Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №7**

**«ИССЛЕДОВАНИЕ ФРАКТАЛОВ»**

**ПО ДИСЦИПЛИНЕ**

**МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ**

Выполнил: студент учебной группы

ИСПк-205-52-00

Ильин Тимофей Анатольевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров 2024

Цель работы: получение навыков реализации алгоритмов с рекурсивными вычислениями, знакомство с фракталами.

**Задание (Вариант 7)**

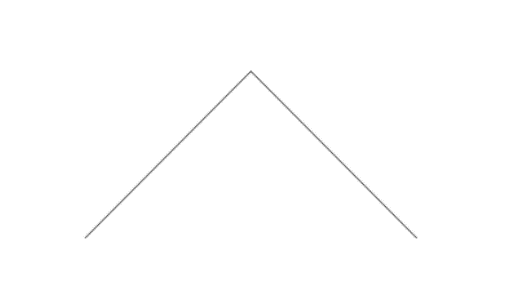
1. ﻿﻿﻿Написать программу для визуализации фрактала "Кривая Хартера-Хейтуэя"
2. ﻿﻿﻿Предусмотреть возможности масштабирования, изменения глубины прорисовки и перемещения полученной фигуры
3. ﻿﻿﻿Построение множества ломанных, образующих фрактал. должно осуществляться в отдельном модуле

**Описание алгоритма**

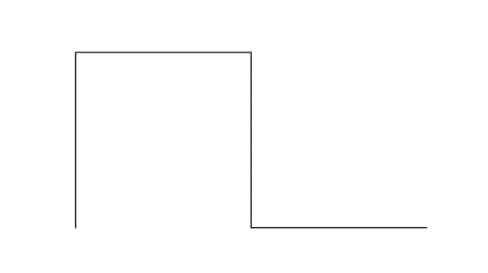
Алгоритм Unit A:

1. Кривая дракона - это одна из фрактальных кривых, которая получается с помощью ломанной-генератора. Построение кривой дракона начинается с прямой линии, которая затем разделяется пополам и каждая половина поворачивается на 90 градусов вправо или влево. Затем линии соединяются, и процесс повторяется для каждой новой линии, создавая все более сложную кривую.
2. Ломанная-генератор:

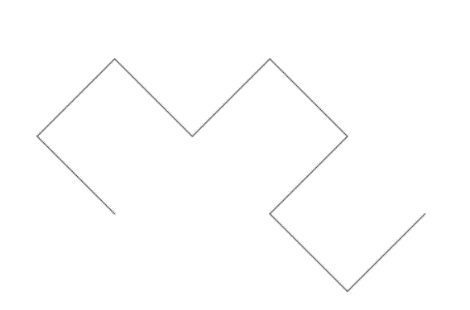
Глубина 1



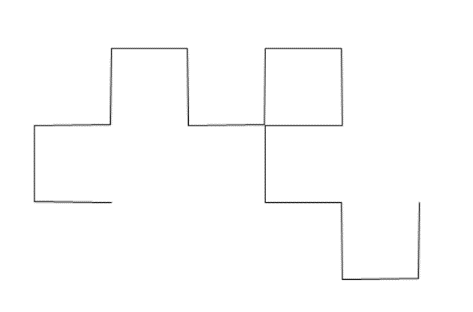
Глубина 2



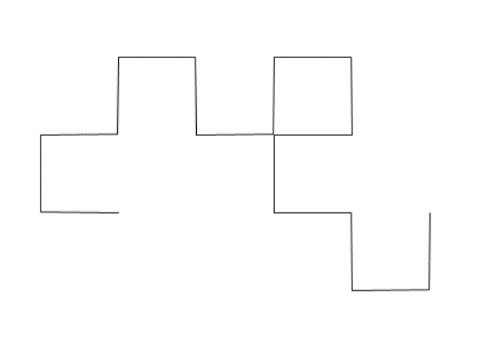
Глубина 3



Глубина 4



Глубина 5



1. Алгоритм построения фрактала кривой дракона может быть следующим:

* Установить начальные точки A и B.
* Установить глубину рекурсии, которая определяет количество шагов построения фрактала.
* Если глубина равна 0, то нарисовать линию AB и завершить.
* Разделить отрезок AB пополам и сохранить новые точки C и D.
* Повернуть отрезок CD на 90 градусов вправо или влево (выбор направления может быть случайным или определенным).
* Соединить точки B и C.
* Рекурсивно вывать алгоритм для новых отрезков BC и CD с глубиной уменьшенной на 1.

**Схема алгоритма с комментариями**

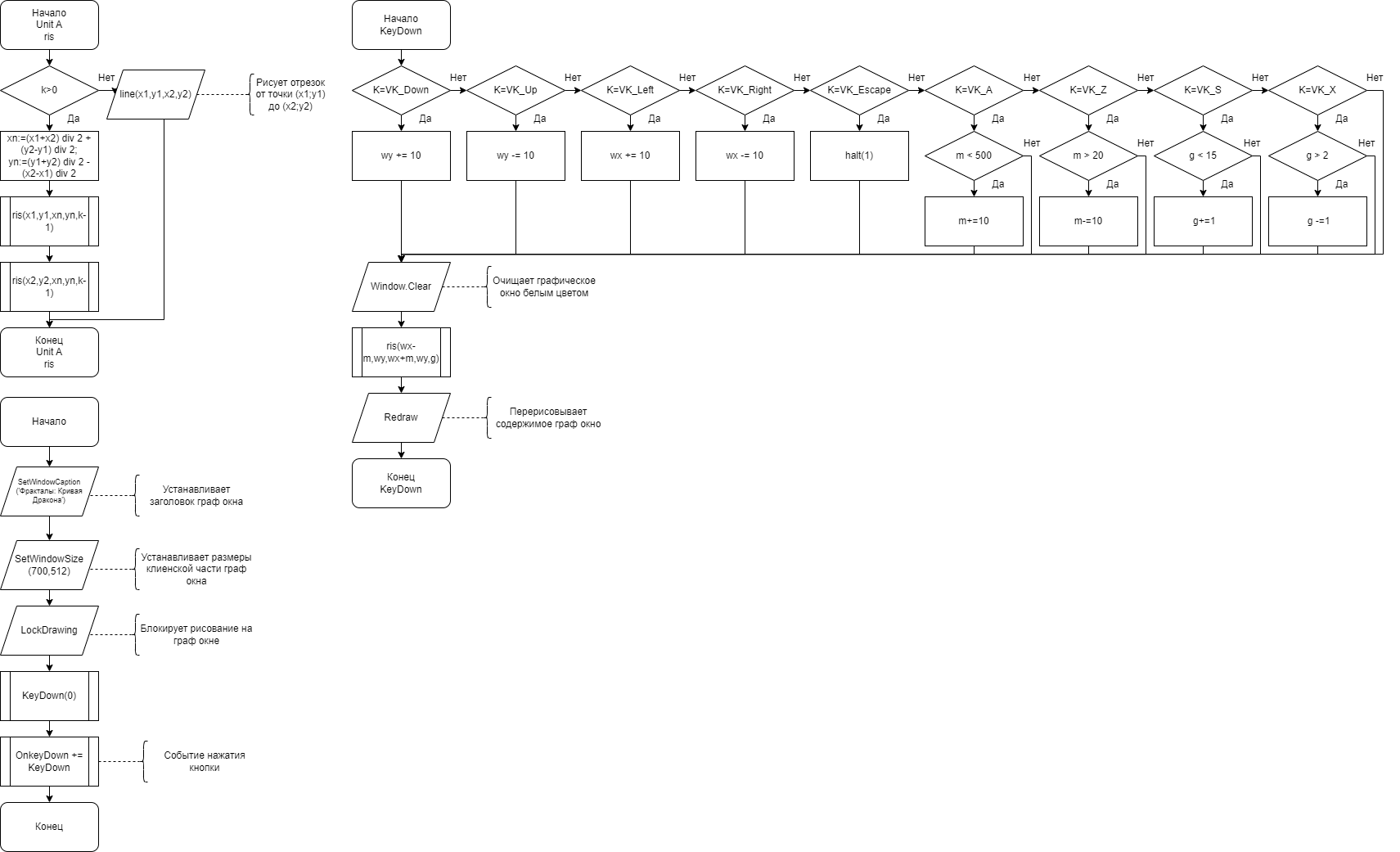
****

Рис.1 Схема алгоритма Unit A procedure ris.

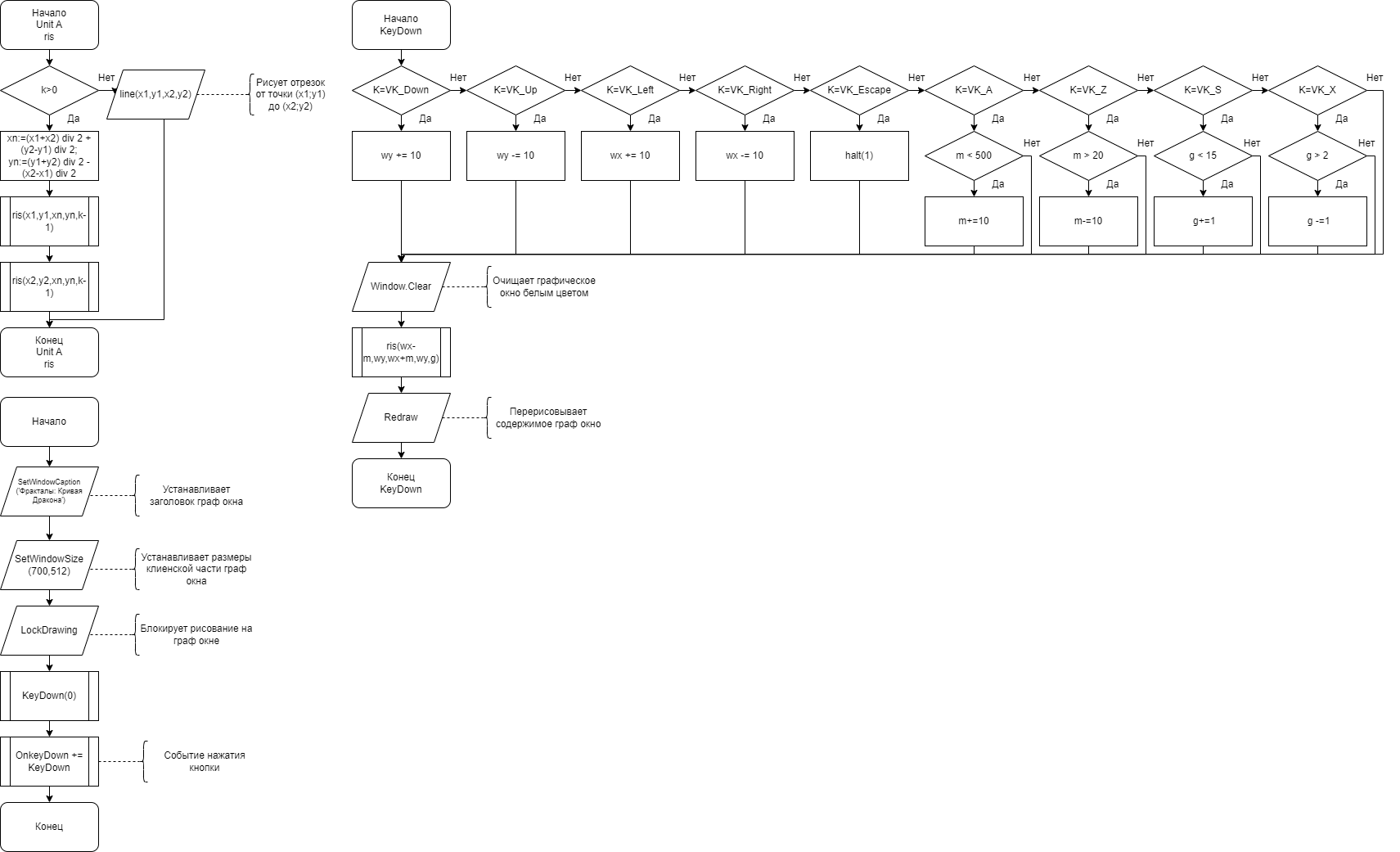
****

Рис.2 Схема алгоритма procedure KeyDown.

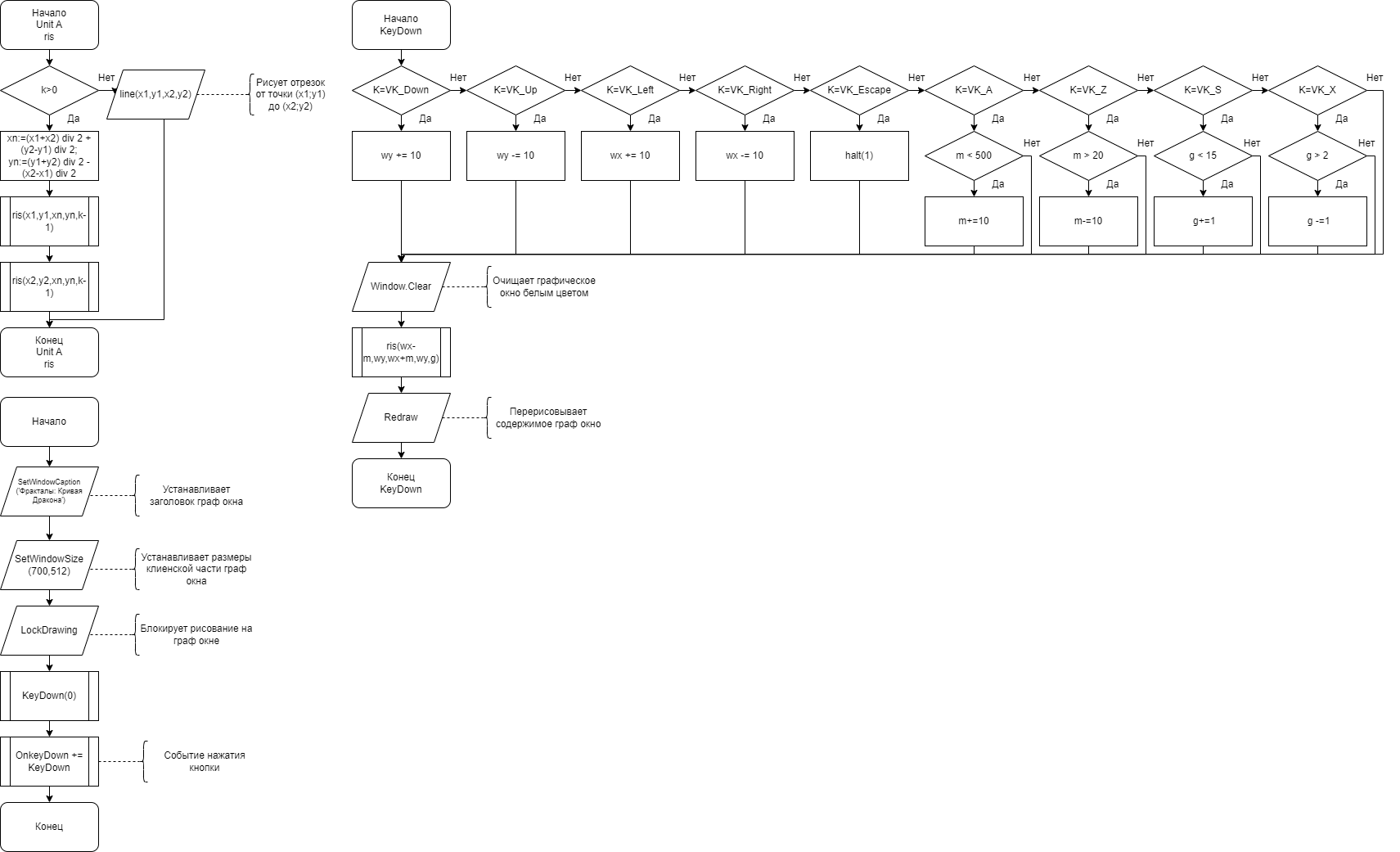


Рис.3 Схема алгоритма.

**Код модуля Unit A**

**Unit** A;

**Uses** GraphABC;

**Procedure** ris(x1,y1,x2,y2,k:integer);

**begin**

**If** k>0 **then begin**

**var** xn:=(x1+x2) **div** 2 +(y2-y1) **div** 2;

**var** yn:=(y1+y2) **div** 2 -(x2-x1) **div** 2;

ris(x1,y1,xn,yn,k-1);

ris(x2,y2,xn,yn,k-1);

**end**

**else** line(x1,y1,x2,y2)

**end**;

**end**.

**Код основной программы**

**Uses** GraphABC, A;

**var** (m,wx,wy,g) := (100, 300,300,0);

**Procedure** KeyDown(k: integer);

**begin**

**case** K **of**

VK\_Down: wy += 10;

VK\_Up: wy -= 10;

VK\_Left: wx -= 10;

VK\_Right: wx += 10;

VK\_Escape: halt(1); //выход

VK\_A: **if** m < 500 **then** m+=10; // + маштаб

VK\_Z: **if** m > 20 **then** m-=10; // - маштаб

VK\_S: **if** g < 15 **then** g+=1; // + глубина

VK\_X: **if** g > 0 **then** g -=1; // - глубина

**end**;

Window.Clear;

ris(wx-m,wy,wx+m,wy,g);

Redraw;

**end**;

**Begin**

SetWindowCaption('Фракталы: Кривая Дракона');

SetWindowSize(700,512);

LockDrawing;

KeyDown(0);

OnkeyDown += KeyDown;

**End**.

**Результат выполнения программы**

1. Глубина 1:

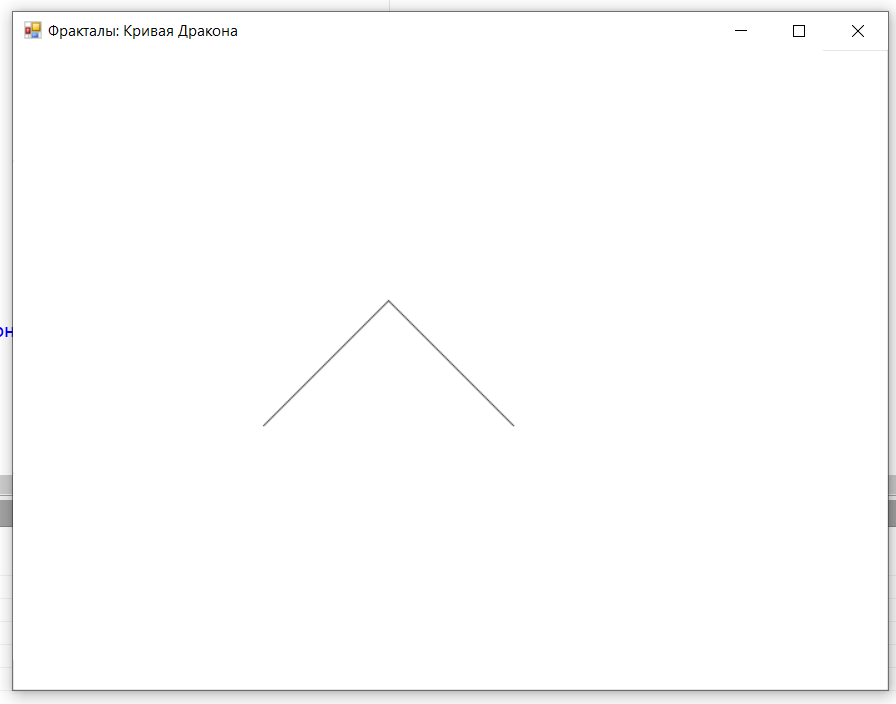


Рис. 4 Глубина 1.

1. Глубина 2:

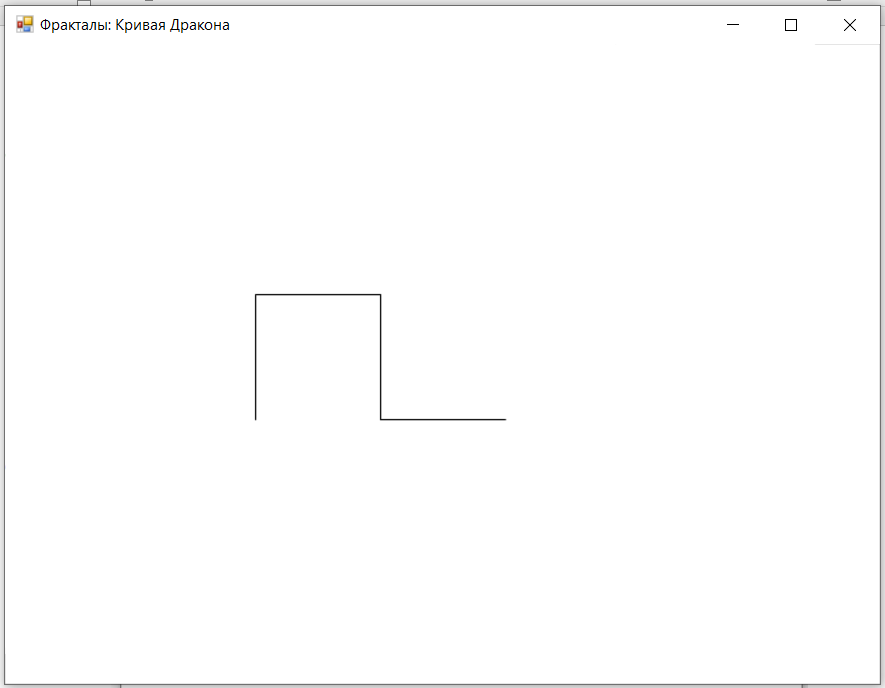


Рис. 5 Глубина 2.

1. Глубина 3:

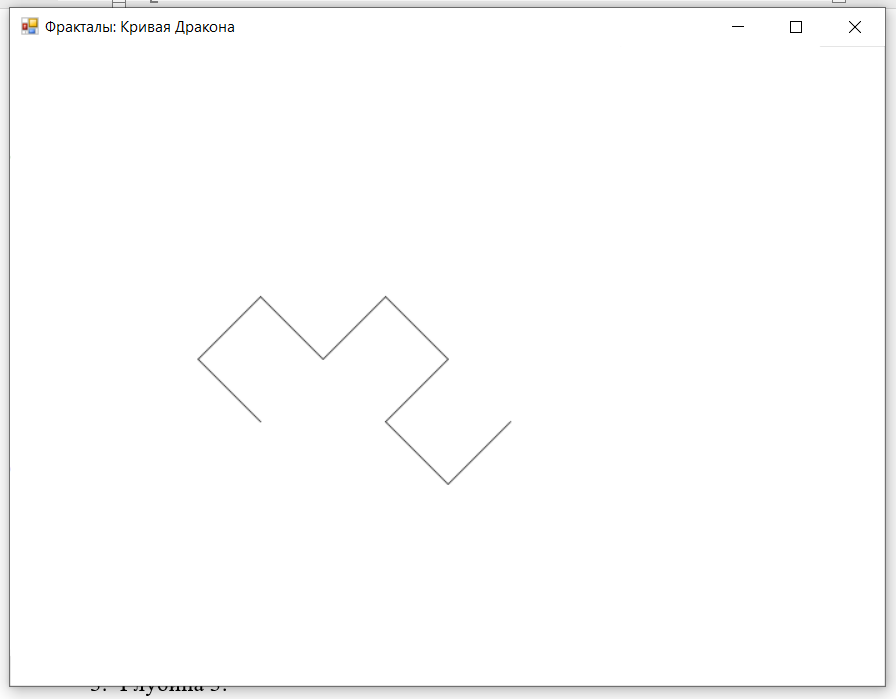


Рис. 6 Глубина 3.

1. Изменение масштаба (+):

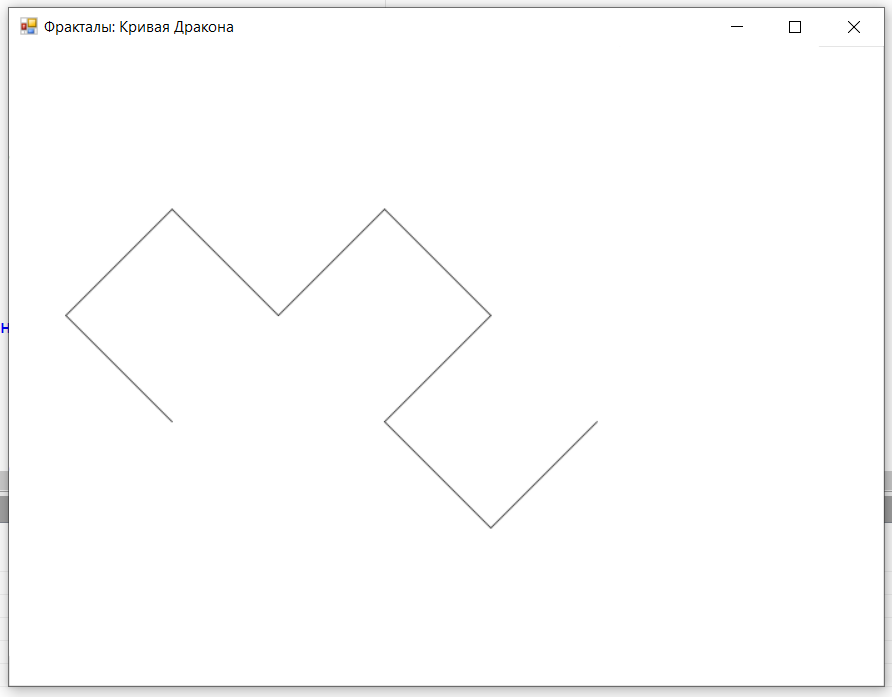


Рис. 7 Изменение масштаба (+)

1. Изменение масштаба (-):

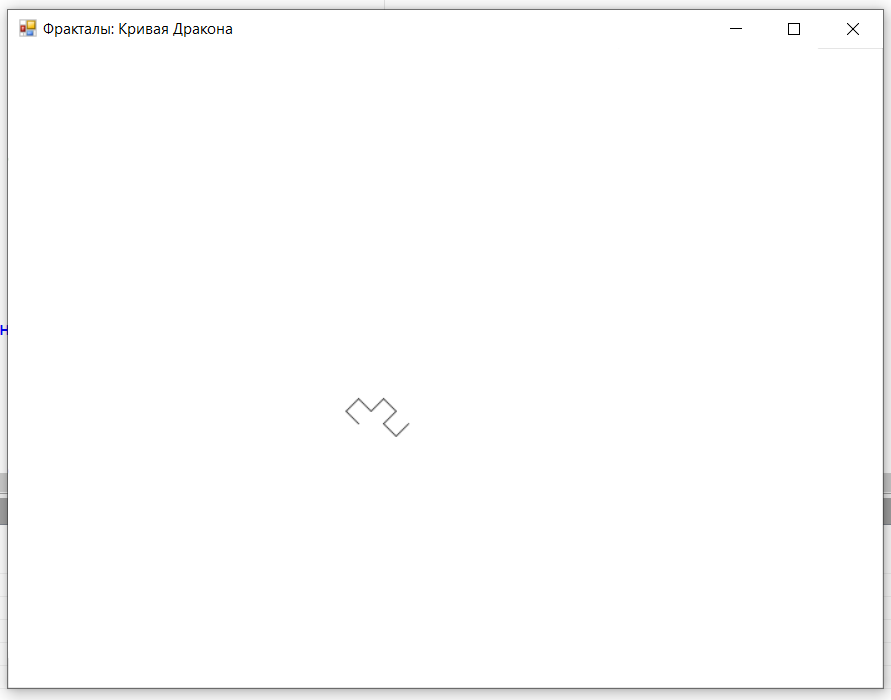


Рис. 8 Изменение масштаба (-)

1. Перемещение вправо:

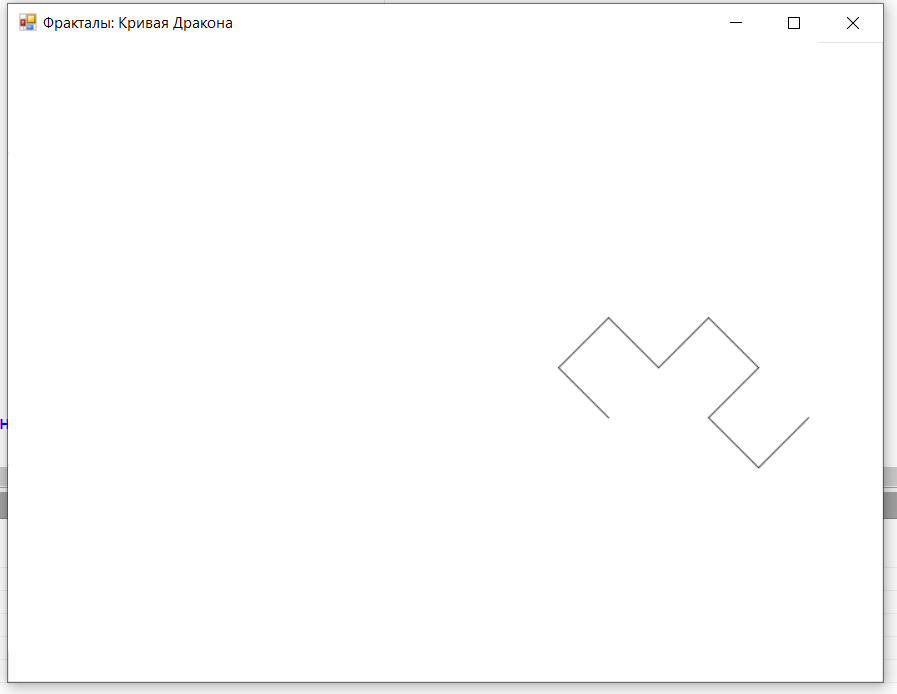


Рис. 8 Перемещение вправо

1. Перемещение влево:

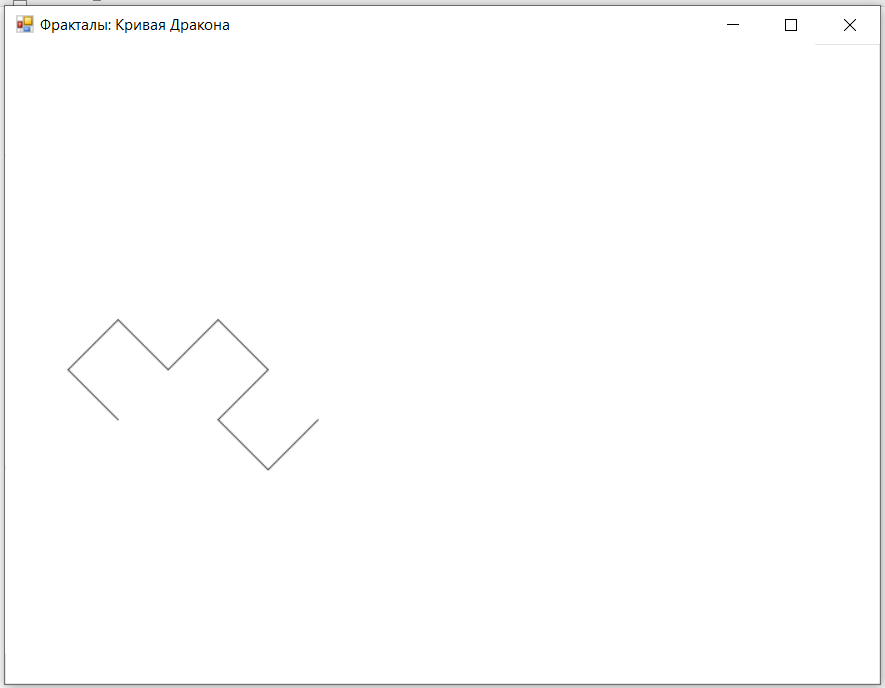


Рис. 9 Перемещение влево

1. Перемещение вверх:

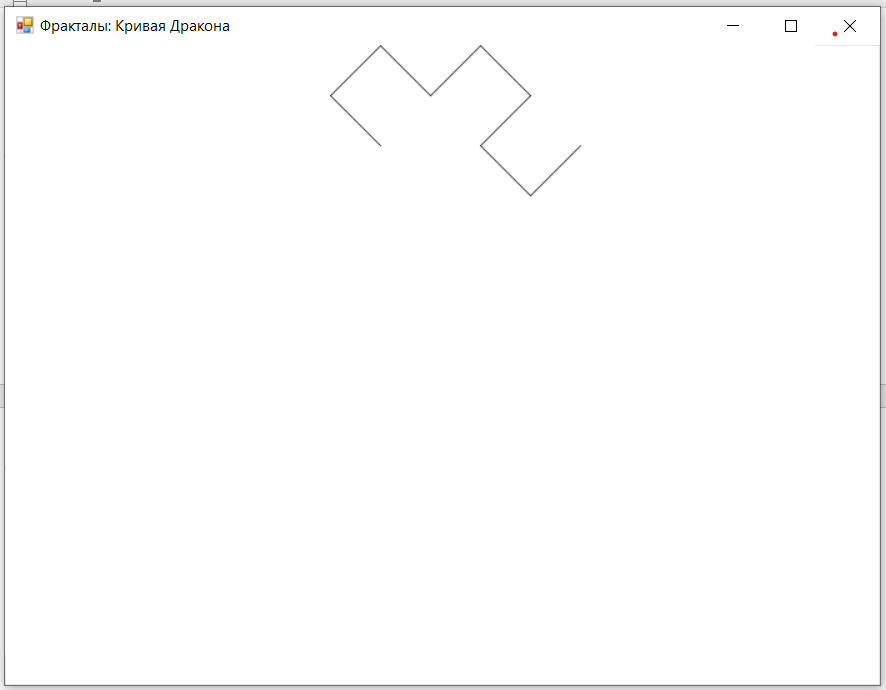


Рис. 10 Перемещение вверх

1. Перемещение вниз:

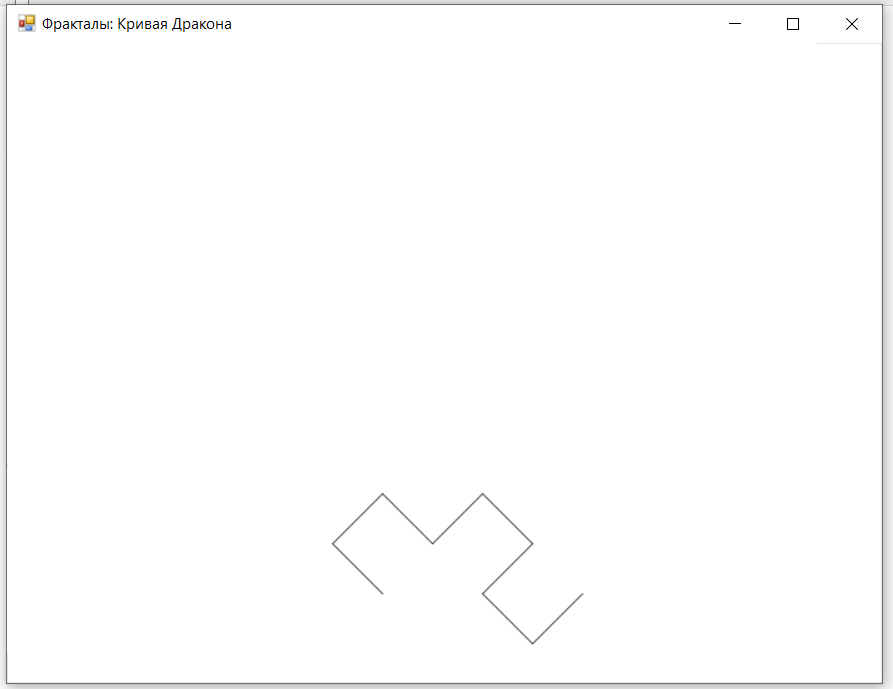


Рис. 11 Перемещение вниз

Вывод: в ходе работы были приобретены навыки по реализации алгоритмов с рекурсивными вычислениями и изучены основы работы с фракталами. Эти навыки могут быть полезны в области программирования, математики и искусства. Полученные знания помогут в дальнейшем развитии и применении умения работы с рекурсией и создании разнообразных фрактальных структур.

Так же был изучен Unit, его использование в коде позволяет более эффективно тестировать программы и обнаруживать ошибки, что способствует более надежному функционированию разработанных алгоритмов. Полученные знания и навыки также будут полезны при работе над проектами в области компьютерной графики, анимации, компьютерного моделирования и других областях, где требуется умение создавать сложные графические структуры с использованием рекурсивных алгоритмов и фрактальных образований.