# Question 3

# Recursive function to generate a geometric Fractal pattern using Python's turtle graphics (Code Explanation)

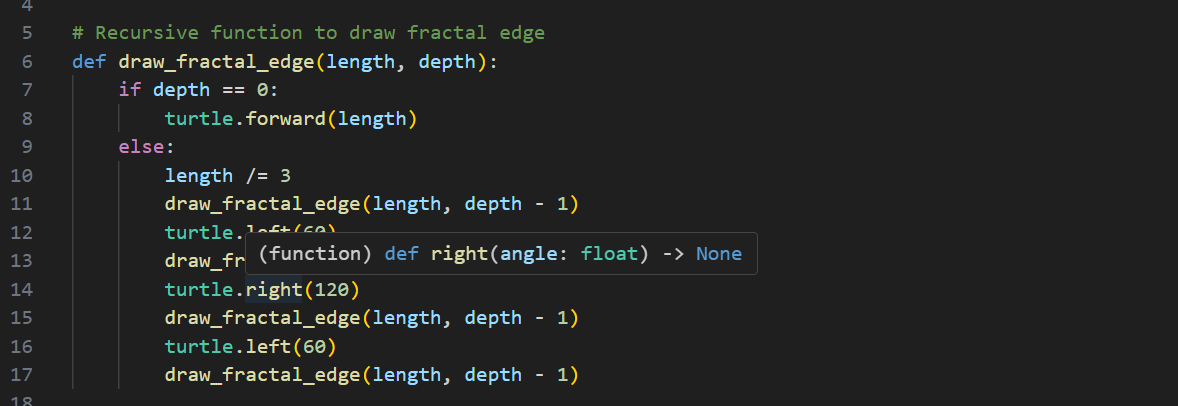
This is a program that creates geometric patterns using a Recursive function and Python’s turtle graphics. According to the question and the scenario pattern starts with a regular polygon or a straight line, and we recursively modify each edge of the polygon to create intricated designs. Below, we have explained each code in detail that we used to generate the pattern to make it more understandable for the viewers.

1. First, we import the turtle Module with the following code.



This will load the built-in turtle graphics library, which will enable a virtual pen that we will be using to draw shapes and patterns.

1. We use a recursive function to draw the fractal edges



A recursive function is the core and important factor in generating a pattern. Here, depth==0 means a simple line without any fractal detail. When depth is 0 it just draws a straight line with the given length.

Recursive case:

* First else function divides the polygon into three equal parts.
* The second function modifies the middle part of the divided line into an equilateral triangle with only two sides pointing inward.
* With the third function, the straight edges are divided into four smaller segments.
* With depth-1, each segment is recursively processed, generating more detail at each level.

By using a recursive function, we will be able to create the zigzag seen in the output image.

1. Users' Input

A screen shot of a computer program

AI-generated content may be incorrect.

This is a simple command that prompts the users to enter the number of sides, the length of a side, and the depth of the fractal pattern.

The number of sides will determine how the shape of the fractal pattern will look, and the depth will determine how many times the indentation is applied.

1. Setting up Turtle

A screen shot of a computer

AI-generated content may be incorrect.

This is a basic prompt to set up the turtle. Here, as you can see, we have set up drawing speed, background colour, and the line colour.

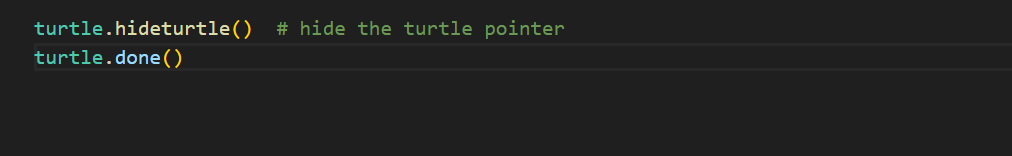
1. Drawing a polygon

A black rectangle with white text

AI-generated content may be incorrect.

This for loop will make sure that the loop is applied through each side of the polygon with a defined length of side and depth, and rotates the turtle by 360 so, so it perfectly connects with the start point of the polygon.

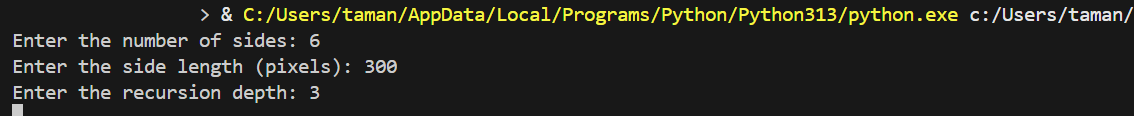
1. Ending the command prompt by hiding the turtle pointer



These prompts will hide the turtle cursor and end the turtle session, leaving a clean and finished fractal pattern on the screen.

# Output: using different numbers of sides, length, and depth

1.



A blue star shaped border

AI-generated content may be incorrect.

2.A screen shot of a computer

AI-generated content may be incorrect.

A blue and white snowflake

AI-generated content may be incorrect.

# To conclude

This Python program creates a fractal design using recursion and turtle graphics, where each of the regular lines or polygons is split into three equal parts, and the edges are segmented into four smaller parts, and the middle being turned into two sides of equilateral triangles pointing inward. The symmetrical fractal depends on the user input, the number of sides, the length of the sides, and the depth level, as each depth level designs a more complex indentation.

All the commands are detailed and explained above with the intention of making it simple to understand.