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»**

**«МДК 05.02 Разработка кода информационных систем»**

Выполнил: студент учебной группы

ИСПк-202-52-00

Рыков Ефим Витальевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

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

Задание:

1. Написать программу для визуализации фракталов «Кривая Хартера-Хейтуэя»
2. Предусмотреть возможность масштабирования, изменения глубины прорисовки и перемещения полученной фигуры
3. Построение множества ломанных, образующих фрактал, должно осуществляется в отдельном модуле.

Описание алгоритма

Для его построения возьмем отрезок. Повернем его на 90 градусов вокруг одной из вершин и добавим полученный отрезок к исходному. Получим уголок из двух отрезков. Повторим описанную процедуру. Повернем уголок на 90 градусов вокруг вершины и добавим полученную ломаную к исходной.

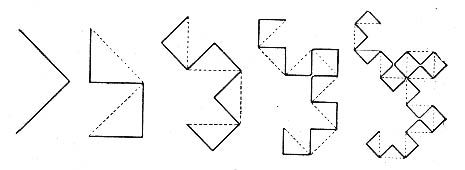
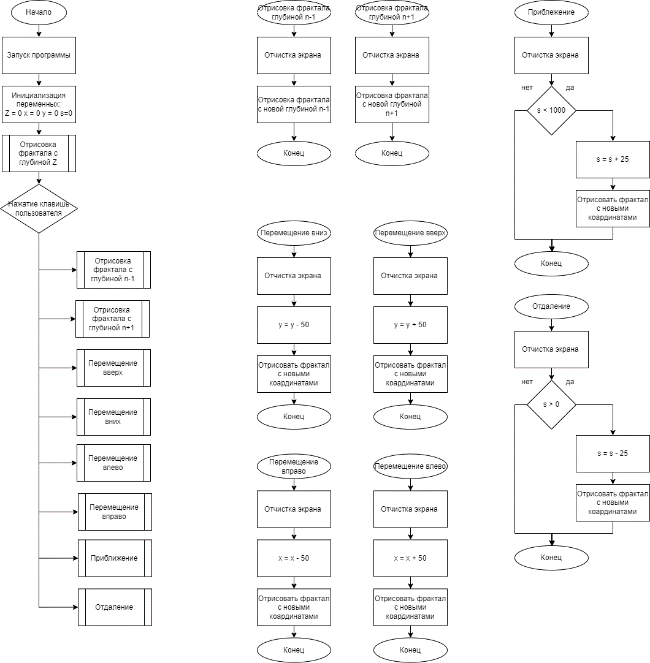


Схема алгоритма с комментариями:



VK\_G

VK\_F

VK\_S

VK\_D

VK\_Rigth

VK\_Left

VK\_Down

VK\_UP

Конец

Код программы

**Uses** GraphABC, draw\_module;

**var**

Z: integer; {Glubina Fraktala}

x,y, s:integer;

**procedure** text;

**begin**

textout(10, 50, Z);

textout(10, 100, 'Left - влево');

textout(10, 120, 'Right - вправо');

textout(10, 140, 'Up - вверх');

textout(10, 160, 'Down - вниз');

textout(10, 180, 'd - crop-in');

textout(10, 200, 's - crop-out');

textout(10, 220, 'g - depth +');

textout(10, 240, 'f - depth -');

**end**;

**procedure** KeyDown(key: integer);//Движ

**begin**

**case** key **of**

VK\_Up: **begin** y := y + 50 **end**;

VK\_Down: **begin** y := y - 50 **end**;

VK\_Left: **begin** x := x + 50 **end**;

VK\_Right: **begin** x := x - 50 **end**;

VK\_S: **begin if** s>0 **then** s:= s-25 **end**;

VK\_D: **begin if** s<1000 **then** s:= s+25 **end**;

VK\_G:

**begin**

Z:= Z+1;

**end**;

VK\_F:

**begin**

**if** z > 0 **then**

Z:= Z-1;

**end**;

**end**;

Window.Clear;

text;

draw(200+x-s,300+y,500+x+s,300+y,Z);

redraw;

**end**;

**Begin**

SetWindowCaption('Фракталы: Кривая Дракона');

SetWindowSize(700,512);

ClearWindow;

LockDrawing;

Z:=1;

x:=0;

y:=0;

draw(200+x,300+y,500+x,300+y,Z);

text;

redraw;

onKeyDown += keydown;

**End**.

**Unit** draw\_module;

**Uses** GraphABC;

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

**Var** xn,yn:integer;

**Begin**

**If** k>0 **Then**

**Begin**

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

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

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

draw(x2,y2,xn,yn,k-1)

**End**

**Else**

line(x1,y1,x2,y2)

**End**;

**begin**

**end**.

Результат выполнения программы

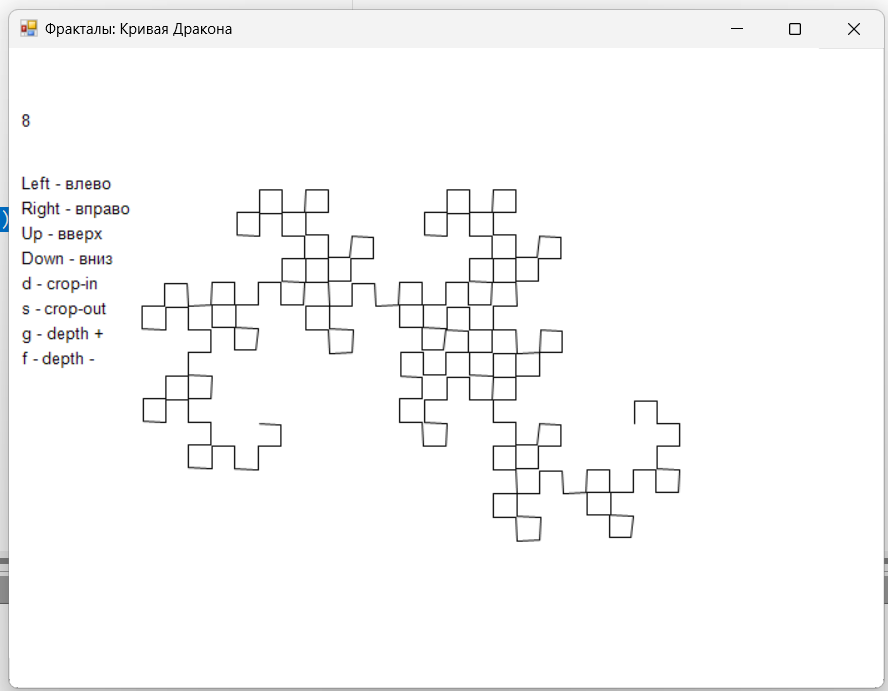


Рисунок 1 - работа программы

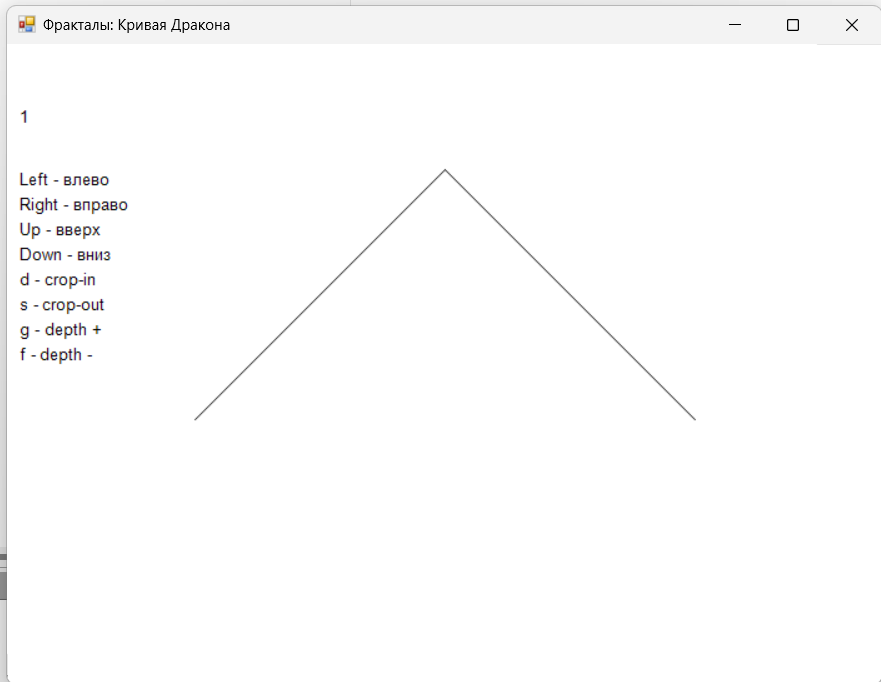


Рисунок 2 - уменьшение глубины

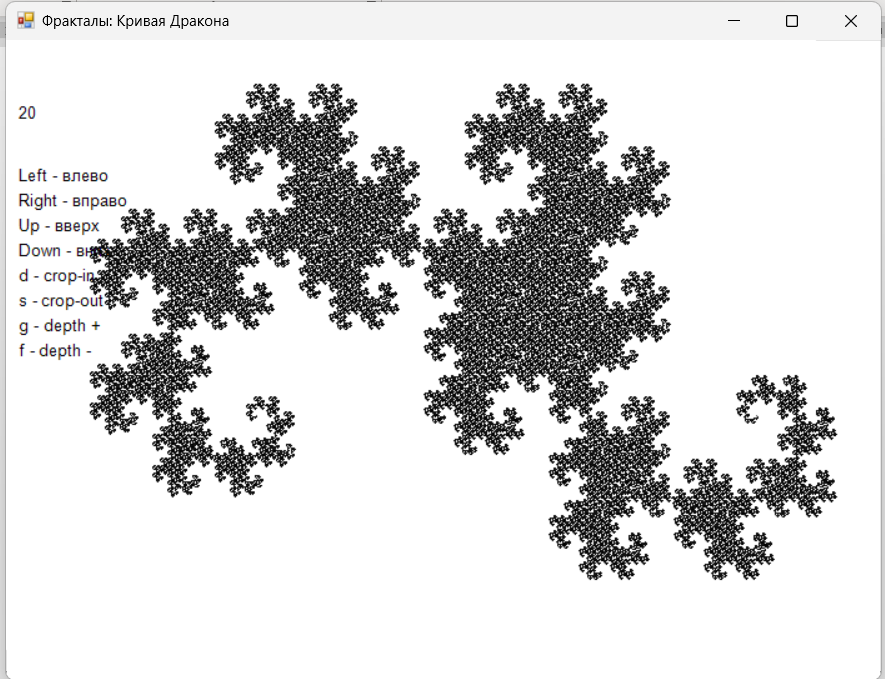


Рисунок 3 - увеличение глубины

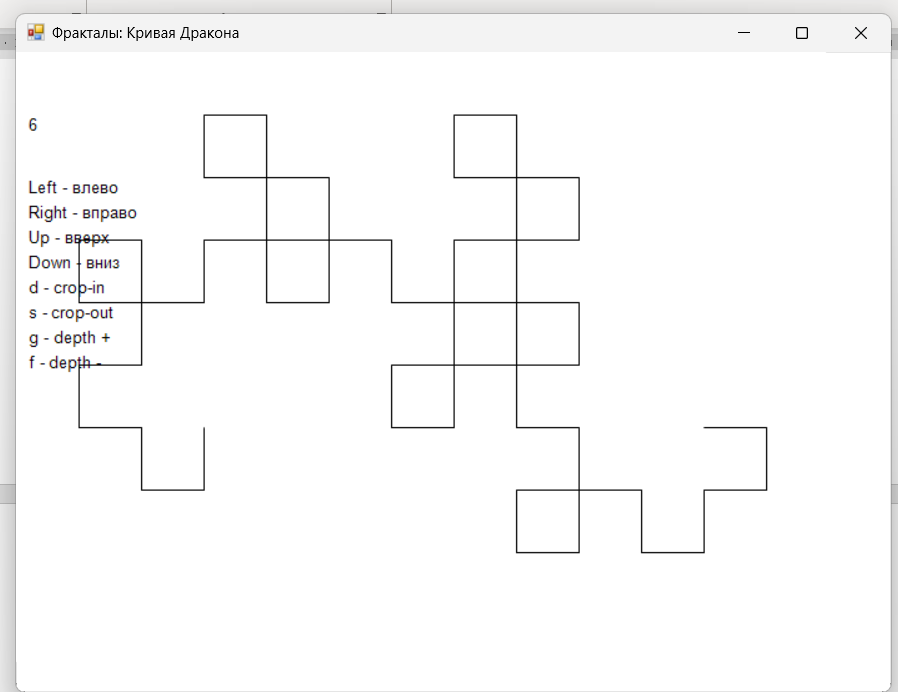


Рисунок 4 - перемещение влево

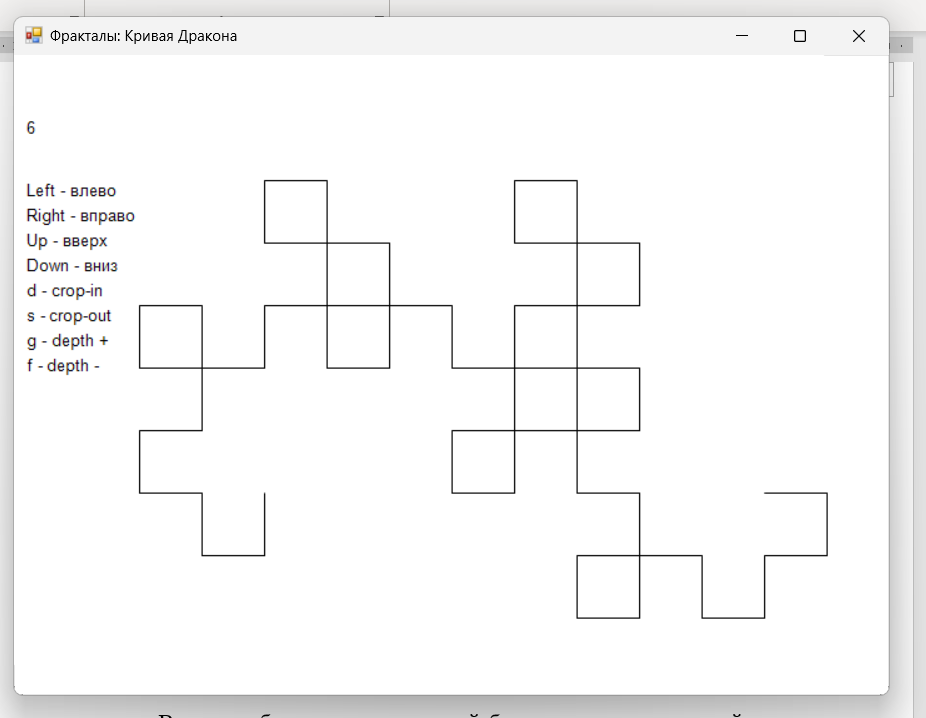


Рисунок 5 - перемещение вверх

Вывод

В ходе роботы над программой был изучен рекурсивный алгоритм построения фракталов: Дракон. При реализации, алгоритм был вынесен в отдельный модуль, с чем возникли сложности, но в конечном итоге алгоритм вызывается из отдельного модуля. Рекурсивный алгоритм требует декомпозиции, в чём, по началу, были логические ошибки, и выделение логики подсчёта координат, которая требовала дополнительных тестирований для корректной работы. Отдельное вынесение кнопок управления для фрактала и их последующая инструкция, которая выводится на экран, то же не обошлась без тестирования прописывания логических частей для корректного управления программой.