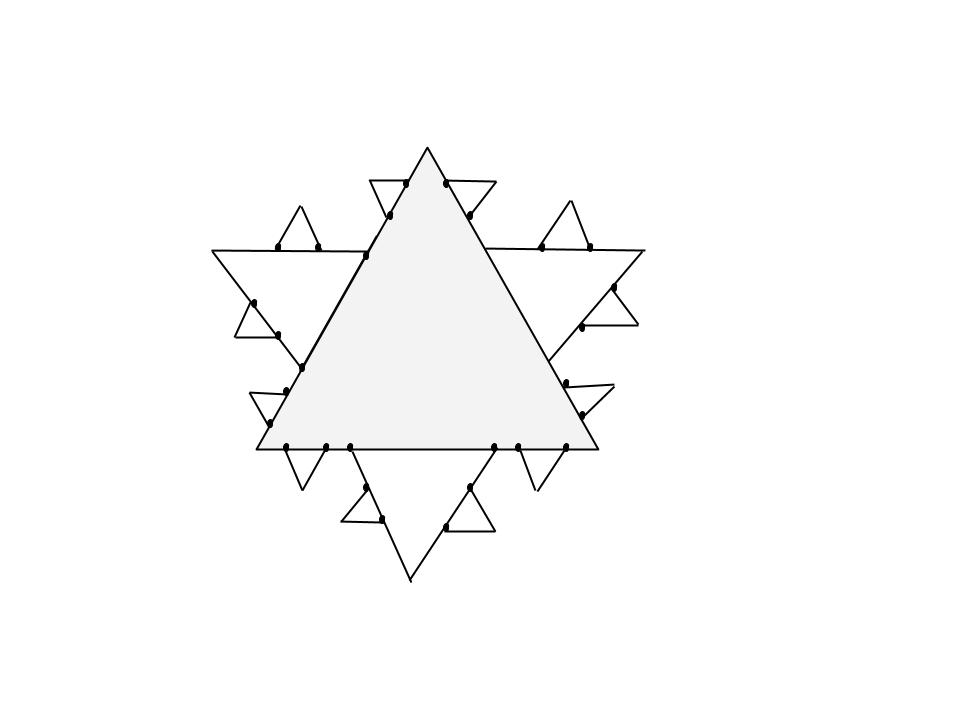


Рисунок 7 – Третье поколение треугольников

3. Реализация программы

Программа генерации «Снежинки Коха» состоит из одного класса Main. В классе есть 2 метода:

- Render;

- RenderKoсh;

Метод Render инициализирует 2 экземпляра Pen, для рисования треугольника. Экземпляры классов Pen рисуют линии разным цветом. Метод Render подготавливает холст, рисует исходный треугольник, а также запускает рекурсивно метод RenderKoch, с необходимым пользователю количеством итераций.

Метод RenderKoсh вызывается рекурсивно и на каждой итерации обрисовывает треугольник в нужной позиции (по алгоритму).

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Snowflake

{

public partial class Main : Form

{

static Pen pencil1;

static Pen pencil2;

static Graphics g;

public Main()

{

InitializeComponent();

}

private void Render(object sender, EventArgs e)

{

//Цвет первого карандаша

pencil1 = new Pen(Color.Lime, 1);

//Цвет второго карандаша

pencil2 = new Pen(Color.Orange, 1);

//Холст

g = CreateGraphics();

//Заливаем холст черным цветом

g.Clear(Color.Black);

//Координаты исходного треугольника

var point1 = new PointF(250, 250);

var point2 = new PointF(625, 250);

var point3 = new PointF(437.5f, 500);

//Рисуем исходный треугольник

g.DrawLine(pencil1, point1, point2);

g.DrawLine(pencil1, point2, point3);

g.DrawLine(pencil1, point3, point1);

//Три кривых Коха на сторонах треугольника

RenderKoсh(point1, point2, point3, (int)this.numericUpDown1.Value);

RenderKoсh(point2, point3, point1, (int)this.numericUpDown1.Value);

RenderKoсh(point3, point1, point2, (int)this.numericUpDown1.Value);

}

static int RenderKoсh(PointF p1, PointF p2, PointF p3, int iteration)

{

if (iteration > 0)

{

//Средняя треть отрезка

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);

//Координаты вершины угла

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);

//Отрисовка

g.DrawLine(pencil1, p4, pn);

g.DrawLine(pencil1, p5, pn);

g.DrawLine(pencil2, p4, p5);

//Рекурсия

RenderKoсh(p4, pn, p5, iteration - 1);

RenderKoсh(pn, p5, p4, iteration - 1);

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

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

}

return iteration;

}

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

{

Application.Exit();

}

}

}

4. Руководство по программному продукту

4.1 Руководство пользователя

Чтобы запустить приложение нужно выполнить файл Snowflake.exe.

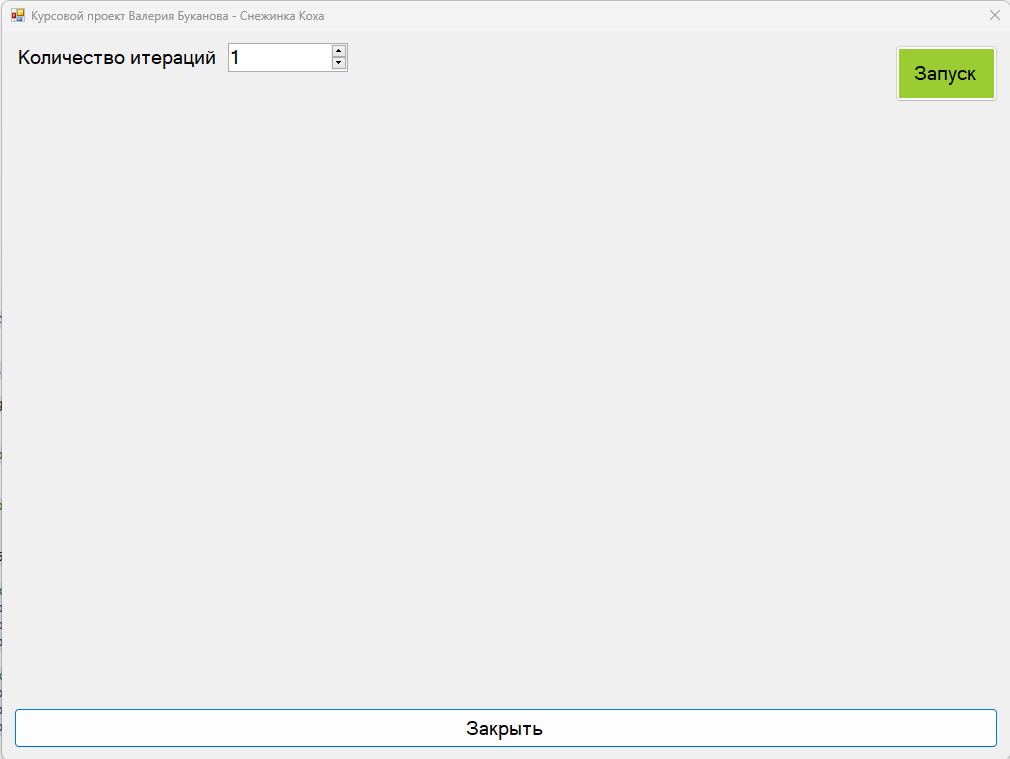


Рисунок 8 – Приложение запущено

Пользователь указывает нужное количество итераций и нажимает на кнопку «Запуск».

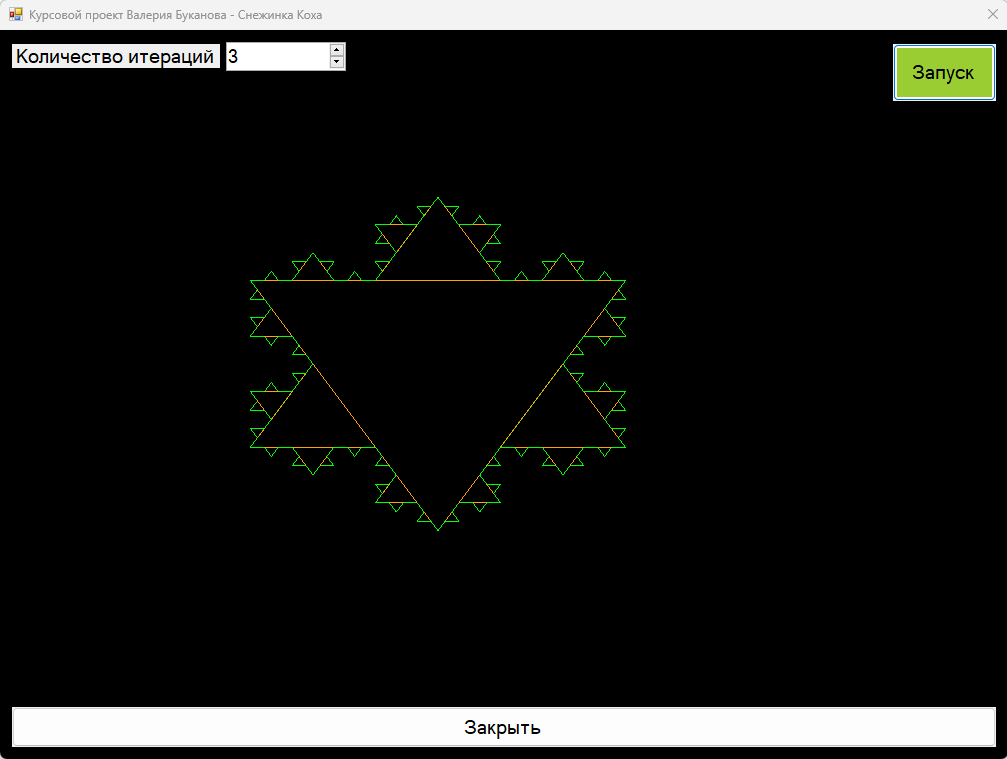


Рисунок 8 – «Снежинка Коха» построена

4.2 Руководство администратора

Для работы программы генерации «Снежинки Коха» на компьютере пользователя должна быть установлена любая операционная система из списка:

- Windows 7;

- Windows 8.1;

- Windows 10;

- Windows 11.

Для запуска приложения нужно установить Microsoft .NET Framework 4.8.

Администратору нужно перенести на ПК пользователя скомпилированное приложение, создав предварительно директорию, а также сообщить пользователю, где располагается оно располагается. Приложению не нужен доступ в интернет.

5. Тестирование программного продукта

Тестирование разработанного программного продукта выполняется с целью проверки соответствия реализованного функционала требованиям технического задания, а также проверки надежности работы и отсутствия ошибок в реализации функционала. Ниже приведена методика тестирования программного продукта.

Таблица 1 Методика тестирования программного продукта

|  |  |
| --- | --- |
| Выполняемая функция | Требуемый результат |
| 1. Запуск приложения | Отображается окно приложения |
| 2. Ввести 3 итерации и нажать кнопку «Запустить» | В окне строится «Снежинка Коха» |
| 3. Ввести 5 итерации и нажать кнопку «Запустить» | В окне строится «Снежинка Коха» |
| 4. Ввести 9 итерации и нажать кнопку «Запустить» | В окне строится «Снежинка Коха» |
| 5. Нажать кнопку Закрыть | Происходит выход из приложения |

Результаты, полученные в ходе тестирования, позволяют сделать заключение в том, что реализация программного продукта соответствует требованиям технического задания.

ЗАКЛЮЧЕНИЕ

В данной курсовой работе в соответствии с заданием разработано приложения для генерации фрактала «Снежинка Коха».

В ходе работы, были выполнены следующие задачи:

- анализ технического задания;

- анализ предметной области;

- выбор средств разработки;

- проектирование приложения;

- реализация приложения;

Приложение разработано на языке программирования C#, при разработке использовался Microsoft .NET Framework 4.8. Разработанная программа позволяет строить «Снежинку Коха» с различными итерациями:

Разработанный программный продукт может быть полезен для анализа построения «Снежинки Коха».

В курсовой работе есть недостаток, если задать большое количество итераций, операционная система может зависнуть, отменить действие нельзя, приходится завершать приложение принудительно.

Список литературы

1. C Sharp // URL: https://ru.wikipedia.org/wiki/C\_Sharp (дата обращения 17.01.2024);

2. Microsoft Visual Studio // URL: https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio (дата обращения 17.01.2024);

3. Давыдов В. Г. Программирование и основы алгоритмизации: Учеб. пособие.- М.: Высш. Шк., 2003.-447 с.:ил.

4. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М: Техносфера, 2005. – 1072 с.