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 МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ**

Выполнил: студент учебной группы

ИСПк-205-52-00

Порошин Тимофей

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

В отчете должны отображаться:

* Цель работы
* Скриншот задания (с вариантом)
* Описание алгоритма
* Код программы
* Результат выполнения программы
* Вывод

**Цель работы:** получение навыков реализации алгоритмов с рекурсивным вычислениями, знакомств с фракталами.

**Скриншот задания:**

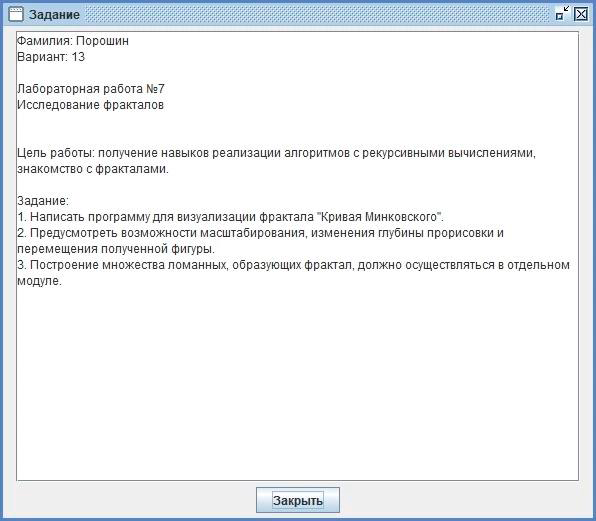


Рисунок 1 – Задание

**Описание алгоритма:** В данном коде используются 5 функции, 4 из которых являются основными, которые рисуют кривую Минковского. Они рисуют линии, переворачивают рисунок в право и лево, образуют кривые. Вот пример одной из функции: RLine (Функция для образования линии по переменным) например: Line (round(x), round(y), round(x1), round(y1)).

Последняя функция создана для того, чтобы редактировать масштаб, изменять глубину рисовки и перемещать рисунок.

**Код программы:**

**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));

**var** m: integer;

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

**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);

Draw(xA, yA, x2, y2);

Draw(x2, y2, x3, y3);

Draw(x3, y3, xB, yB);

Draw(xB, yB, x4, y4);

Draw(x4, y4, x5, y5);

Draw(x5, y5, xC, yC);

Draw(xC, yC, x1, y1);

**end**;

**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>0 **then** m -= 1;

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

vk\_F:

**end**;

Window.Clear;

draw(x, y, x1, y1);

redraw;

**end**;

**begin**

LockDrawing;

x := 100;

y := 200;

x1 := 400;

y1 := 200;

m:= 2;

draw(x, y, x1, y1);

redraw;

onKeyDown += keydown;

**end**.

**Результат выполнения программы:**

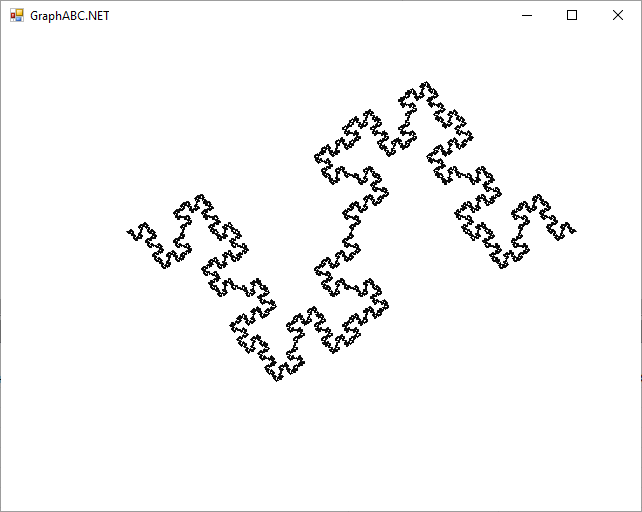


Рисунок 2 – Результат

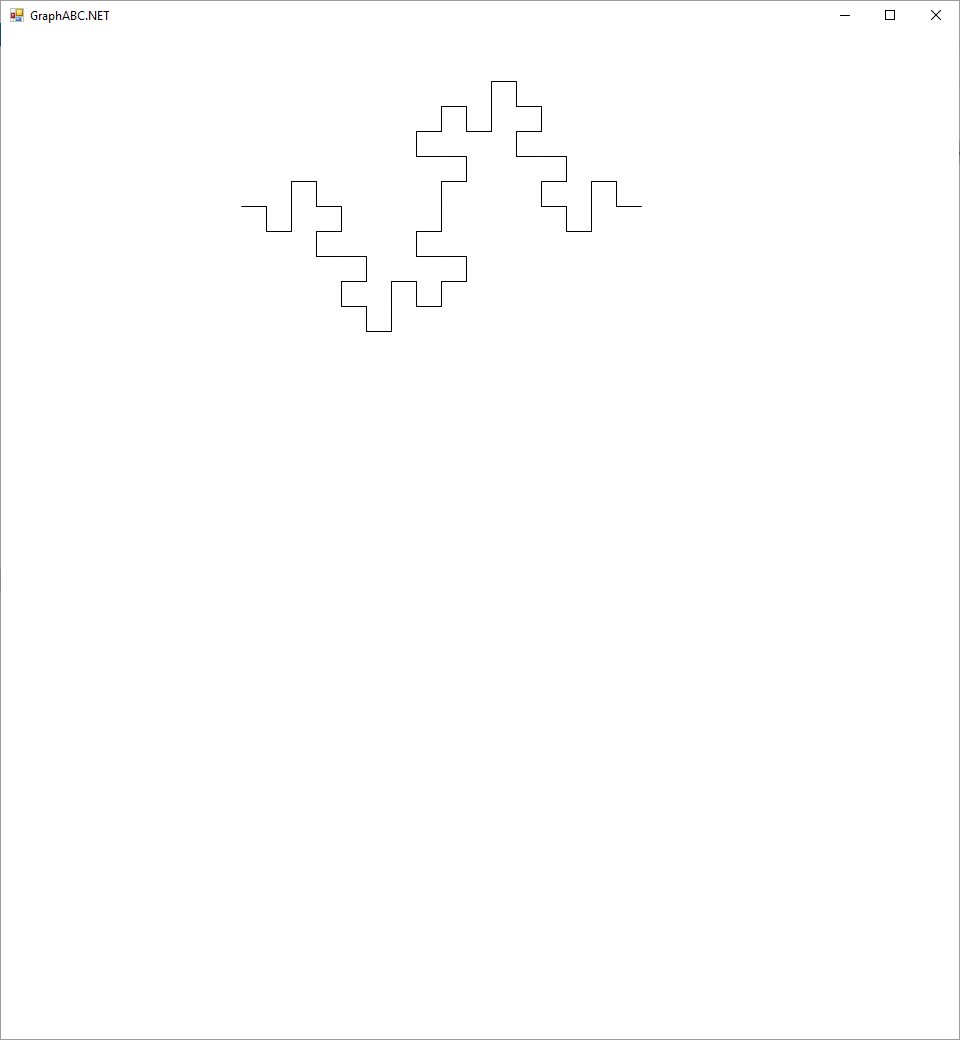
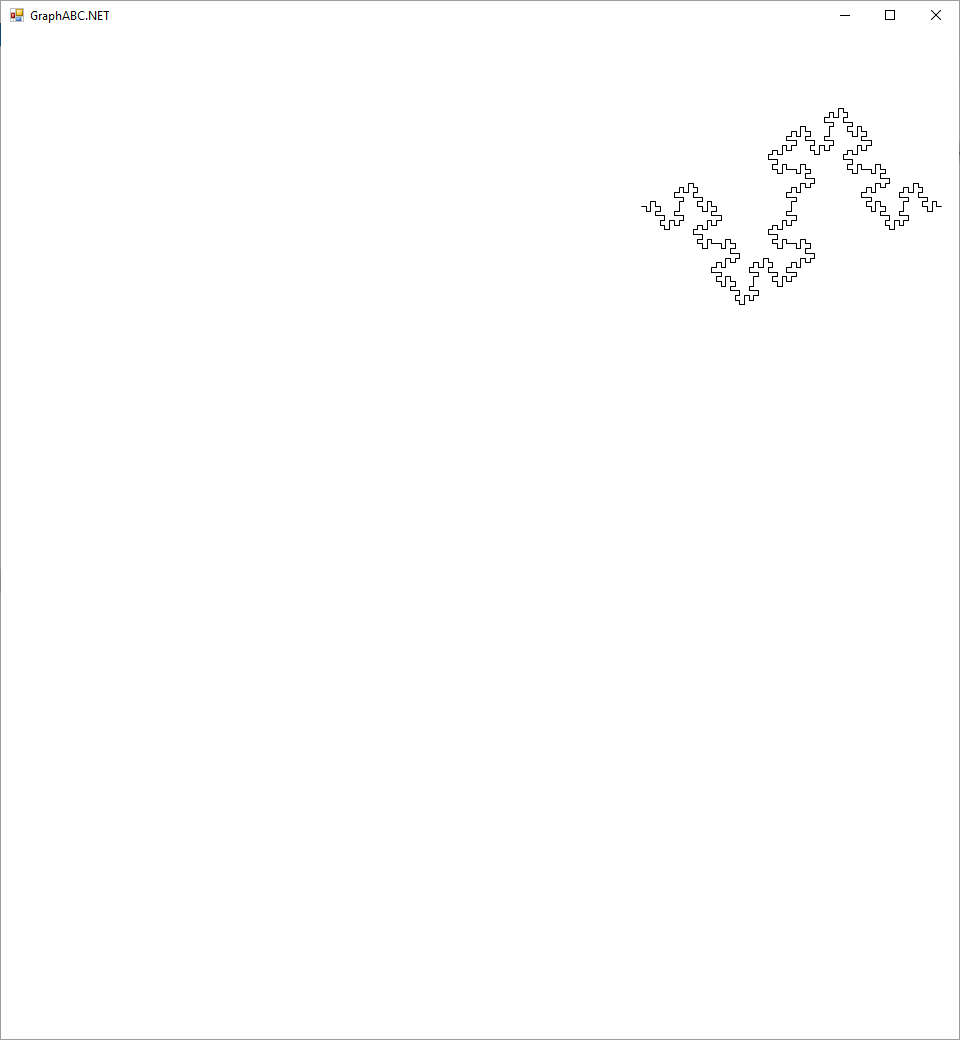


Рисунок 3 – Результат



**Вывод:** В ходе данной контрольной работы я научился пользоваться рекурсией и рисовать кривую Минковского. Получили навыки реализации алгоритмов с рекурсивным вычислениями, знакомств с фракталами.

В этой программе используется косвенная рекурсия – это последовательность взаимных вызовов нескольких функций, организованная в виде циклического замыкания на тело первоначальной функции, но с иным набором параметров.