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-101-51-00

Костин Борис Сергеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы**

Получение навыков реализации алгоритмов с рекурсивными вычислениями, знакомство с фракталами.

**Задание**

Вариант 15

1.Написать программу для визуализации фрактала “Кривая Минковского”

2.Предусмотреть возможности масштабирования, изменения глубины прорисовки и перемещения полученной фигуры.

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

**Описание алгоритма**

Построение «Кривой Минковского»: изначально есть прямой отрезок, который специфически преобразуется в ломаную, а при каждой следующей итерации к каждому прямому отрезку на ломанной применяется аналогичная процедура.

**Код программы**

**uses** GraphABC;

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

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

**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);

**begin**

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

**if** r < 10 **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);

//рисовка базовой линии

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: integer;

**procedure** KeyDown(key: integer); //Движение фрактала

**begin**

**case** key **of**

VK\_Up:

**begin**

y -= 10; y1 -= 10;

**end**;

VK\_Down:

**begin**

y += 10; y1 += 10;

**end**;

VK\_Left:

**begin**

x -= 10; x1 -= 10;

**end**;

VK\_Right:

**begin**

x += 10; x1 += 10;

**end**;

**end**;

Window.Clear;

**end**;

**procedure** KeyUp(a: integer);//глубина фрактала

**begin**

**case** a **of**

VK\_A:

**begin**

x := x - 50;

**end**;

Vk\_D:

**begin**

x := x + 50;

**end**;

**end**;

Window.Clear;

**end**;

**begin**

LockDrawing;

x := 100;

y := 200;

x1 := 400;

y1 := 200;

**repeat**

redraw;

draw(x, y, x1, y1);

onKeyDown := keydown;

onKeyUp := keyup;

**until**(false);

**end**.

**Результат выполнения программы**

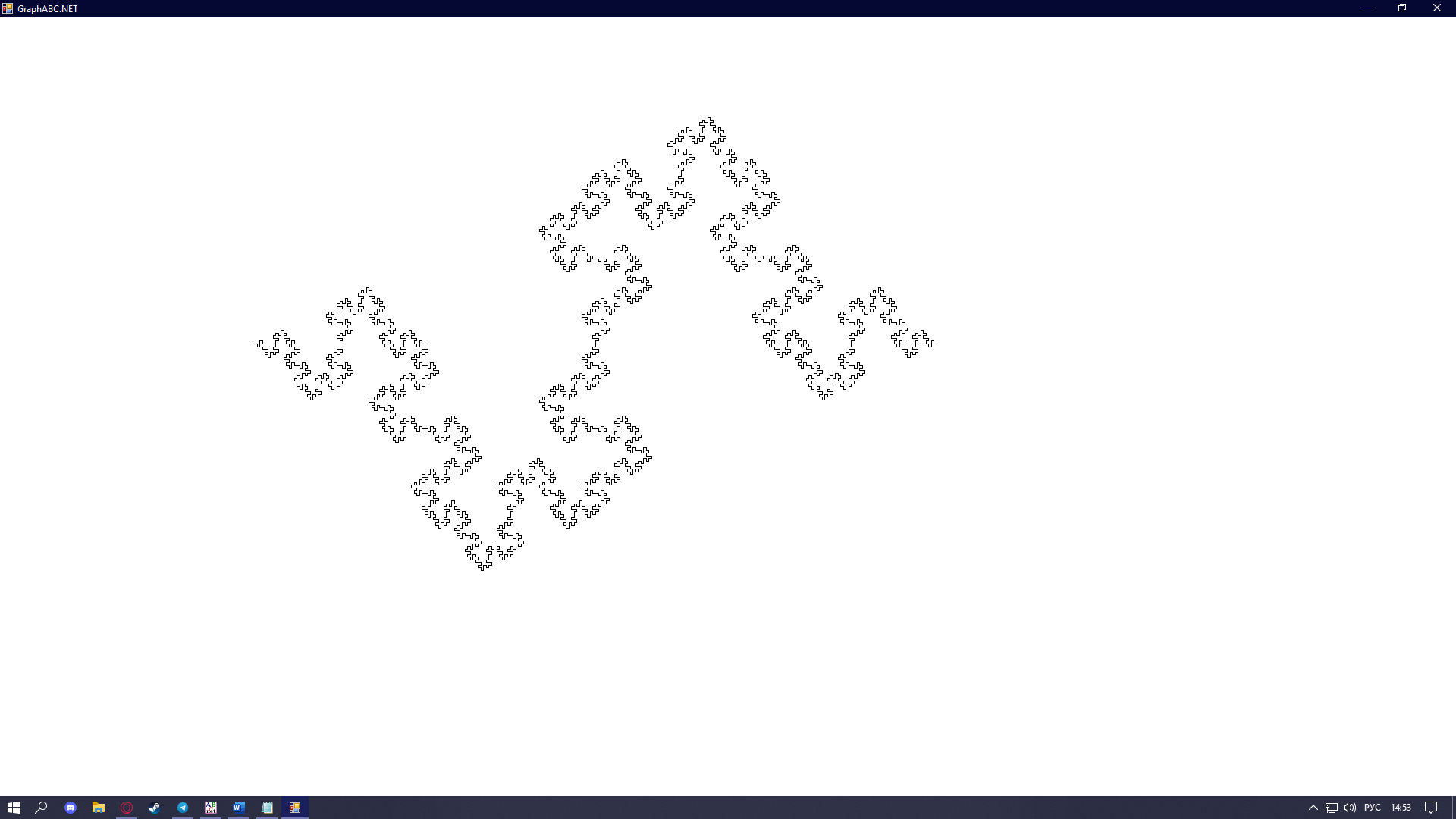


Рисунок 1 – Результат выполнения программы

**Вывод**

Выполняя данную домашнюю контрольную работу, я научился реализовывать алгоритмы с рекурсивными вычислениями, столкнулся со сложностями, связанными с предоставлением возможности изменения глубины прорисовки, масштабирования, а также с перемещением “Кривой Минковского”.