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 ДИСЦИПЛИНЕ «Основы алгоритмизации и программирования»**

Выполнил: студент учебной группы

ИСПк-205-52-00

Головёшкин Андрей Евгеньевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2025

**Вариант 2**

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

**Задание:**

1. Написать программу для визуализации фрактала "Снежинка Коха".
2. Предусмотреть возможности масштабирования, изменения глубины прорисовки и перемещения полученной фигуры
3. Построение множества ломанных, образующих фрактал, должно осуществляться в отдельном модуле

**Описание алгоритма:**

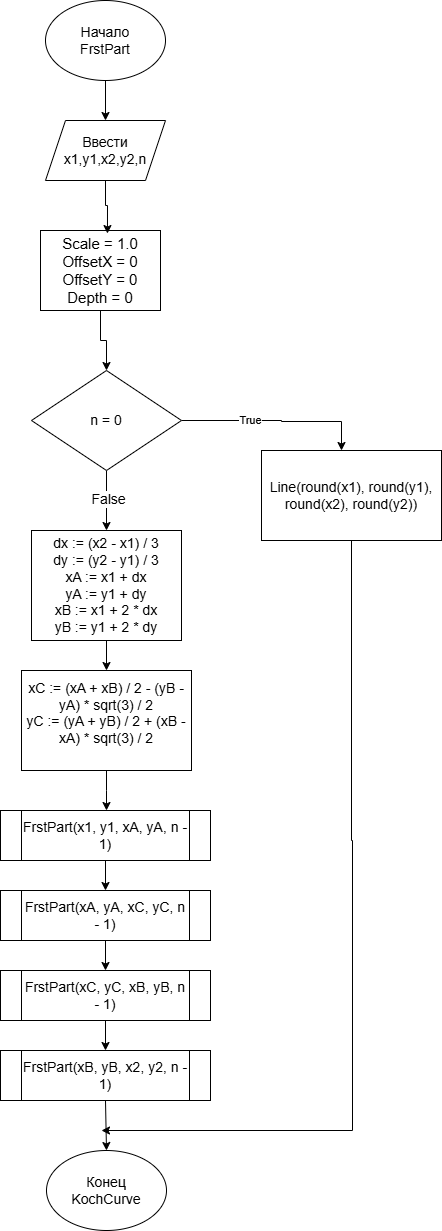
1. **Инициализация графического окна:**
   * Устанавливаем размер окна 600×600 пикселей.
   * Очищаем экран перед отрисовкой.
2. **Определение функции FrstPart(x1, y1, x2, y2, n):**
   * Входные параметры:
     + (x1, y1), (x2, y2) — начальная и конечная точки отрезка.
     + n — уровень рекурсии.
   * Если n = 0:
     + Рисуем прямую линию между (x1, y1) и (x2, y2).
   * Иначе:

* Разделяем отрезок на три равные части, вычисляя координаты точек A и B.
* Вычисляем координаты C, повернув точку на 60° против часовой стрелки:
* C формирует равносторонний треугольник с A и B.
* Рекурсивно вызываем FrstPart для четырех новых отрезков:
  + - * (x1, y1) → (xA, yA)
      * (xA, yA) → (xC, yC)
      * (xC, yC) → (xB, yB)
      * (xB, yB) → (x2, y2)

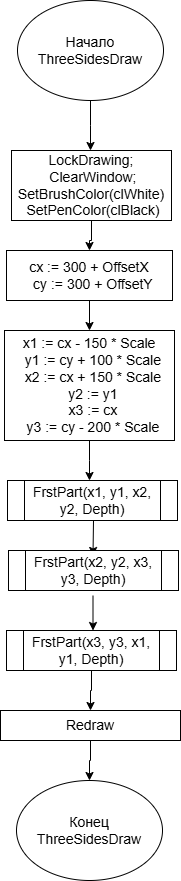
1. **Определение функции ThreeSidesDraw(size, n):**
   * Входные параметры:
     + size — длина стороны исходного треугольника.
     + n — уровень рекурсии.
   * Вычисляем координаты вершин равностороннего треугольника:
     + Вершина 1: (x1, y1) — левая нижняя.
     + Вершина 2: (x2, y2) — правая нижняя.
     + Вершина 3: (x3, y3) — верхняя.
   * Рисуем три стороны снежинки с помощью FrstPart (), поворачивая координаты по кругу.
2. **Запуск программы:**
   * Вызываем **ThreeSidesDraw**(300, 2) для создания снежинки с размером 300 и рекурсией 2.
   * Ждем нажатия клавиши перед закрытием окна.

**Схема алгоритма**

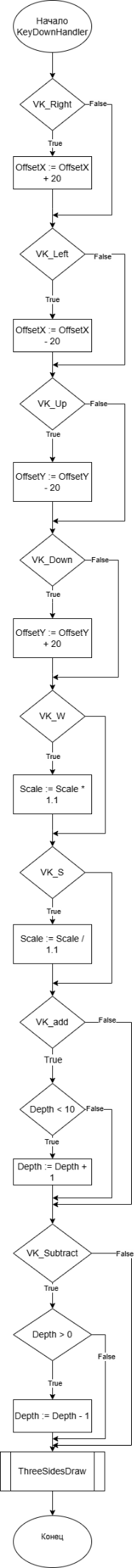
**Процедуры и функции:**

****

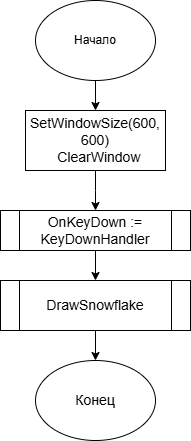
*Процедура 1*

**

*Процедура 2*

**

*Процедура 3*

**

*Программа 1*

**Код программы**

**Snow\_main**

**unit main;**

**uses GraphABC;**

**var**

**Scale: real = 1.0;**

**OffsetX: integer = 0;**

**OffsetY: integer = 0;**

**Depth: integer = 5;**

**procedure FrstPart(x1, y1, x2, y2: real; n: integer);**

**var**

**dx, dy, xA, yA, xB, yB, xC, yC: real;**

**begin**

**if n = 0 then**

**Line(round(x1), round(y1), round(x2), round(y2))**

**else**

**begin**

**dx := (x2 - x1) / 3;**

**dy := (y2 - y1) / 3;**

**xA := x1 + dx;**

**yA := y1 + dy;**

**xB := x1 + 2 \* dx;**

**yB := y1 + 2 \* dy;**

**xC := (xA + xB) / 2 - (yB - yA) \* sqrt(3) / 2;**

**yC := (yA + yB) / 2 + (xB - xA) \* sqrt(3) / 2;**

**FrstPart(x1, y1, xA, yA, n - 1);**

**FrstPart(xA, yA, xC, yC, n - 1);**

**FrstPart(xC, yC, xB, yB, n - 1);**

**FrstPart(xB, yB, x2, y2, n - 1);**

**end;**

**end;**

**procedure DrawSnowflake;**

**var**

**x1, y1, x2, y2, x3, y3: real;**

**cx, cy: real;**

**begin**

**LockDrawing;**

**ClearWindow;**

**SetBrushColor(clWhite);**

**SetPenColor(clBlack);**

**cx := 300 + OffsetX;**

**cy := 300 + OffsetY;**

**x1 := cx - 150 \* Scale;**

**y1 := cy + 100 \* Scale;**

**x2 := cx + 150 \* Scale;**

**y2 := y1;**

**x3 := cx;**

**y3 := cy - 200 \* Scale;**

**FrstPart(x1, y1, x2, y2, Depth);**

**FrstPart(x2, y2, x3, y3, Depth);**

**FrstPart(x3, y3, x1, y1, Depth);**

**Redraw;**

**end;**

**procedure KeyDownHandler(Key: integer);**

**begin**

**case Key of**

**VK\_Right: OffsetX := OffsetX + 20;**

**VK\_Left: OffsetX := OffsetX - 20;**

**VK\_Up: OffsetY := OffsetY - 20;**

**VK\_Down: OffsetY := OffsetY + 20;**

**VK\_W: Scale := Scale \* 1.1;**

**VK\_S: Scale := Scale / 1.1;**

**VK\_Add:**

**if Depth < 10 then**

**Depth := Depth + 1;**

**VK\_Subtract:**

**if Depth > 0 then**

**Depth := Depth - 1;**

**end;**

**DrawSnowflake;**

**end;**

**end.**

**mod.pas**

**program Snezhinka;**

**uses GraphABC, main;**

**begin**

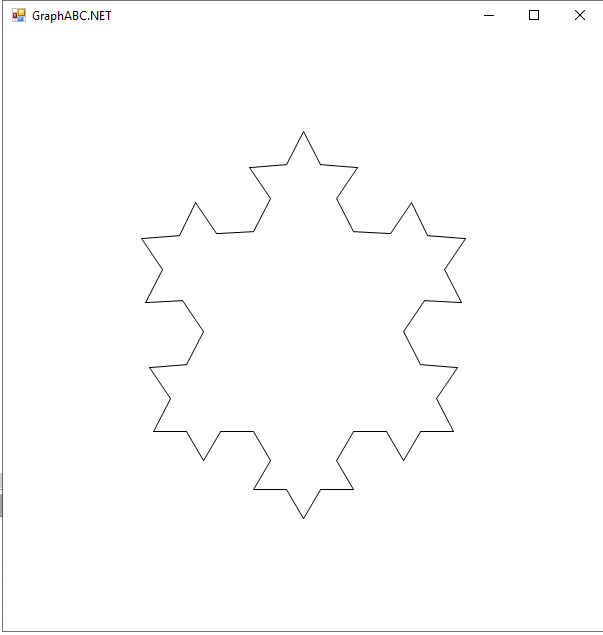
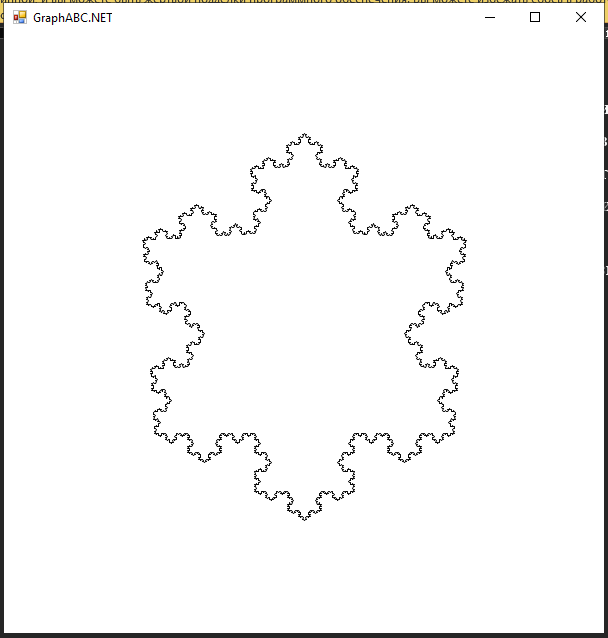
**SetWindowSize(600, 600);**

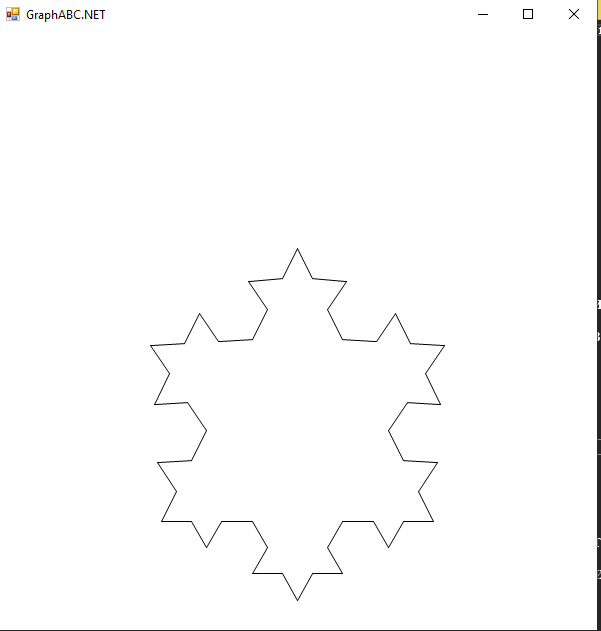
**OnKeyDown := KeyDownHandler;**

**DrawSnowflake;**

**end.**

**Скрины:**

****

****

**Вывод:**

В данной работе была реализована программа для построения **снежинки Коха** с использованием **GraphABC** Основой алгоритма стало рекурсивное разбиение отрезков и добавление нового сегмента в форме треугольника. В результате удалось построить фрактальную фигуру, демонстрирующую самоподобие. Работа позволила закрепить навыки использования рекурсии и работы с графикой в Pascal. Код можно модифицировать, изменяя уровень рекурсии или добавляя анимацию.