Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №1**

**«Фракталы»**

**ПО МДК 05 02 «Разработка кода информационных систем»**

Выполнил: студент учебной группы

ИСПк-202-52-00

Бабенко Александр Александрович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

1.Цель работы:

Получение навыков реализации алгоритмов с рекурсивными вычислениями, знакомство с фракталами

2.Задание:

1.) Написать программу для визуализации фрактала “Кривая Минковского”

2.) Предусмотреть возможность масштабирования, изменение глубины прорисовки, перемещения полученной фигуры

3.) Построение множества ломанных, образующих фрактал, должно осуществляться в отдельном модуле.

3.Описание алгоритма

Исходная кривая Минковского представляет собой ломаную линию, состоящую из повторяющихся шагов. Каждый шаг включает в себя разделение отрезка на три равные части и замену средней части отрезка равносторонним треугольником. Этот процесс повторяется для каждого нового отрезка, что приводит к увеличению числа сегментов и более сложной форме кривой.

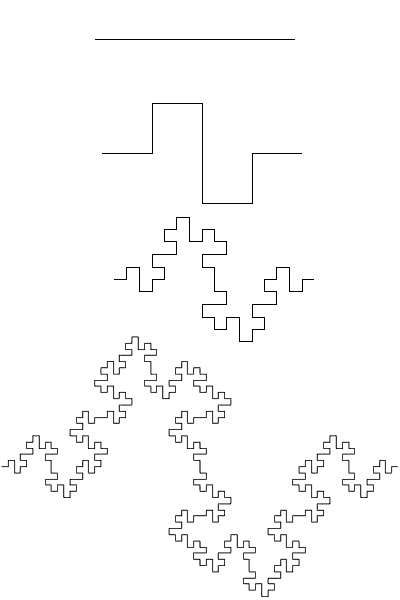


Рисунок 1 – Кривая Минковского в зависимости от глубины

4. Схема алгоритма с комментариями

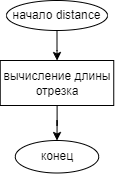
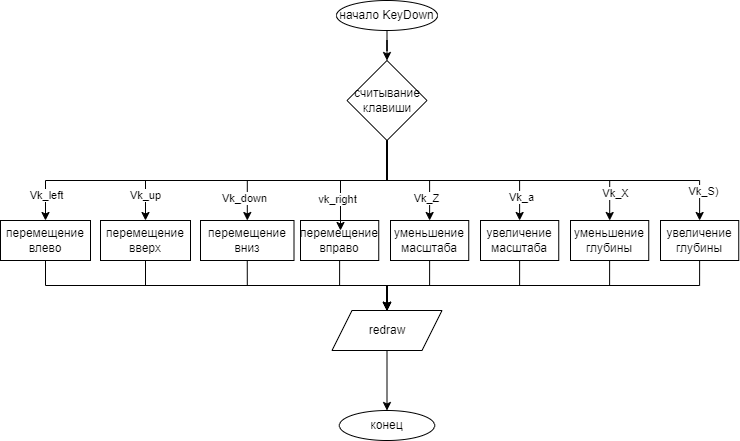


Рисунок 2 – схема функции distance



Рисунок 3 – схема функции GetAngle

Рисунок 4 – процедура keydown

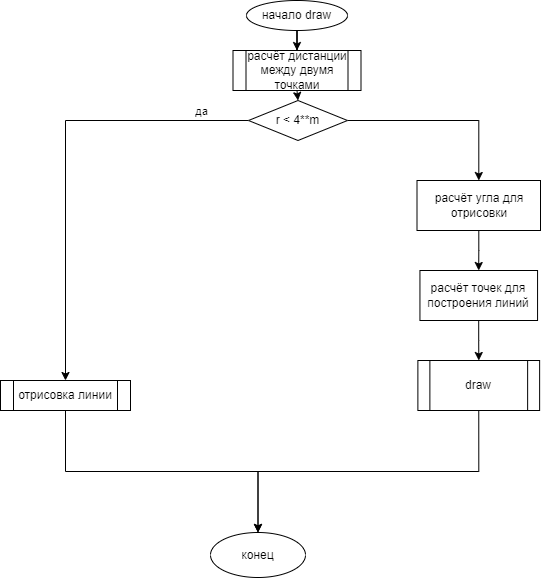


Рисунок 5 – процедура draw

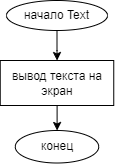


Рисунок 6 – процедура Text

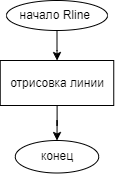


Рисунок 7 - процедура Rline

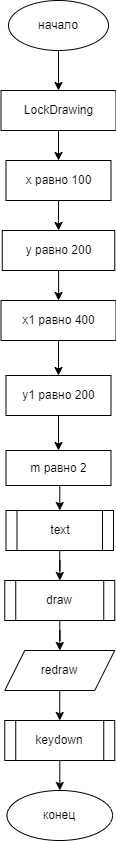


Рисунок 8 – основная программа

5.Код программы

**uses** GraphABC, drawunit;

**var** m: integer;

**procedure** text;

**begin**

textout(10, 100, 'Left - влево');

textout(10, 120, 'Right - вправо');

textout(10, 140, 'Up - вверх');

textout(10, 160, 'Down - вниз');

textout(10, 180, 'z - уменьшение');

textout(10, 200, 'a - увеличение');

textout(10, 220, 'x - уменьшение глубины');

textout(10, 240, 's - увеличение глубины');

textout(10,10, abs(5-m))

**end**;

**var** x, y, x1, y1, k: integer;

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

**begin**

**case** key **of**

VK\_Up: **begin** y := y + 5; y1 := y1 + 5 **end**;

VK\_Down: **begin** y -= 5; y1 -= 5 **end**;

VK\_Left: **begin** x := x + 5; x1 := x1 + 5 **end**;

VK\_Right: **begin** x := x - 5; x1 := x1 - 5 **end**;

VK\_A: x := x - 50;

Vk\_Z: x := x + 50;

vk\_s: **if** m>-40 **then** m -= 1;

vk\_x:**if** m<4 **then** m += 1;

vk\_F:

**end**;

Window.Clear;

text;

draw(x, y, x1, y1,m);

redraw;

**end**;

**begin**

LockDrawing;

x := 100;

y := 200;

x1 := 400;

y1 := 200;

m:= 2;

text;

draw(x, y, x1, y1,m);

redraw;

onKeyDown := keydown;

**end**.

Код подпрограммы unit

**unit** drawunit;

**uses** GraphABC;

**procedure** RLine(x, y, x1, y1: real) := Line(Round(x), Round(y), Round(x1), Round(y1));

**function** GetAngle(x, y, x2, y2: real): real;

**begin**

**var** angle := Abs(RadToDeg(ArcTan((y2 - y) / (x2 - x))));

**if** (x2 = x) **and** (y2 = y) **then**

Result := 0

**else**

**if** x2 > x **then**

**if** y2 > y **then** Result := angle **else** Result := 360 - angle

**else**

**if** y2 > y **then** Result := 180 - angle **else** Result := 180 + angle;

**end**;

**function** Distance(x, y, x1, y1: real) := Sqrt(Sqr(x1 - x) + Sqr(y1 - y));

**procedure** Draw(x, y, x1, y1: real; m: integer);

**begin**

**var** r := Distance(x, y, x1, y1);

**if** r < 4\*\*m **then**

RLine(x, y, x1, y1)

**else**

**begin**

**var** angle := GetAngle(x, y, x1, y1);

**var** angleP := DegToRad(angle + 90);

**var** angleM := DegToRad(angle - 90);

r /= 4;

**var** dx := (x1 - x) / 4;

**var** dy := (y1 - y) / 4;

**var** xA := x + dx;

**var** yA := y + dy;

**var** xB := xA + dx;

**var** yB := yA + dy;

**var** xC := xB + dx;

**var** yC := yB + dy;

**var** x2 := xA + r \* Cos(angleP);

**var** y2 := yA + r \* Sin(angleP);

**var** x3 := xB + r \* Cos(angleP);

**var** y3 := yB + r \* Sin(angleP);

**var** x4 := xB + r \* Cos(angleM);

**var** y4 := yB + r \* Sin(angleM);

**var** x5 := xC + r \* Cos(angleM);

**var** y5 := yC + r \* Sin(angleM);

Draw(x, y, xA, yA,m);

Draw(xA, yA, x2, y2,m);

Draw(x2, y2, x3, y3,m);

Draw(x3, y3, xB, yB,m);

Draw(xB, yB, x4, y4,m);

Draw(x4, y4, x5, y5,m);

Draw(x5, y5, xC, yC,m);

Draw(xC, yC, x1, y1,m);

**end**;

**end**;

**begin**

**end**.

Результат выполнения программы:

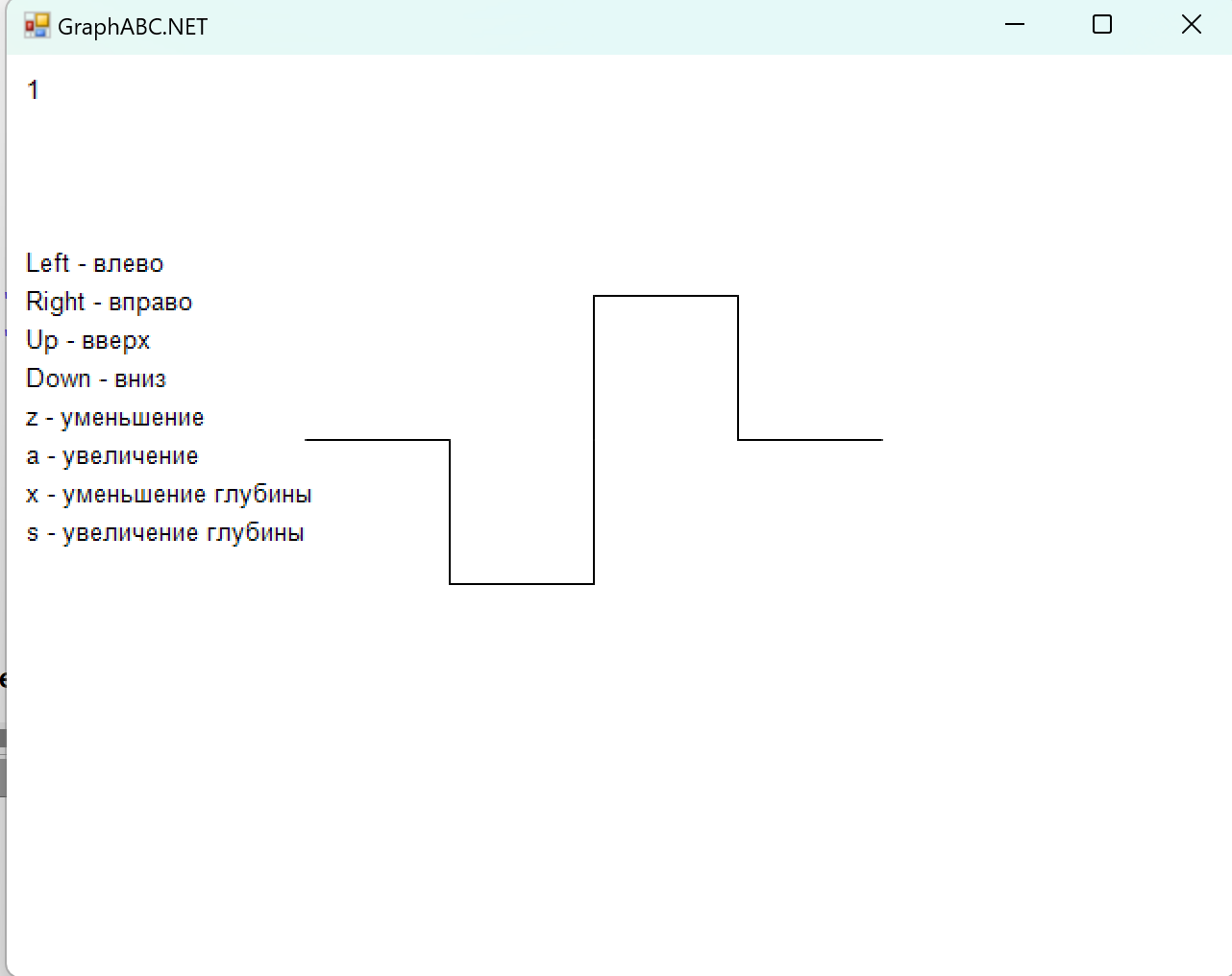


Рисунок 9 – результат выполнения программы по глубине 1

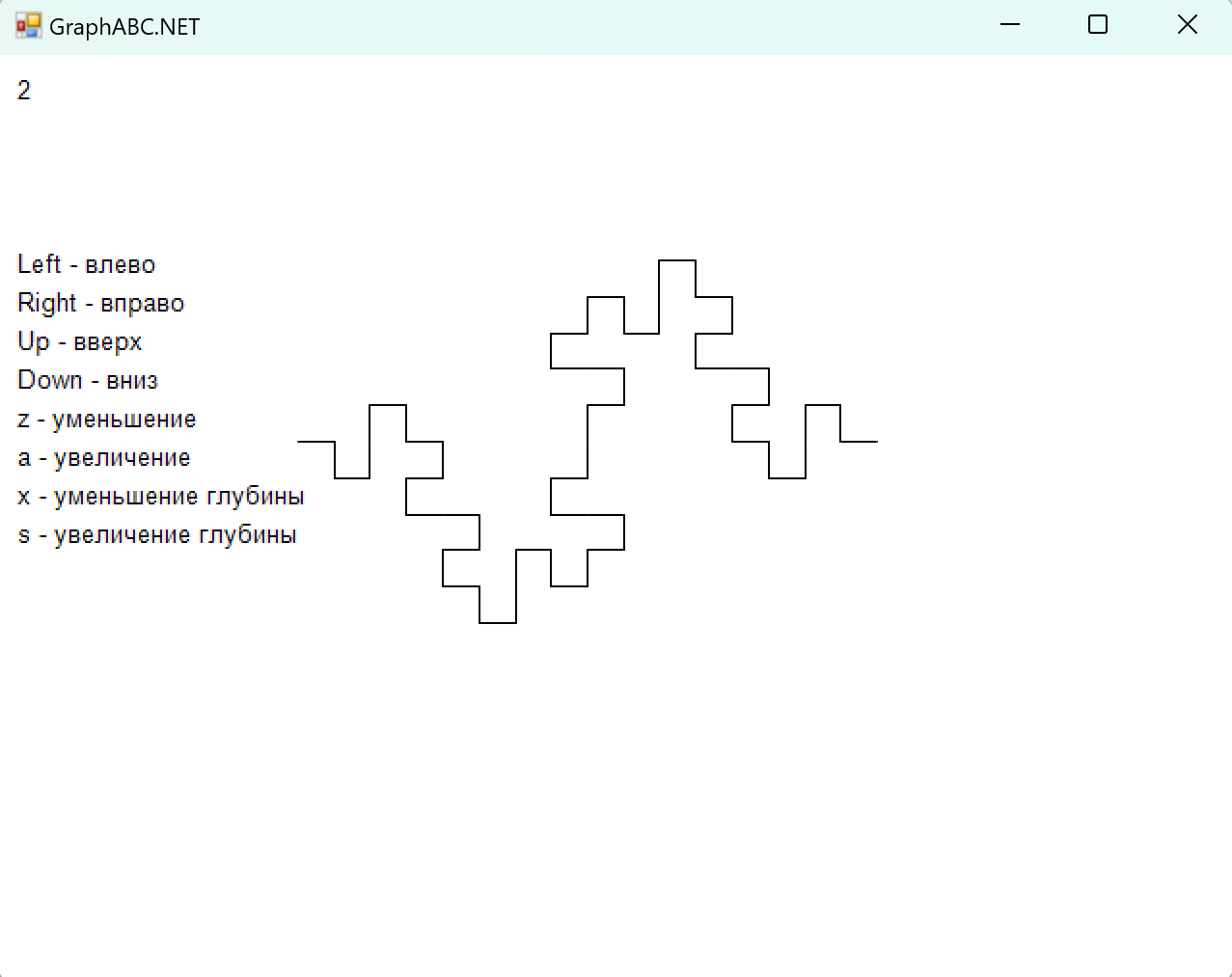


Рисунок 10 – Результат выполнения программы по глубине 2

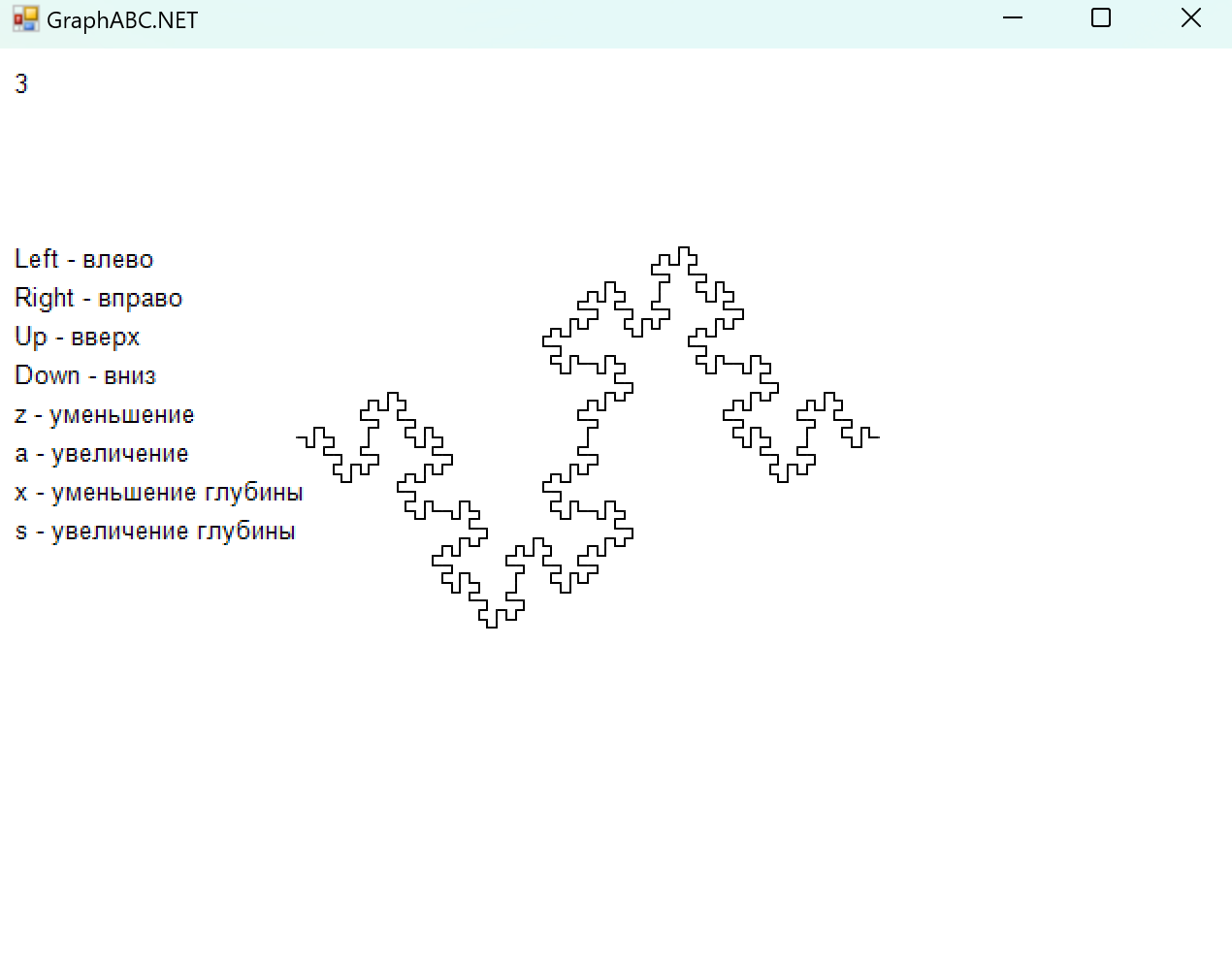


Рисунок 11 – результат выполнения программы по глубине 3

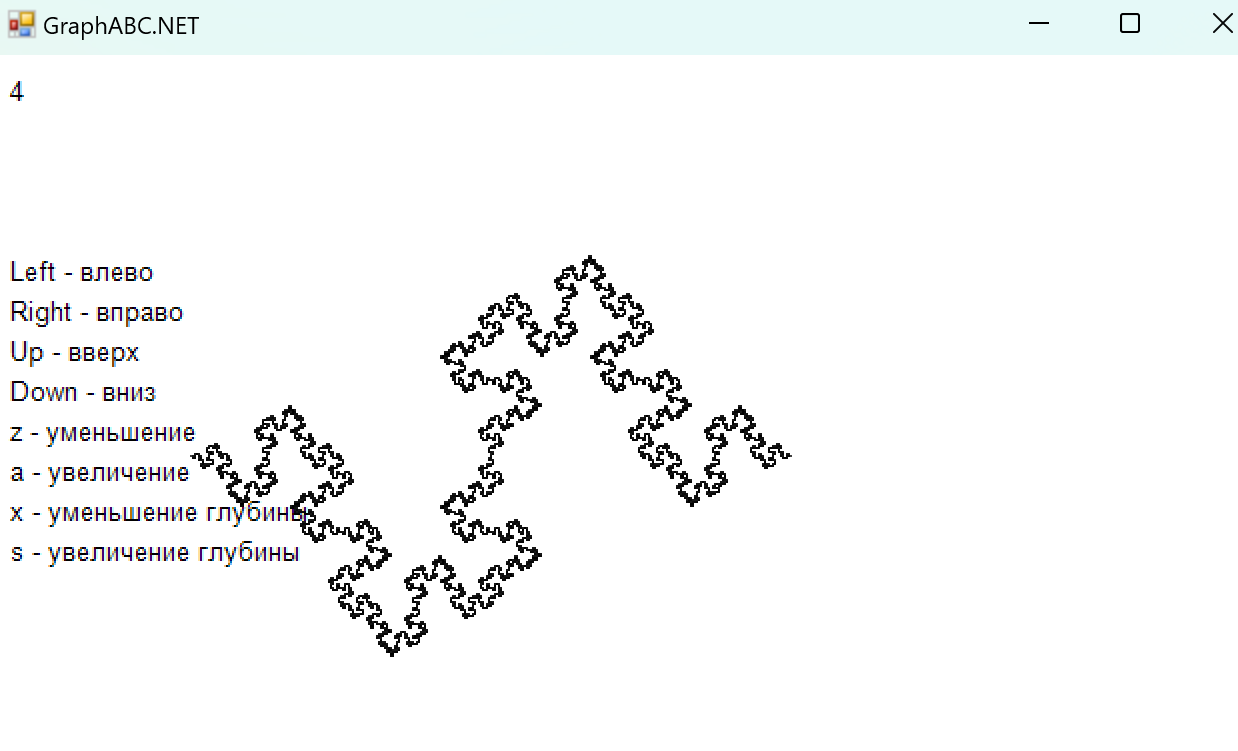


Рисунок 12 – результат выполнения программы по глубине 4

Вывод: По итогам данной работы я получил навыки по работе с рекурсивными алгоритмами и реализовал программу по отрисовке фрактала кривая Минковского. В течении работы я столкнулся с такими проблемами, как медленная скорость отрисовки фрактала разной глубины, а также перенос подпрограмм из основной программы в отдельный юнит.