НЕРАВНОВЕСНАЯ АГРЕГАЦИЯ, ФРАКТАЛЫ

Виноградова Варвара НФИбд-01-18

Жижченко Глеб НФИбд-01-18

Жижченко (Ветошкина) Валерия НФИбд-03-18

Греков Максим НФИбд-01-18

Кондратьева Анастасия НФИбд-01-18

Иванова Ольга НФИбд-01-18

Тема №1, этап 4

ЦЕЛЬ РАБОТЫ

- Изучить теоретический материал по теме;
- Собрать, подготовить и структурировать материал для создания универсальной программы для моделирования выбранного фрактала;
- Выполнить реализацию данной программы на языке программирования Python.

ВВЕДЕНИЕ

Существуют разнообразные физические процессы, основная черта которых — **неравновесная агрегация.**

Она заключается в необратимом прилипании частиц к растущему кластеру из-за сильного смещения равновесия в сторону твердой фазы, вырастают разветвленные агрегаты.

Простейший случай — агрегация, ограниченная диффузией (Diffusion Limited Aggregation, DLA).

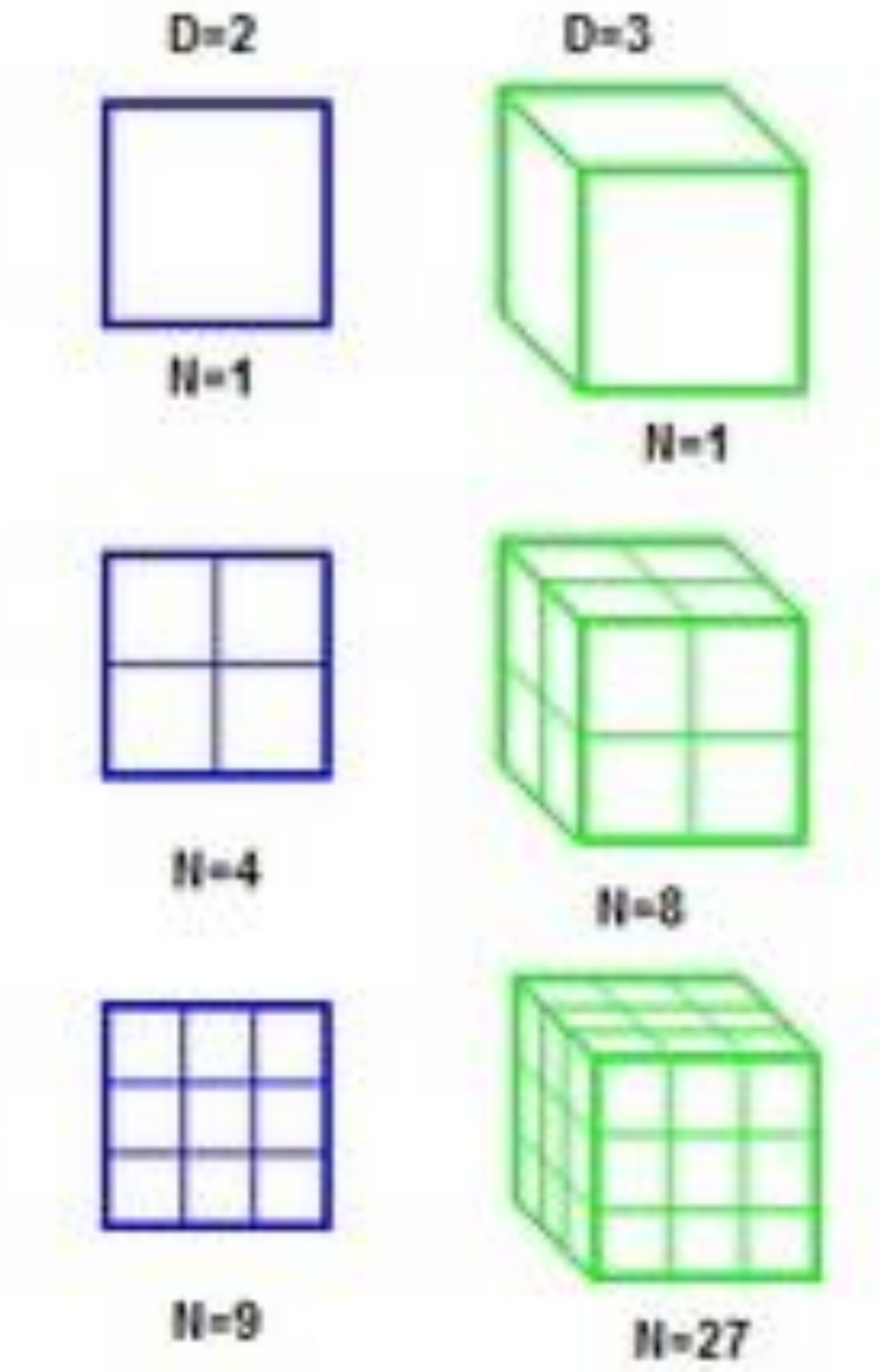
Также существуют:

- Химически-ограниченная агрегация;
- Баллистическая агрегация;
- Кластер–кластерная агрегация.

Примеры:

- образование частиц сажи;
- рост осадков при электрическом осаждении;
- «вязкие пальцы» при вытеснении вязкой жидкости менее вязкой в пористой среде (например, нефти — водой или газом внутри пласта);
- электрический пробой.

Рост правильных ограненных кристаллов происходит в условиях, близких к равновесным, когда возможно как прилипание частиц, так и их обратный переход в раствор.



ФРАКТАЛЬНАЯ РАЗМЕРНОСТЬ

Фигура на плоскости или тело в пространстве имеют размерность.

Интуитивно мы понимаем термин размерность как число координат, необходимых для задания положения точки внутри фигуры. Так, любая линия (например, окружность или прямая) одномерна — достаточно всего одной координаты, чтобы точно указать точку, а плоскость и поверхность шара двумерны.

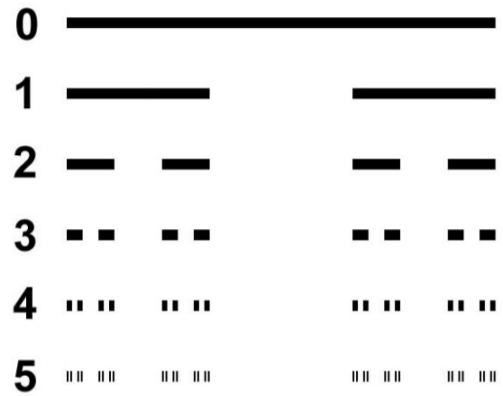
Определить ее можно разными способами:

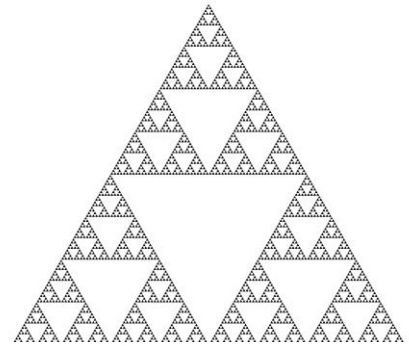
1. Метод сфер или ящиков;
2. Метод подсчета клеток;
3. Существует метод, применяемый при наблюдении за процессом роста агрегата от центра.
4. Метод малоуглового рассеяния света, рентгеновских лучей или нейтронов.

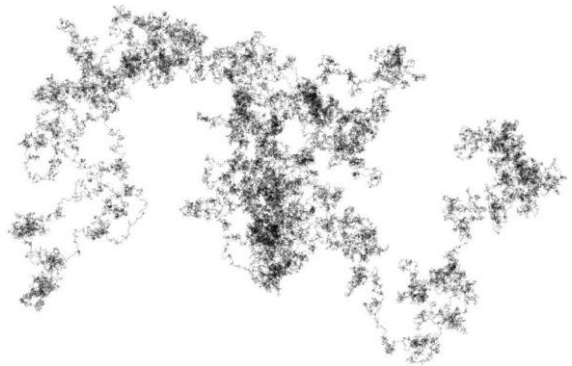
Рисунок 1. Размерности фигуры

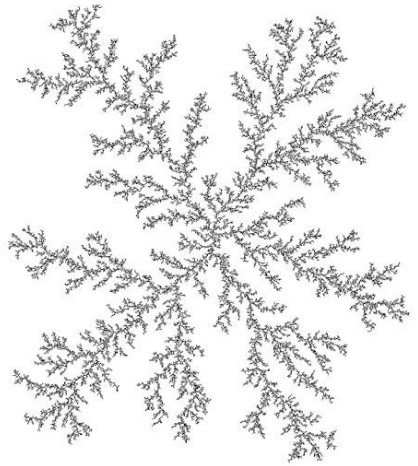
ПРИМЕРЫ «МАТЕМАТИЧЕСКИХ ФРАКТАЛОВ»

- множество Кантора;
- кривая Коха;
- треугольник Серпинского;
- траектория Броуновской частицы;
- бессеточная модель;
- химически-ограниченная агрегация;
- баллистическая агрегация;
- кластер–кластерная агрегация.









ДЕРЕВО ПИФАГОРА

Для построения Дерева Пифагора используется генератор в виде прямоугольного треугольника с квадратом на гипотенузе. Применяя этот генератор к самому себе, получим первую итерацию с двумя новыми треугольниками.

Продолжая процесс, получим новые поколения, для которых количество треугольников увеличивается каждый в два раза. В пределе этого процесса «вырастет» Дерево Пифагора

Одним из свойств дерева Пифагора является то, что, если площадь первого квадрата равна единице, то на каждом уровне сумма площадей квадратов тоже будет равна единице.

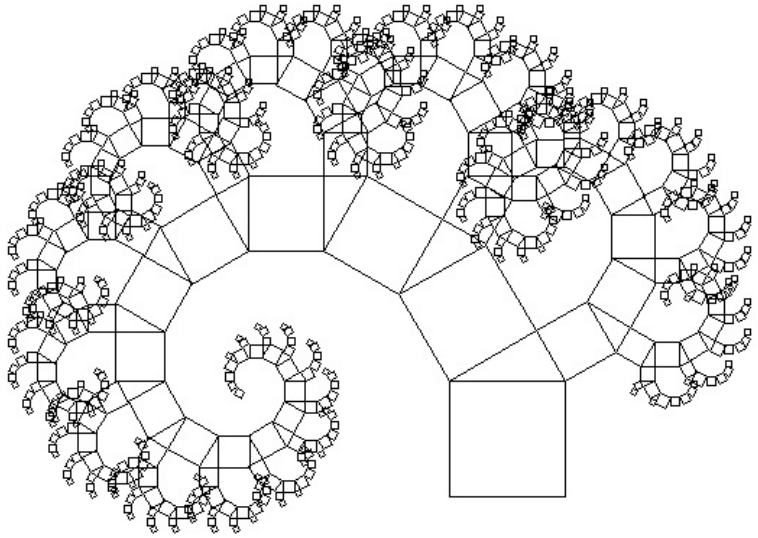
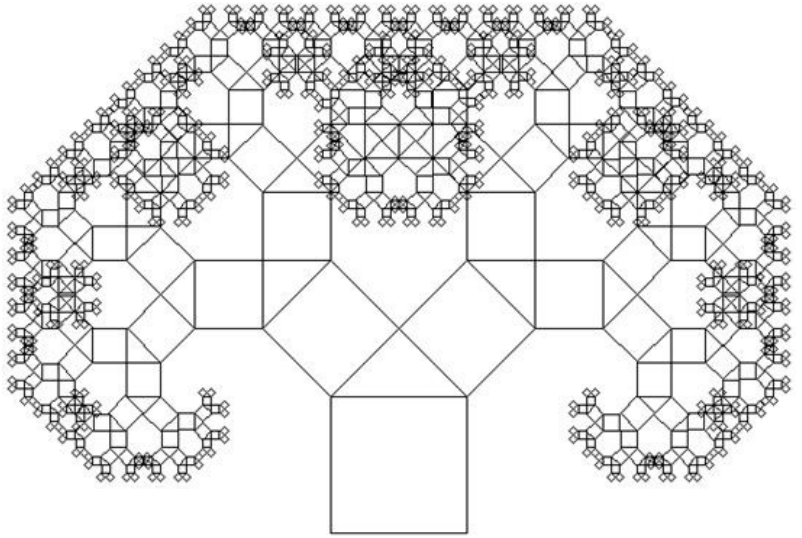


Рисунок 2, 3. Дерево Пифагора



ФРАКТАЛЬНАЯ РАЗМЕРНОСТЬ ДЕРЕВА ПИФАГОРА

Если площадь начального квадрата S – площади на втором этапе – S/2 + S/2.

Коэффициент сжатия (масштабирования) –

$$r = \frac{1}{\sqrt{2}};$$

$$d = \frac{log(2)}{log(^{2}/_{\sqrt{2}})} = 2.$$

Рекурсивная функция (от лат. recursio — возвращение) — это функция f(n) аргументов, которая в своей записи содержит себя же.

1 ЭТАП

На первом этапе мы изучили теоретический материал по теме «Неравновесная агрегация, фракталы», а именно:

1. Изучили теоретическое описание задачи и модели.
2. Рассмотрели различные типы фракталов (множество Кантора, кривая Коха, треугольник Серпинского, траектория броуновской частицы).
3. Изучили способы определения фрактальной размерности (метод сфер или ящиков, метод подсчета клеток и т.д.)

КАК ЧЕРТИТЬ ФРАКТАЛЫ ПРИ ПОМОЩИ PYTHON?

Как правило, отрисовка фракталов сложна, так как глубинная природа фракталов определяется концепцией рекурсии. Говоря о графиках и их вычерчивании, мы обычно считаем, что они образованы пикселями или векторами, но количество пикселей или векторов всегда ограничено, а фракталы по определению бесконечно рекурсивны. Таким образом, попытавшись нанести фрактал на координатную сетку, мы в какой-то момент должны будем остановиться, и именно поэтому мы в данном случае говорим об «итерациях». На каждой итерации фрактал становится все сложнее, и в какой-то момент становится невозможно отличить две его итерации, следующие друг за другом (такой момент наступает, когда изменения происходят на уровне, сравнимом с размером пикселя). Здесь логично остановиться, но, как правило, форма фрактала вырисовывается быстрее, и остановиться можно еще раньше.

ВХОДНЫЕ ДАННЫЕ

Точка нижний левый угол

Длина стороны (lenght)

Угол наклона основания (base_angle)

Угол наклона левой ветви (angle)

ЗАДАНИЕ ТОЧЕК

x1 = x0 + length * cos(base_angle)

y1 = y0 - length * sin(base_angle)

x2 = x0 + length * cos(base_angle + 90)

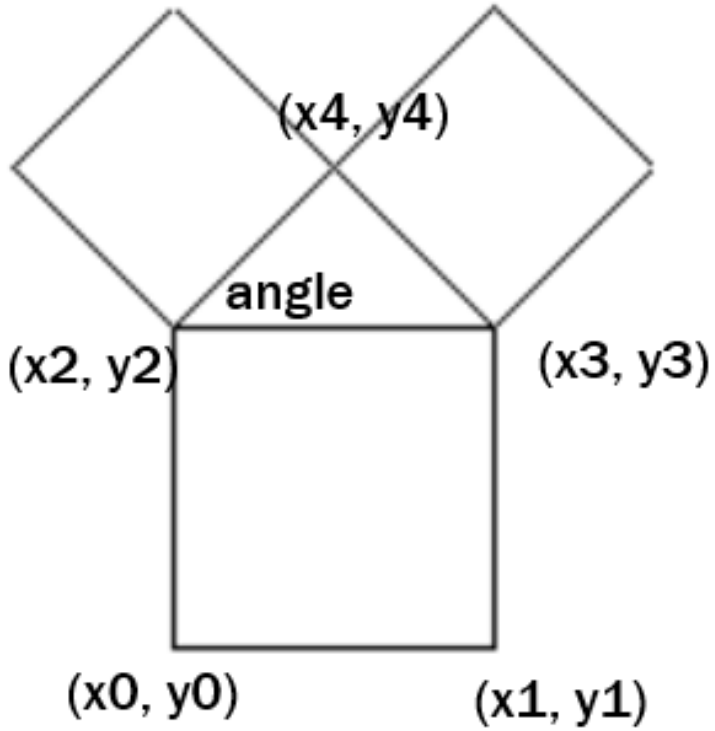
y2 = y0 - length * sin(base_angle + 90)

x3 = x1 + length * cos(base_angle + 90)

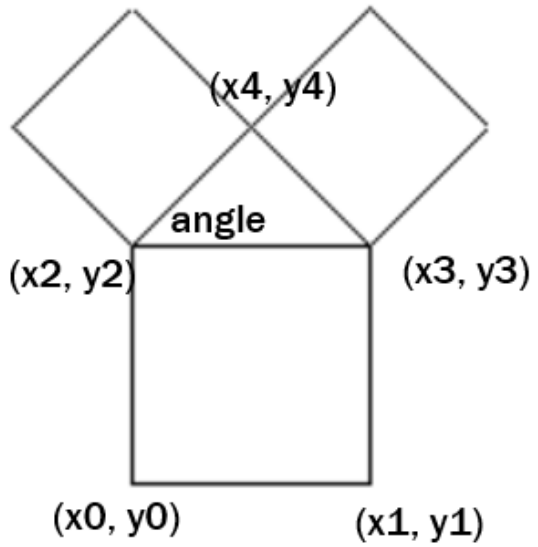
y3 = y1 - length * sin(base_angle + 90)

x4 = x2 + length * cos(angle) * cos(base_angle + angle)

y4 = y2 - length * cos(angle) * sin(base_angle + angle)



АЛГОРИТМ ПРОХОЖДЕНИЯ ИТЕРАЦИЙ



Каждая следующая итерация начинается с нижнего левого угла, после чего мы находим все точки и рисуем их.

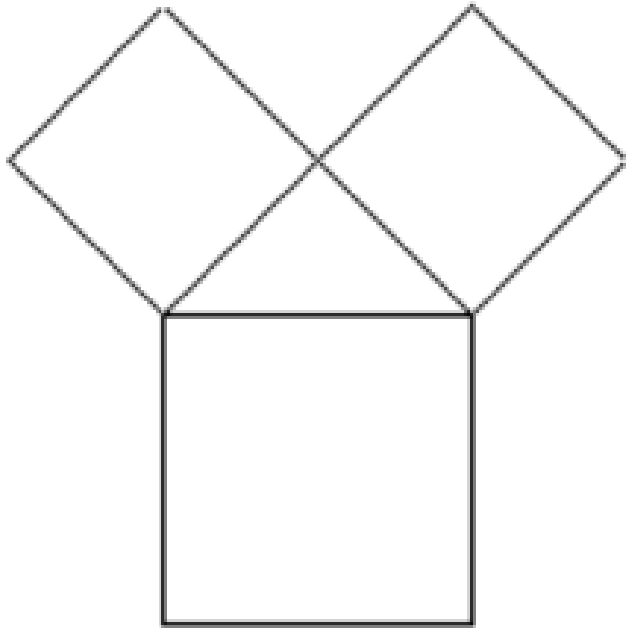
Затем соединяем точки между собой, получая линии.

ОТРИСОВКА ЛЕВОЙ ВЕТВИ

Нижняя левая точка для левой ветви совпадает с (x2,y2).

Длина нового левого дочернего квадрата равна *left_length = length * cos(angle)*.

base_angle для левого квадрата *base_angle родительского + angle*

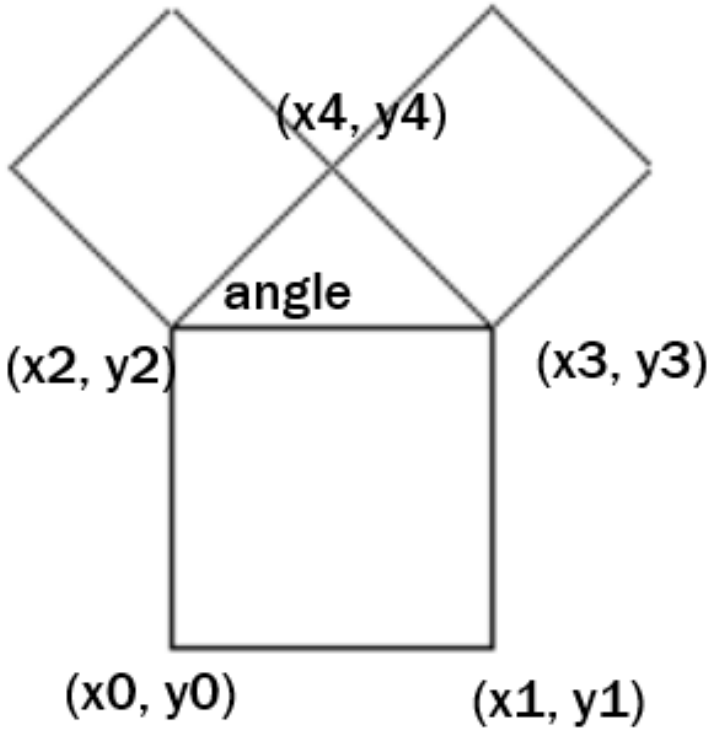


ОТРИСОВКА ПРАВОЙ ВЕТВИ

Нижняя левая точка для правого ветви совпадает с (x4,y4).

Длина нового правого дочернего квадрата равна *right_length = length * sin(angle)*.

base_angle для правого квадрата *base_angle родительского - 90` + angle*



2 ЭТАП

На втором этапе мы:

1. Рассмотрели разновидности Дерева Пифагора (классическое, обнаженное, обдуваемое ветром).
2. Выбрали вид дерева для моделирования (классическое).
3. Разобрались с этапами моделирования: задание точек, алгоритм прохождения итераций, отрисовка левой и правой ветви.
4. Написали уравнения, которые будут использованы в коде.

ФУНКЦИЯ TREE И ЕЕ ВХОДНЫЕ ДАННЫЕ TREE(C, N, X0, Y0, LENGTH, BASE_ANGLE, ANGLE)

C – полотно для рисования

N – количество итераций

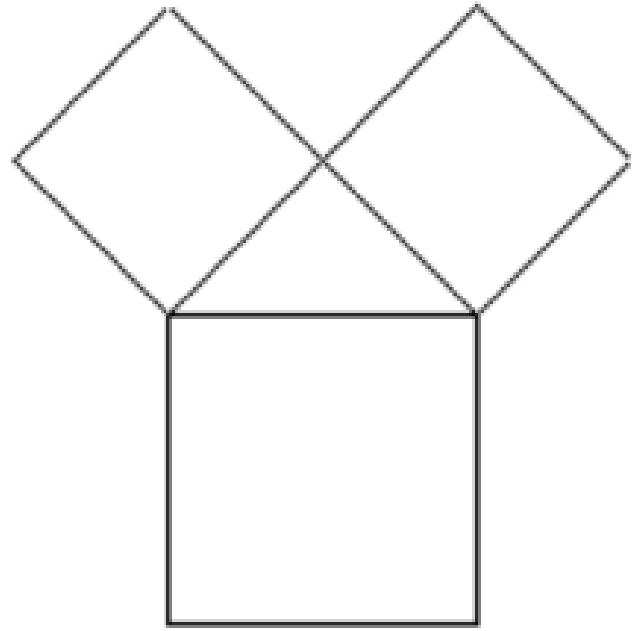
X0 – координата нижней левой точки по оси X для первого квадрата

Y0 - координата нижней левой точки по оси Y для первого квадрата

Length – длина стороны первого квадрата

base_angle – базовый угол наклона

Angle – угол наклона левой ветви

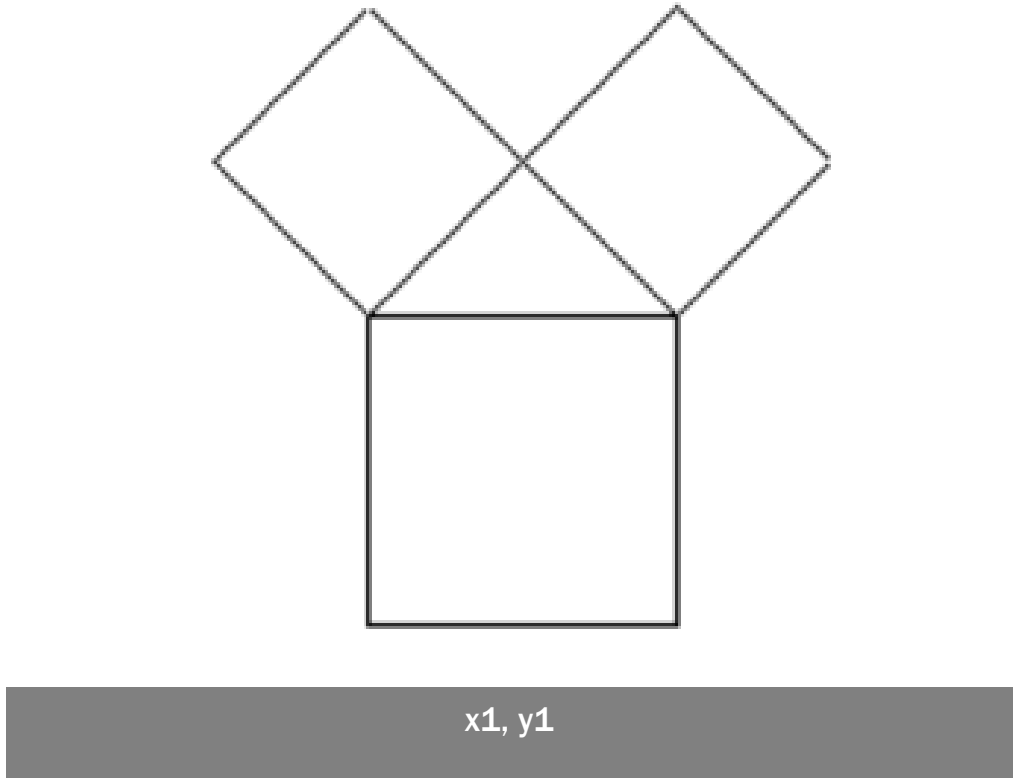


x0, y0

(X1, Y1)

x1 = x0 + (int)(length * cos(base_angle))

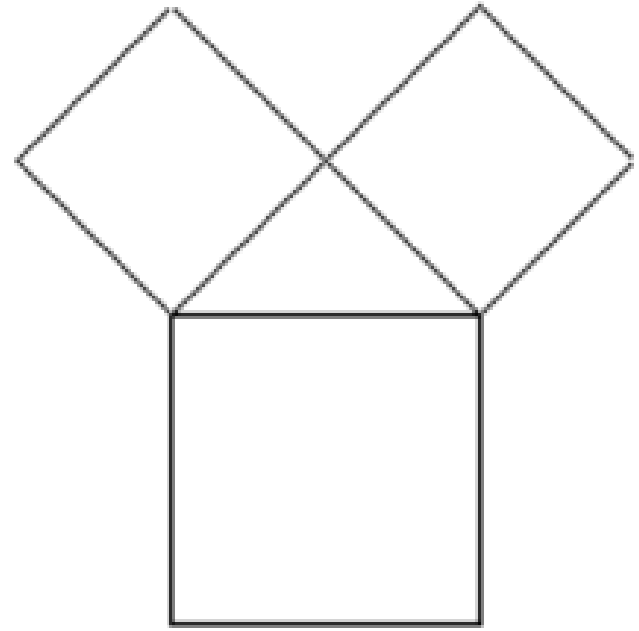
y1 = y0 - (int)(length * sin(base_angle))



(X2, Y2)

x2 = x0 + (int)(length * cos(base_angle + pi / 2))

y2 = y0 - (int)(length * sin(base_angle + pi / 2))

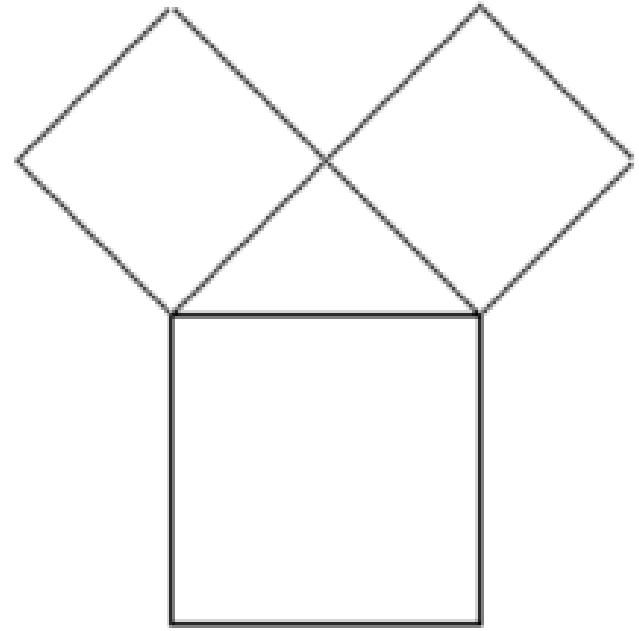


x2, y2

(X3, Y3)

x3 = x1 + (int)(length * cos(base_angle + pi / 2))

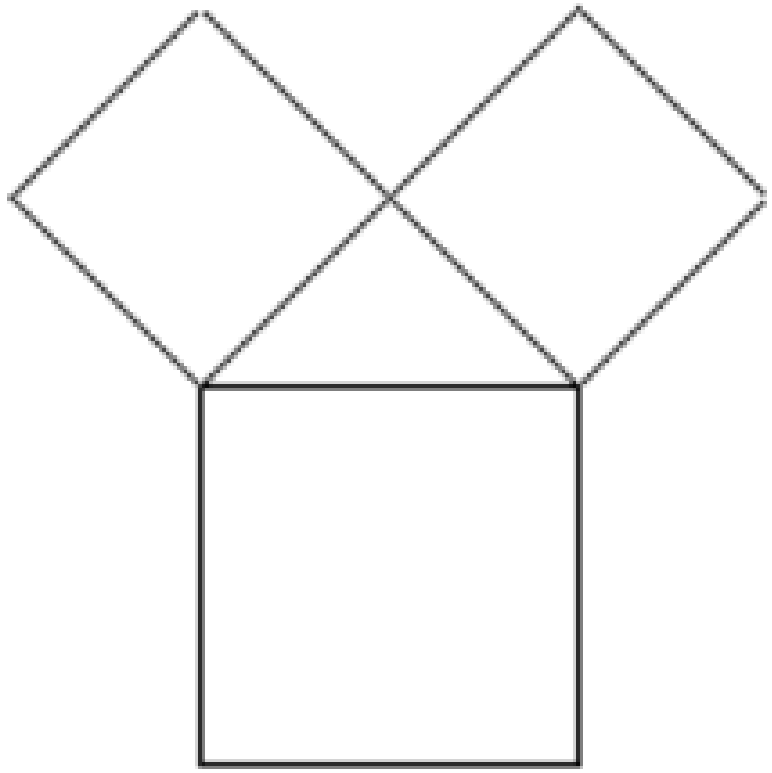
y3 = y1 - (int)(length * sin(base_angle + pi / 2))



x3, y3

(X4, Y4)

```
x4 = x2 + (int)(length *
cos(angle) * cos(base_angle
+ angle))

y4 = y2 - (int)(length *
cos(angle) * sin(base_angle
+ angle))
```



СОЕДИНЯЕМ ТОЧКИ ЛИНИЯМИ

```
c.create_line(x0, y0, x1, y1)

c.create_line(x0, y0, x2, y2)

c.create_line(x2, y2, x3, y3)

c.create_line(x1, y1, x3, y3)
```

IF N > 0:

ОТРИСОВКА ЛЕВОЙ ВЕТВИ

```
left_base_angle = base_angle + angle

left_length = length * cos(angle)

tree(c, n - 1, x2, y2, left_length, left_base_angle, angle)
```

ОТРИСОВКА ПРАВОЙ ВЕТВИ

```
right_base_angle = base_angle - pi / 2 + angle

right_length = length * sin(angle)

tree(c, n - 1, x4, y4, right_length, right_base_angle, angle)
```

ИНИЦИАЛИЗАЦИЯ ПОЛОТНА ДЛЯ РИСОВАНИЯ

```
root = Tk()
c = Canvas(root, width=700, height=700, bg='white')
c.pack()
```

ВЫЗОВ ФУНКЦИИ ДЛЯ ОТРИСОВКИ ДЕРЕВА ПИФАГОРА

```
tree(c, 10, 300, 500, 100, 0, pi / 4)

root.mainloop()
```

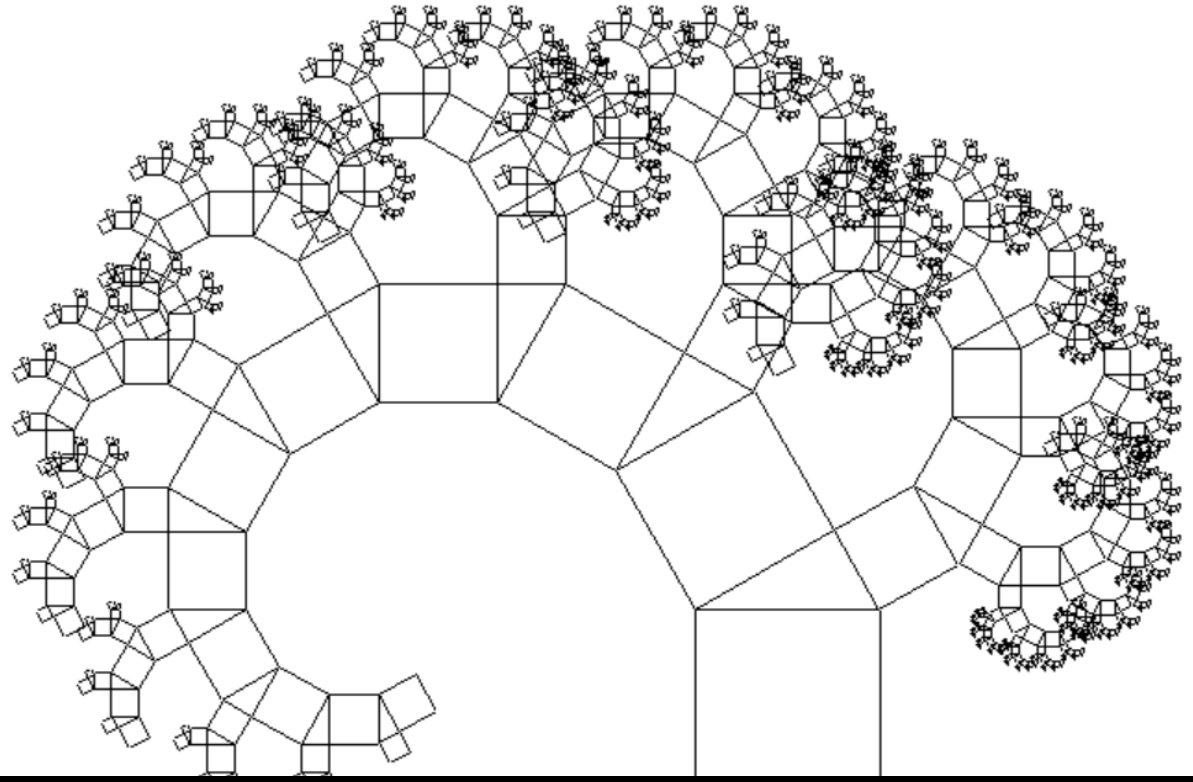
КОД ПРОГРАММЫ

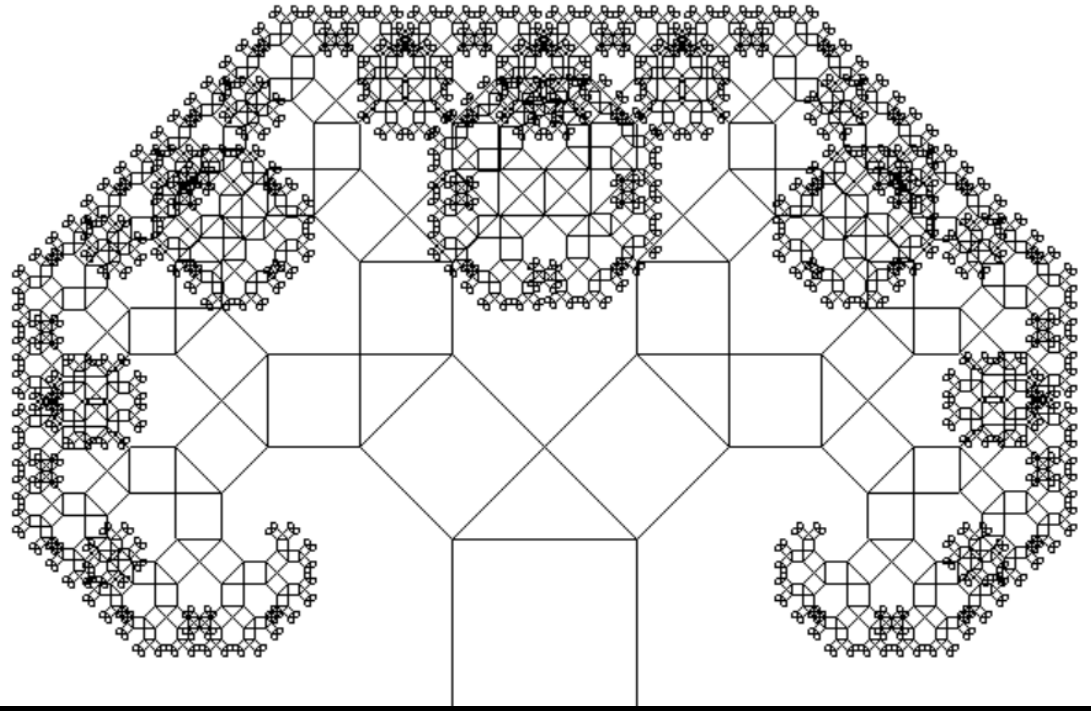
In [1]:
```
from tkinter import *
from numpy import pi, cos, sin
```

In [2]:
```
def tree(c, n, x0, y0, length, base_angle, angle):
    x1 = x0 + (int)(length * cos(base_angle))
    y1 = y0 - (int)(length * sin(base_angle))
    x2 = x0 + (int)(length * cos(base_angle + pi / 2))
    y2 = y0 - (int)(length * sin(base_angle + pi / 2))
    x3 = x1 + (int)(length * cos(base_angle + pi / 2))
    y3 = y1 - (int)(length * sin(base_angle + pi / 2))
    x4 = x2 + (int)(length * cos(angle) * cos(base_angle + angle))
    y4 = y2 - (int)(length * cos(angle) * sin(base_angle + angle))

    c.create_line(x0, y0, x1, y1)
    c.create_line(x0, y0, x2, y2)
    c.create_line(x2, y2, x3, y3)
    c.create_line(x1, y1, x3, y3)

    if n > 0:
        left_base_angle = base_angle + angle
        left_length = length * cos(angle)

        tree(c, n - 1, x2, y2, left_length, left_base_angle, angle)

        right_base_angle = base_angle - pi / 2 + angle
        right_length = length * sin(angle)

        tree(c, n - 1, x4, y4, right_length, right_base_angle, angle)
```

In [3]:
```
root = Tk()
c = Canvas(root, width=700, height=700, bg='white')
c.pack()
```

In [4]:
```
tree(c, 12, 300, 500, 100, 0, pi / 4)
root.mainloop()
#Kernel->Reastart & RunAll
```





УНИВЕРСАЛЬНОСТЬ ПРОГРАММЫ

3 ЭТАП

На данном этапе мы создали универсальную программу для отрисовки дерева Пифагора - с помощью неё мы можем построить любую разновидность дерева, изменив значения углов, - а также подробно описали все составные части кода программы.

Закончили смысловую часть проекта.

РЕЗУЛЬТАТЫ ПРОЕКТА

Выполнили проектную работу по созданию программного кода на языке программирования Python для моделирования дерева Пифагора.

Успешно завершили все этапы группового проекта.

Получили рабочую универсальную программу.