**Практическая работа №2**

**Фракталы и фрактальная графика**

1. Что такое фрактал? Какие виды бывают?

**Фрактал** (fractus — дроблёный, сломанный, разбитый) — это сложная, бесконечно самоподобная геометрическая фигура, каждый фрагмент которой повторяется при уменьшении масштаба.

Классификация фракталов:

* 1. Геометрические
  2. Алгебраические
  3. Стахостические

1. Что такое ломанная-генератор?

За один шаг алгоритма каждый из отрезков, составляющих ломаную, заменяется на ***ломаную-генератор***, в соответствующем масштабе.

1. Опишите фрактал из своего варианта ДКР.

Фрактал Хартера-Хейтуэя (также известный как кривая Хартера-Хейтуэя) – это фрактальная кривая, обладающая крайне сложной и красивой геометрией. Она была впервые представлена математиками Хартером и Хейтуэем. Кривая Хартера-Хейтуэя строится следующим образом:

1. Начнем с прямоугольника (или квадрата) как начальной фигуры.

2. Разделите его на девять равных прямоугольников (три строки по три столбца).

3. Удалите центральный прямоугольник.

4. Для каждого из оставшихся внешних прямоугольников повторите шаги с 2 по 4.

Повторяя этот процесс рекурсивно, вы получите все более сложные и красивые узоры, напоминающие змейку или лабиринт.

1. Укажите ломанную-генератор и опишите алгоритм построения фрактала в зависимости от глубины.

Алгоритм построения фрактала Хартера-Хейтуэя в зависимости от глубины можно описать следующим образом:

* Задаем начальную фигуру, которая может быть прямоугольником или квадратом.
* Разделим начальную фигуру на девять равных частей: три строки по три столбца.
* Удаляем центральную часть (центральный квадрат/прямоугольник).
* Рекурсивно применяем алгоритм к каждой из восьми оставшихся частей.
* Устанавливаем глубину рекурсии, которая определит количество итераций (уровней детализации) фрактала.
* Повторяем шаги 2-4 для каждого уровня глубины, увеличивая детализацию фрактала с каждым следующим уровнем.
* При достижении заданной глубины нужно остановится и завершить алгоритм.
* Каждый уровень глубины будет добавлять дополнительные детали и сложность фрактала.

1. Запишите рекурсивную подпрограмму для построения фрактала, в ней укажите рекурсивную триаду.

В данной рекурсивной подпрограмме присутствует рекурсивная триада, где основной прямоугольник разделяется на 9 частей, центральная часть заполняется, а затем рекурсивно вызывается функция для каждой из восьми оставшихся частей, исключая центральную.

Таким образом, данная подпрограмма рекурсивно строит фрактал Хартера-Хейтуэя на заданной глубине, создавая красивый и сложный узор из самоподобных структур.

Рекурсивная триада DrawFractal находится внутри основного блока программы, и она вызывается для построения фрактала Хартера-Хейтуэя в начале программы после инициализации окна и текста.

**program** fractal;

**uses** GraphABC;

**var**

i: integer;

**procedure** DrawHarterHeytwayFractal(x, y, side, depth: Integer);

**begin**

**if** depth = 0 **then**

**begin**

LineTo(x + side \* dx, y + side \* dy);

**end**

**else**

**begin**

side := side **div** 2;

DrawHarterHeytwayFractal(x, y, side, depth - 1);

dx := -dy;

dy := dx;

DrawHarterHeytwayFractal(x + side \* dx, y + side \* dy, side, depth - 1);

**end**;

**end**;

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

**begin**

**end**;

**begin**

// инициализация окна и текста

LockDrawing;

x := 200;

y := 140;

dx := 0;

dy := -4;

DrawHarterHeytwayFractal(x, y, 200, 5); // вызов рекурсивной процедуры для построения фрактала Хартера-Хейтуэя

redraw;

onKeyDown += keydown;

**end**.

**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 - увеличение');

textout(10, 200, 'x - уменьшение');

textout(10, 220, 'a - фрактал');

textout(10, 240, 'd - фрактал');

**end**;

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

**begin**

i := 0;

**case** key **of**

VK\_Up: **begin** y := y - 50 **end**;

VK\_Down: **begin** y := y + 50 **end**;

VK\_Left: **begin** x := x - 50 **end**;

VK\_Right: **begin** x := x + 50 **end**;

VK\_z:

**begin**

i := i + 1;

x := x + i;

dx := dx - i;

dy := dy + i;

y := y - i;

**end**;

Vk\_x:

**begin**

i := i - 1;

x := x - i;

dx := dx + i;

dy := dy - i;

y := y + i;

**end**;

vk\_a: **if** i > 0 **then** i -= 1;

vk\_d: **if** i < 4 **then** i += 1;

vk\_F:

**end**;

Window.Clear;

text;

draw(x, y, dx, dy);

redraw;

**end**;

**begin**

writeln('Left - влево');

writeln('Right - вправо');

writeln('Up - вверх');

writeln('Down - вниз');

writeln('z - увеличение');

writeln('x - уменьшение');

writeln('a - фрактал');

writeln('d - фрактал');

LockDrawing;

x := 200;

y := 140;

dx := 0;

dy := -4;

draw(x, y, dx, dy);

redraw;

onKeyDown += keydown;

**end**.