Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
им. Н.П. ОГАРЁВА»
(ФГБОУ ВО «МГУ им. Н.П. Огарёва»)

Факультет математики и информационных технологий

Кафедра систем автоматизированного проектирования

ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №5

по дисциплине: Интеллектуальные системы

ФРАКТАЛЬНЫЕ ВЫЧИСЛЕНИЯ

Автор отчёта о лабораторной работе _____ А. Е. Конышев
подпись, дата

Обозначение лабораторной работы ЛР–02069964–02.03.02–08–23

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Руководитель работы
преподаватель кафедры
систем автоматизированного
проектирования _____ А. А. Шалаева
подпись, дата

Саранск 2023

Цель работы: ознакомление с «мягкими» вычислениями; приобретение навыков фрактальных вычислений.

Ход работы

Задание 1. Изучить основные понятия и принципы фрактальной геометрии и фрактальных вычислений.

Задание 2. Реализовать программу на выбранном языке программирования, которая строит фрактальные структуры. Программа должна обладать возможностью настройки параметров фрактала, таких как масштаб, цвета, глубина рекурсии и т.д.

Задание 3. Необходимо сначала вывести базовую фигуру или «затравку»

Задание 4. Далее вывести фигуру первого поколения. Она должна быть идентичной базовому объекту.

Задание 5. Вывести несколько фигур различного поколения. Ограничения рекурсии задать номером поколения (уровнем глубины).

Задание 6. Вывести несколько фигур различного поколения. Ограничения рекурсии задать размером базовой фигуры.

Описание выполнения работы

Задание 1, 2, 3. Разработаем программу для рисования кривой Пеано, где ограничим рисование глубиной рекурсии

```
def fract_draw_by_depth(stp, rule, ang, dept):
    if dept < 1:
        return
    if rule == "a":
        turtle.left(ang)
        fract_draw_by_depth(stp, "b", ang, dept - 1)
        turtle.forward(stp)
        turtle.right(ang)
        fract_draw_by_depth(stp, "a", ang, dept - 1)
        turtle.forward(stp)
        fract_draw_by_depth(stp, "a", ang, dept - 1)
        turtle.right(ang)
        turtle.forward(stp)
        fract_draw_by_depth(stp, "b", ang, dept - 1)
        turtle.left(ang)
    if rule == "b":
        turtle.right(ang)
        fract_draw_by_depth(stp, "a", ang, dept - 1)
        turtle.forward(stp)
        turtle.left(ang)
        fract_draw_by_depth(stp, "b", ang, dept - 1)
        turtle.forward(stp)
        fract_draw_by_depth(stp, "b", ang, dept - 1)
        turtle.left(ang)
        turtle.forward(stp)
        fract_draw_by_depth(stp, "a", ang, dept - 1)
        turtle.right(ang)


turtle.speed(10)
turtle.penup()
turtle.goto(-200, -100)
turtle.pendown()
fract_draw_by_depth(30, "a", 90, 1)
turtle.done()
```

В функции `fract_draw_by_depth()` 4 параметром мы указываем глубину рисования, эта функция вызывается внутри себя но уже с уменьшенной на единицу глубиной, и когда эта глубина меньше единицы, то выходим из функции. Результат работы программы продемонстрирован на рисунке 4.1

Рисунок 4.1 – Базовая фигура кривой Пеано

Задание 4, 5. Выведем 4 фигуры различного поколения, начиная с первой, равной базовой. Результаты отображены на соответствующих рисунках.

Рисунок 4.2 – фигура поколения 1 или базовая фигура

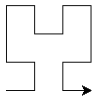

Рисунок 4.3 – фигура поколения 2

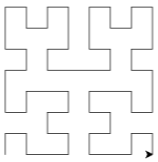

Рисунок 4.4 – фигура поколения 3

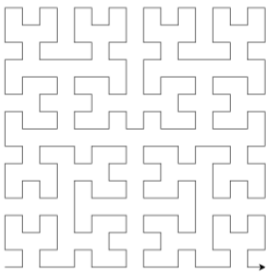

Рисунок 4.5 – фигура поколения 4

Задание 6. Для выполнения задания необходимо переписать исходную программу. Задав ограничение рекурсии длиной базовой фигуры

```
def fract_draw_by_step(stp, rule, ang):
    if stp < 40:
        return
    stp /= 1.1
    if rule == "a":
        turtle.left(ang)
        fract_draw_by_step(stp, "b", ang)
        turtle.forward(stp)
        turtle.right(ang)
        fract_draw_by_step(stp, "a", ang)
        turtle.forward(stp)
        fract_draw_by_step(stp, "a", ang)
        turtle.right(ang)
        turtle.forward(stp)
        fract_draw_by_step(stp, "b", ang)
        turtle.left(ang)
    if rule == "b":
        turtle.right(ang)
        fract_draw_by_step(stp, "a", ang)
        turtle.forward(stp)
        turtle.left(ang)
        fract_draw_by_step(stp, "b", ang)
        turtle.forward(stp)
        fract_draw_by_step(stp, "b", ang)
        turtle.left(ang)
        turtle.forward(stp)
        fract_draw_by_step(stp, "a", ang)
        turtle.right(ang)
```

Такое ограничение «искусственное», так как необходимо изначально рассчитать его на основе изначального размера базовой фигуры. Для демонстрации будем вызывать fract_draw_by_step() с различным первым аргументом. Сначала с stp=40, затем с stp=40*1.1 и так далее. Результаты показаны на соответствующий рисунках.

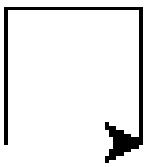

Рисунок 4.6 – фигура поколения 1

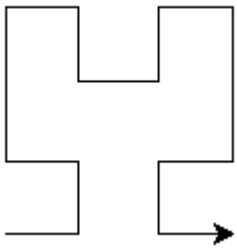

Рисунок 4.7 – фигура поколения 2

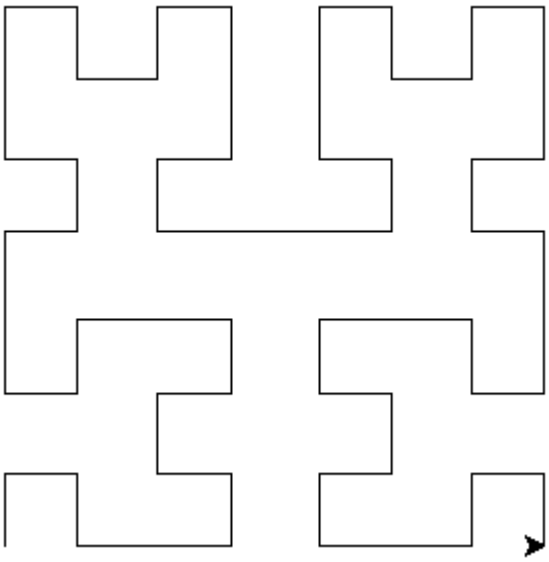

Рисунок 4.8 – фигура поколения 3

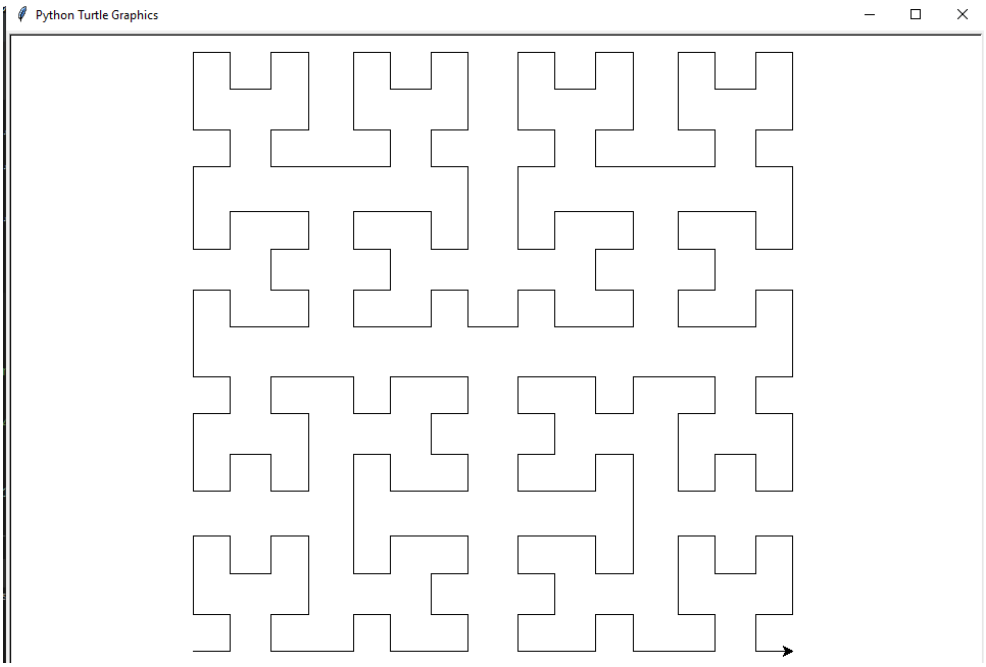

Рисунок 4.9 – фигура поколения 4