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»**

Выполнила: студентка учебной группы

ИСПк-202-52-00

Широнина Анна Андреевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2025

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

1. **Формулировка заданий. Вариант 19**

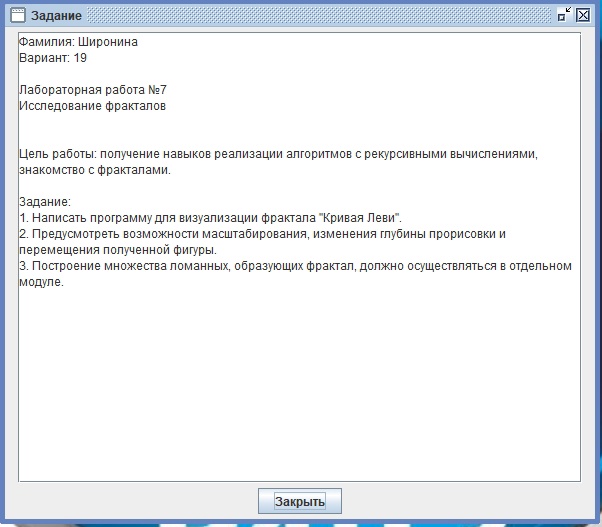


Рисунок 1 – Постановка задачки

1. **Описание алгоритма**

Фракталами называют геометрические фигуры, которые повторяют свою структуру на различных масштабах. **Кривая Леви – фрактал, предложенный французским математиком П. Леви. Процесс построения:** нужно взять половину квадрата, а затем каждую сторону заменить таким же фрагментом, и, повторяя эту операцию, в пределе получить кривую Леви. **Стандартная кривая Леви**строится с помощью равнобедренных треугольников с углами при основании 45°. **Вариации кривой Леви** можно построить с помощью равнобедренных треугольников с другими, отличными от 45° углами.

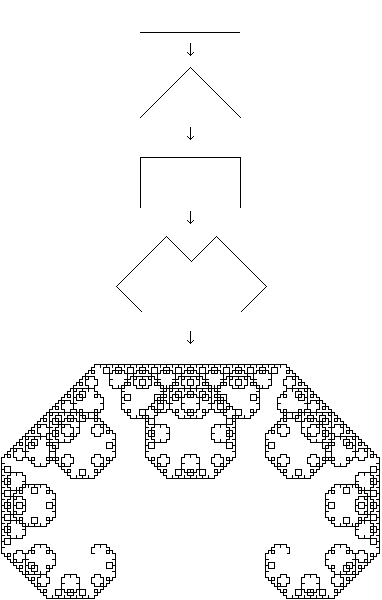
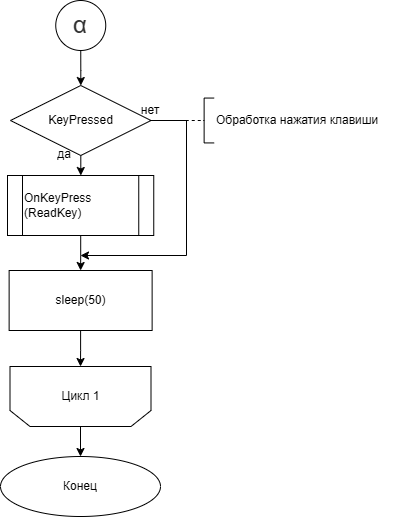
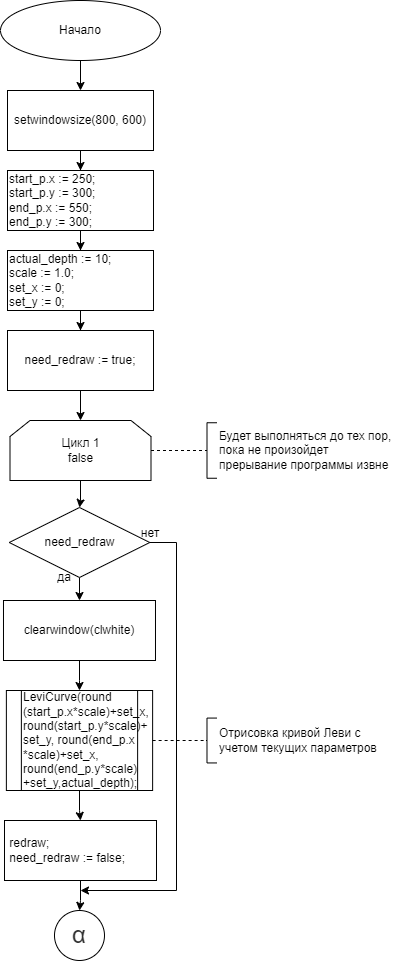


Рисунок 2 – Кривая Леви

Установка размеров графического окна. Задаем начальные и конечные точки кривой. Задаем начальную глубину рекурсии, масштаб и флажок необходимости перерисовки ставим на true. Далее идет цикл. Повторять пока не false: если нужна перерисовка, то очищаем графическое окно белым и вызываем рекурсивную процедуры для отрисовки кривой Леви, вычисляем параметры для подпрограммы. Перерисовываем и флажок ставим на false. Далее выполняем обработку нажатия клавиш. Если клавиша нажата, то вызов процедуры с case-меню, параметром для процедуры является нажатая клавиша. После обработки нажатия клавиши программа приостанавливается на 50 мс. Перемещение у нас идет с помощью стрелочек, масштабирование с помощью клавиш “-” и “+”, а изменение глубины рекурсии с помощью клавиш “1”..“9”.

1. **Схема алгоритма**

****

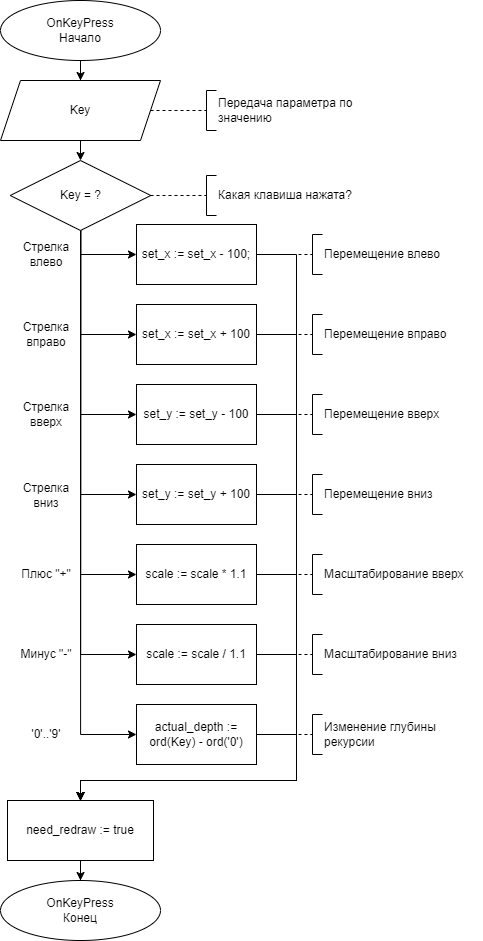
****

Рисунок 3 – Osnova.pas

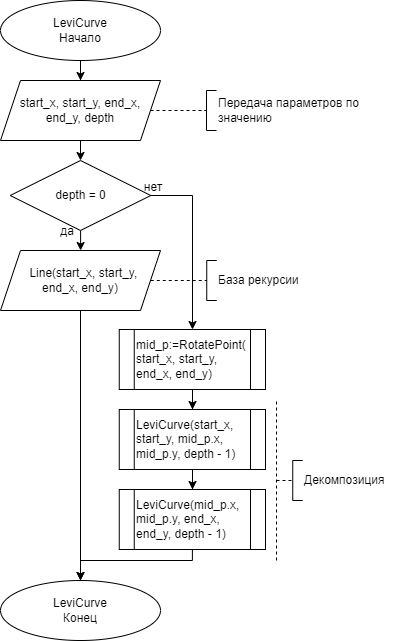
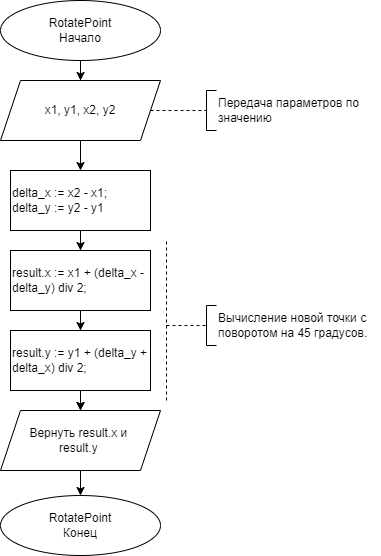
****

Рисунок 4 – LeviCurve\_m.pas

1. **Код программы**

**program** Osnova;

**uses**

LeviCurve\_m, Crt, GraphABC;

**const**

max\_depth = 10; //Максимальная глубина рекурсии

**type**

point = **record**

x, y: integer;

**end**;

//Новый тип - точка на плоскости с координатами x и y

**var**

start\_p, end\_p: point; //Начальная и конечная точки

actual\_depth: integer; //Текущая глубина

scale: real; //Коэффициент масштабирования

set\_x, set\_y: integer; // Смещения по осям X и Y

need\_redraw: boolean; // Необходимость перерисовки

//Обработчик нажатия клавиши

**procedure** OnKeyPress(Key: char);

**begin**

**case** Key **of**

#37: //Стрелка влево

set\_x := set\_x - 100; //Перемещение влево

#39: //Стрелка вправо

set\_x := set\_x + 100; //Перемещение вправо

#38: // Стрелка вверх

set\_y := set\_y - 100; //Перемещение вверх

#40: // Стрелка вниз

set\_y := set\_y + 100; //Перемещение вниз

'+':

scale := scale \* 1.1; //Масштабирование вверх

'-':

scale := scale / 1.1; //Масштабирование вниз

'0'..'9': //Изменение глубины

actual\_depth := ord(Key) - ord('0'); //Преобразование символа цифры в число

**end**;

need\_redraw := true; //Установить необходимость перерисовки

**end**;

//Основная программа

**begin**

setwindowsize(800, 600);

start\_p.x := 250;

start\_p.y := 300;

end\_p.x := 550;

end\_p.y := 300;

actual\_depth := 9; //Начальная глубина

scale := 1.0;

set\_x := 0;

set\_y := 0;

need\_redraw := true; //Изначально требуется перерисовка

**repeat**

**if** need\_redraw **then**

**begin**

clearwindow(clwhite);

// Отрисовка кривой Леви с учетом текущих параметров

LeviCurve(round(start\_p.x \* scale) + set\_x, round(start\_p.y \* scale) + set\_y,

round(end\_p.x \* scale) + set\_x, round(end\_p.y \* scale) + set\_y, actual\_depth);

redraw;

need\_redraw := false;

**end**;

**if** KeyPressed **then** //Обрабатывает нажатие клавиши

OnKeyPress(ReadKey);

sleep(50);

**until** false;

**end**.

**unit** LeviCurve\_m;

**uses**

GraphABC;

**type**

point = **record**

x, y: integer;

**end**;

//Процедура для отрисовки прямой линии

**procedure** DrawLine(start\_x, start\_y, end\_x, end\_y: integer);

**begin**

Line(start\_x, start\_y, end\_x, end\_y);

**end**;

//Процедура для вычисления координат новой точки при повороте на 45 градусов

**function** RotatePoint(x1, y1, x2, y2: integer): point;

**var**

delta\_x, delta\_y: integer;

**begin**

delta\_x := x2 - x1;

delta\_y := y2 - y1;

result.x := x1 + (delta\_x - delta\_y) **div** 2;

result.y := y1 + (delta\_y + delta\_x) **div** 2;

**end**;

//Рекурсивная процедура для отрисовки кривой Леви

**procedure** LeviCurve(start\_x, start\_y, end\_x, end\_y: integer; depth: integer);//Параметризация

**var**

mid\_p: point;

**begin**

**if** depth = 0 **then** //База рекурсии

**begin**

DrawLine(start\_x, start\_y, end\_x, end\_y); //Базовый случай - рисуем прямую линию

**end**

**else**

**begin**

mid\_p := RotatePoint(start\_x, start\_y, end\_x, end\_y); //Вычисление средней точки

//Декомпозиция

LeviCurve(start\_x, start\_y, mid\_p.x, mid\_p.y, depth - 1); //Рекурсивный вызов для первой половины

LeviCurve(mid\_p.x, mid\_p.y, end\_x, end\_y, depth - 1); //Рекурсивный вызов для второй половины

**end**;

**end**;

**end**.

1. **Результат выполнения программы**

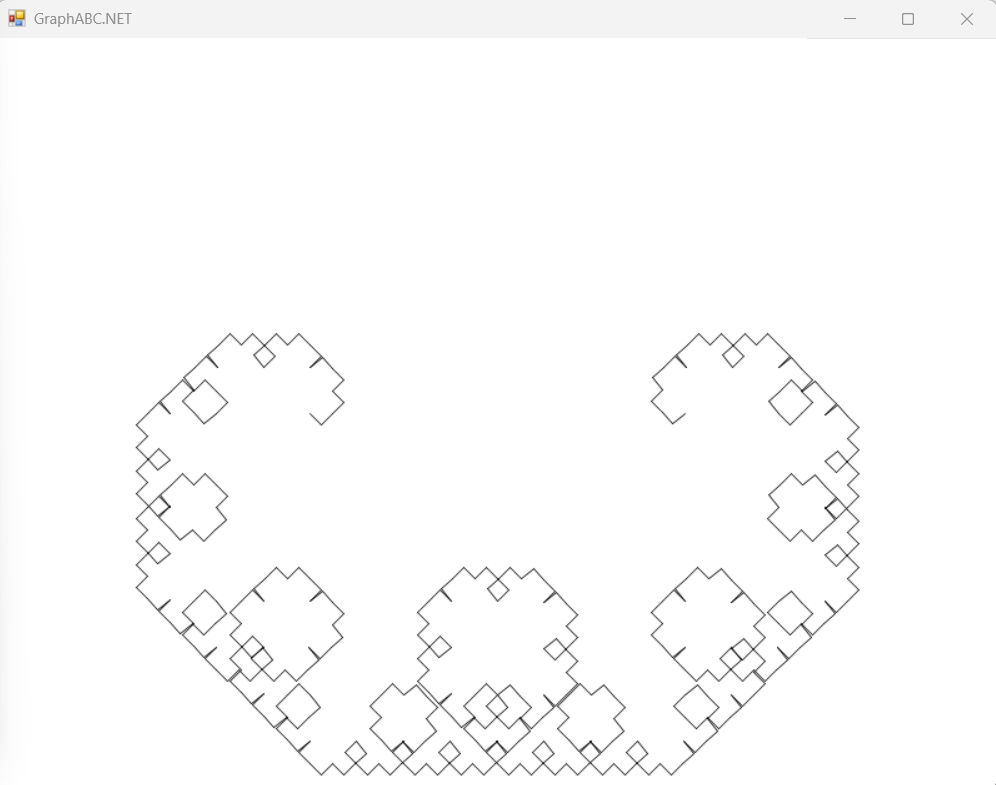
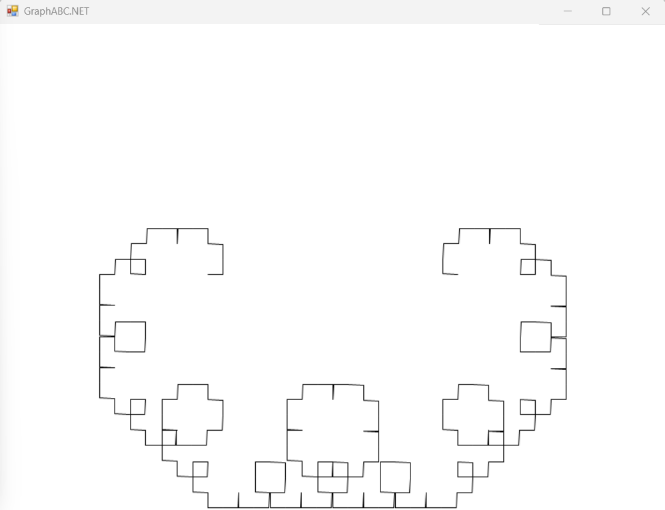
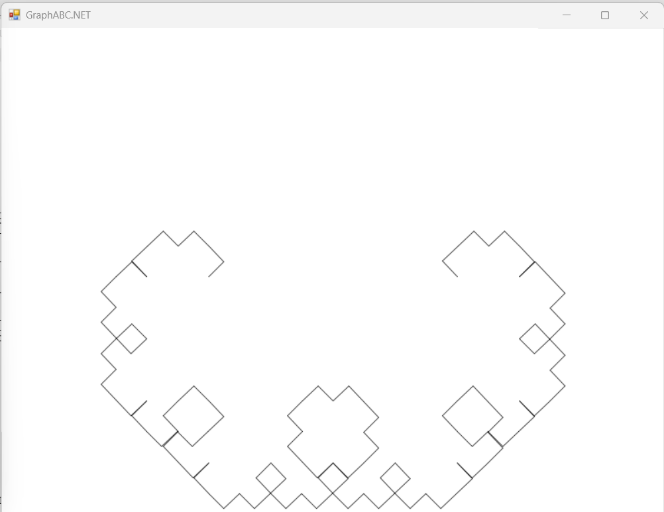
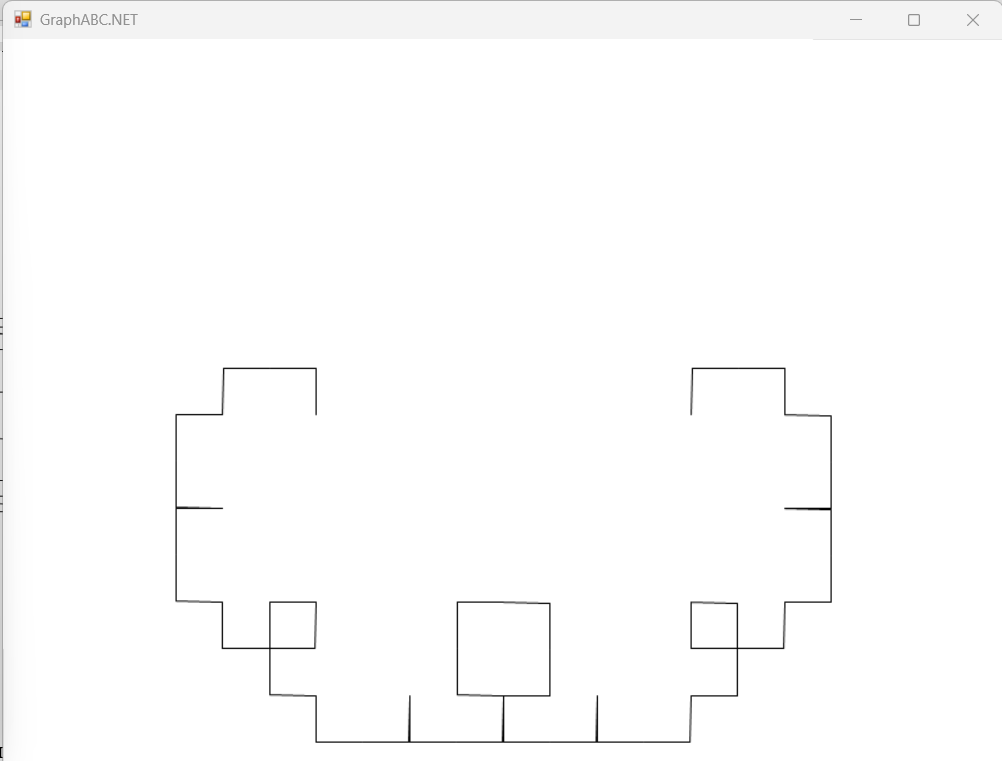
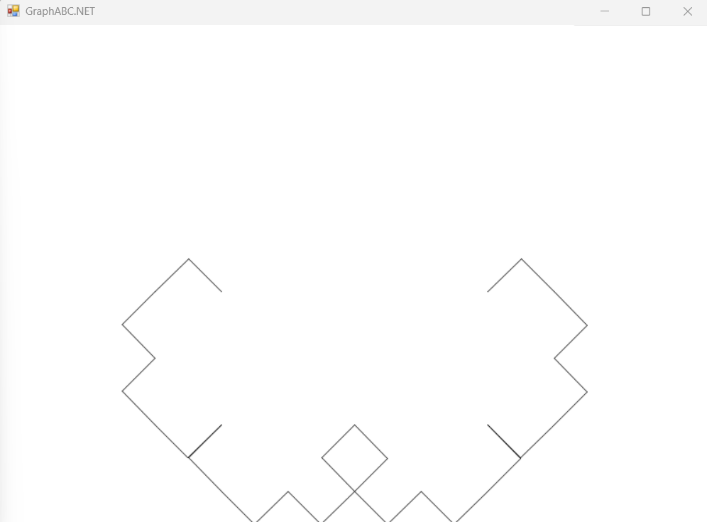
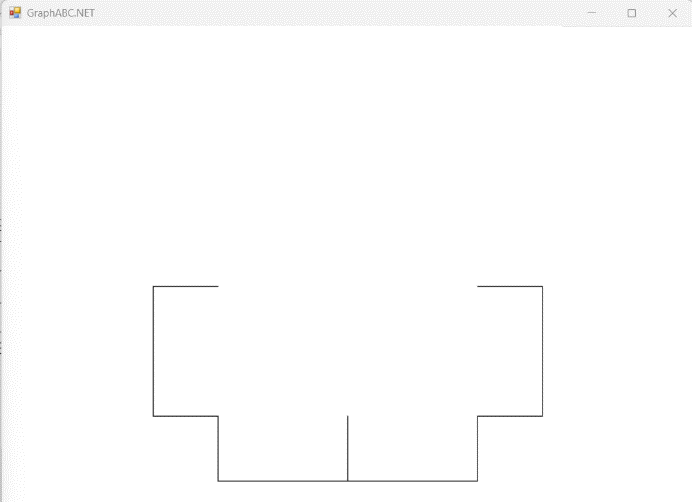
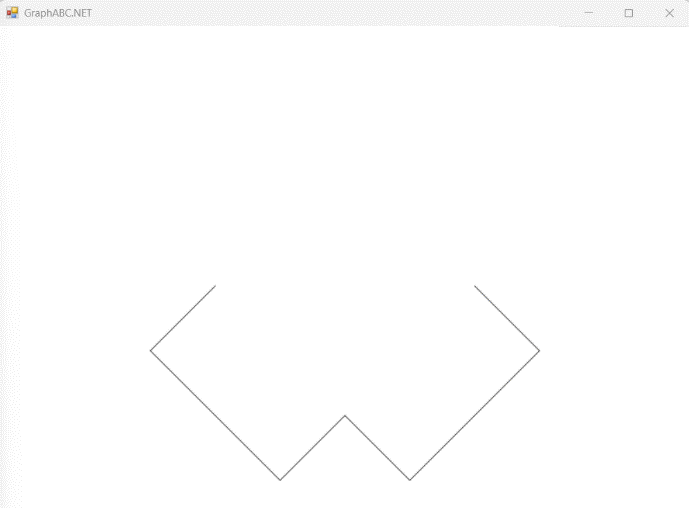
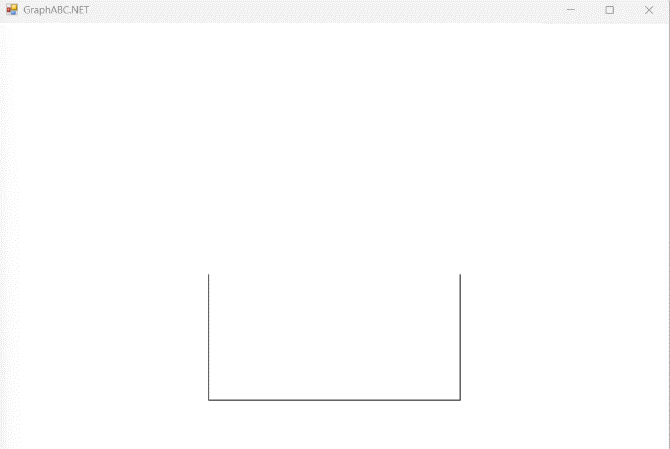
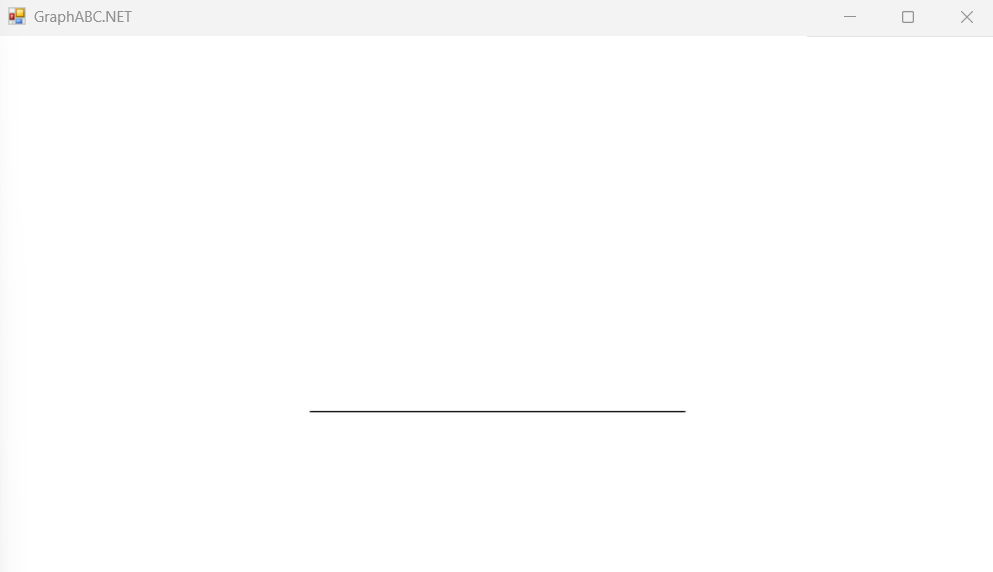


Рисунок 5 – Результат изменения глубины рекурсии от 0 до 9

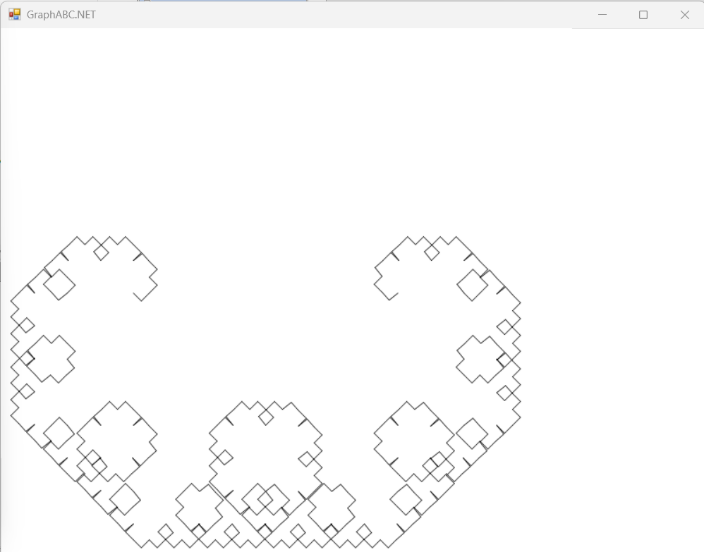
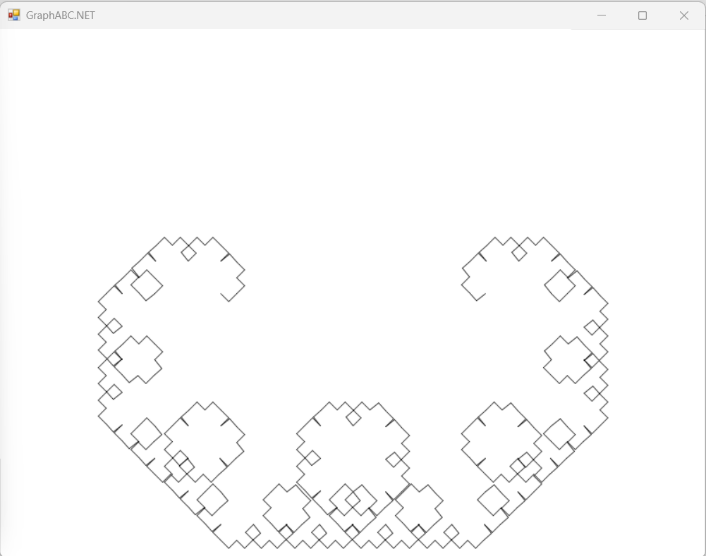


Рисунок 6 – Результат перемещения кривой влево

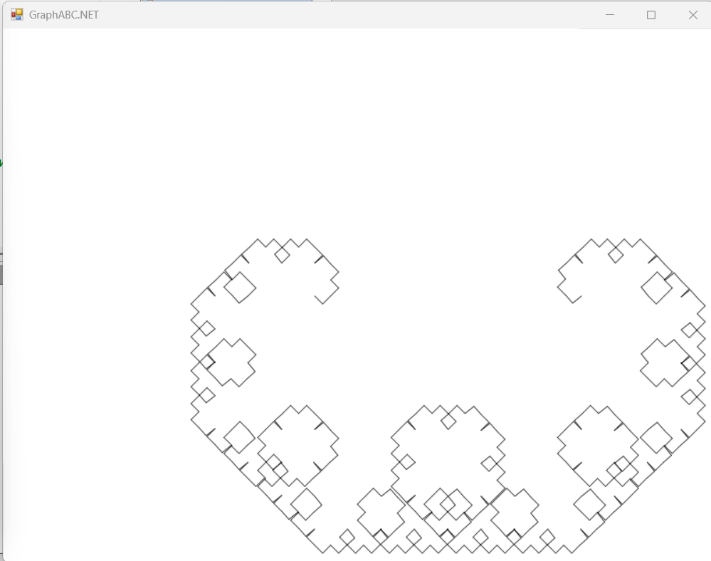
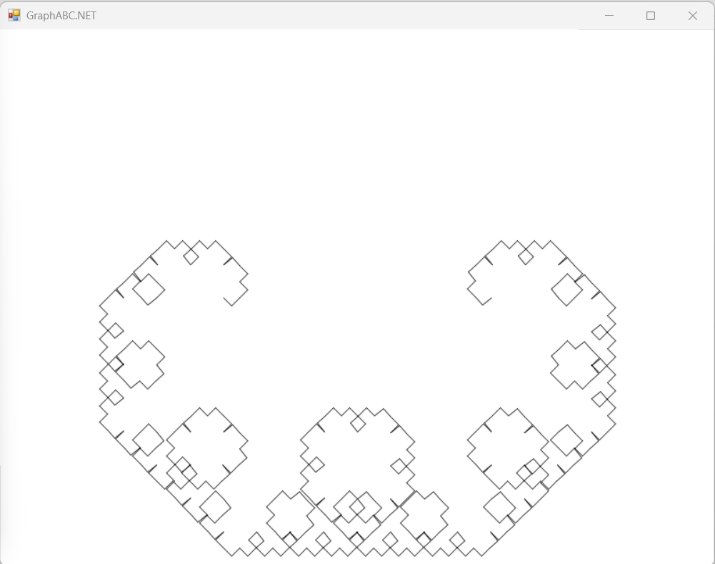


Рисунок 7 – Результат перемещения кривой вправо

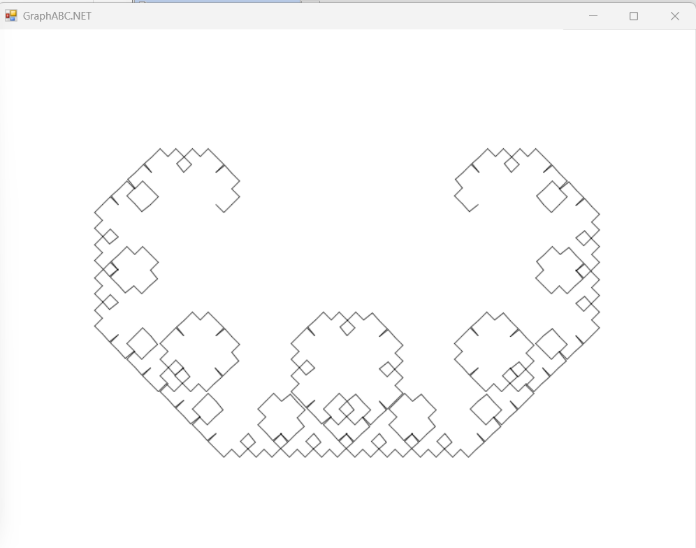
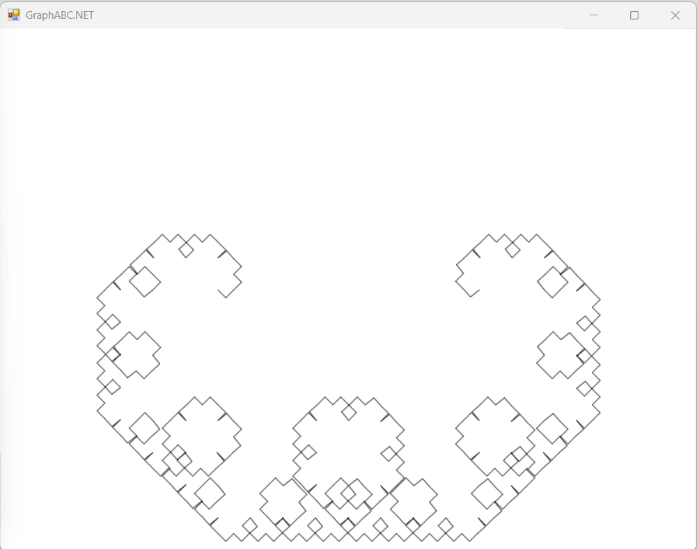


Рисунок 8 – Результат перемещения кривой вверх

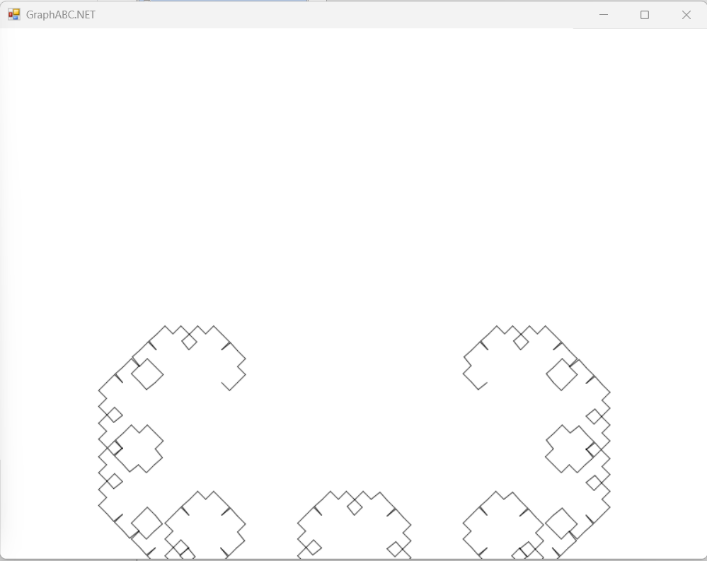
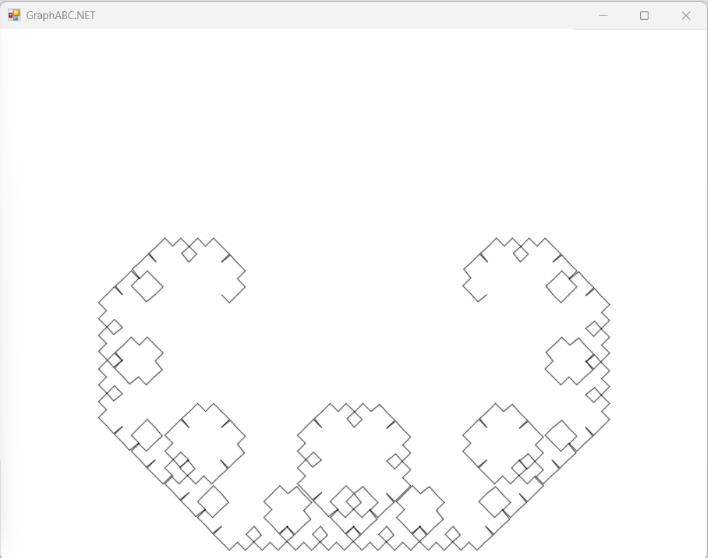
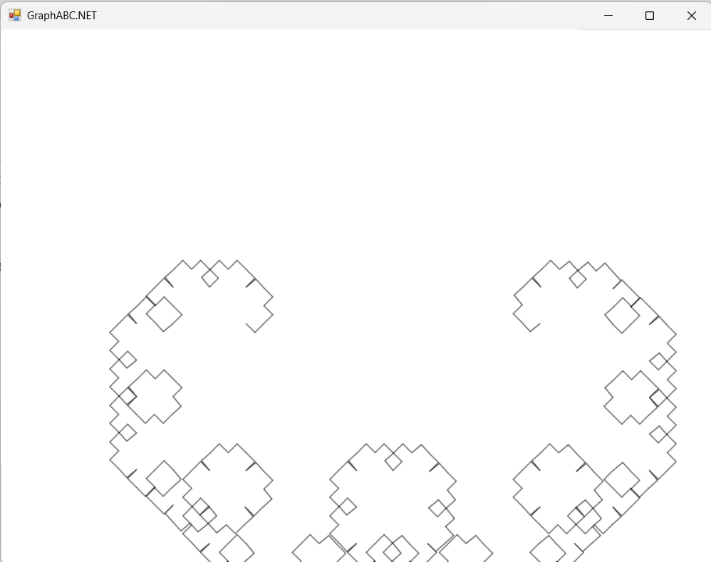
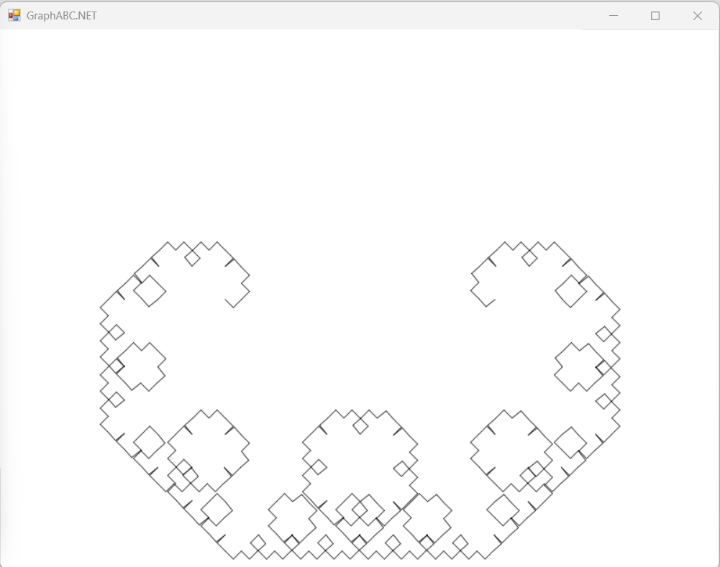


Рисунок 9 – Результат перемещения кривой вниз



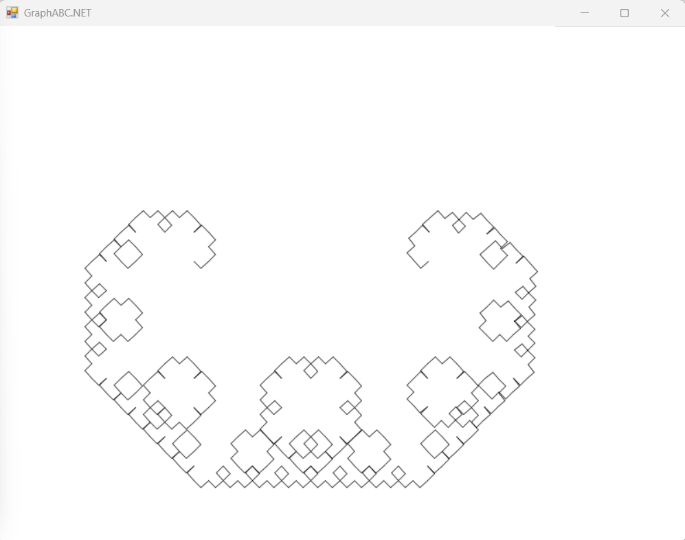
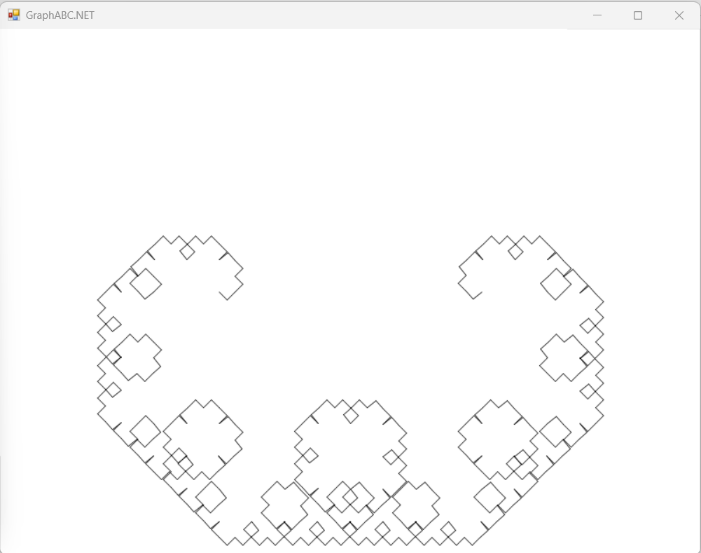


Рисунок 10 – Результат изменения масштаба кривой

1. **Вывод**

В процессе разработки программы для визуализации фрактала Кривая Леви на языке Паскаль мы получили навыки реализации алгоритмов с рекурсивными вычислениями и ознакомились с фракталами. Кроме того освоили создание модульной структуры программы и научились работать с клавишами в программе.

Для работы над схемами алгоритма необходимо четко сформулировать проблему, которую необходимо решить, собрать входные данные, необходимые для алгоритма, описать последовательность шагов, которые необходимо выполнить для решения задачи, использовать стандартные символы для построения схемы алгоритма (овалы – начало и конец, прямоугольники – действия или процессы, ромбы – условия или проверки и т.д.), проверить алгоритм на разных входных данных для выявления возможных ошибок, проанализировать алгоритм и внести изменения для повышения его эффективности, записать и оформить полученные результаты. Работа над схемами алгоритма помогает визуализировать процесс и обеспечить более легкое понимание и оптимизацию решения задачи.

Таким образом, в итоге у нас получилось реализовать рекурсивную триаду для построения фрактала, что позволило эффективно отобразить сложную структуру с использованием минимального объема кода. Также освоили создание модульной структуры программы и научились работать с клавишами в программе.