Module 6: Mathematical Sequences

JHU EP 606.206 - Introduction to Programming Using Python

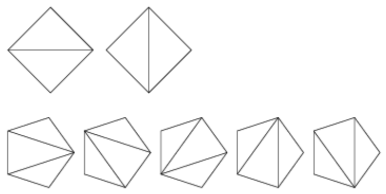
Introduction

In this assignment you will use functions to generate famous mathematical number sequences, namely Catalan numbers and the Pascal Triangle. Rather than writing a single, monolithic piece of code, you'll create a function that can then be called by your main method to accept a parameter(s) and produce the proper results.

Skills: logical operators, loops, functions, recursion

Catalan Numbers

Catalan numbers are used to answer the question: “In how many ways can a regular p-gon be divided into p-2 triangles if different orientations are counted separately?” So, for example, how many ways can we divide a square into 2 triangle or a pentagon into 3 triangles? Shown below are the order 2 and order 3 Catalan numbers for a square and a pentagon respectively:



They are generated by the following formula (where the capital pi symbol is the multiplication equivalent of a capital sigma for addition). n in this case refers to the “order”, so n = 2 for a square, n = 3 for a pentagon, and so on:

$$\prod_{k=2}^{n} \frac{n+k}{k} \qquad \text{for } n \geq 0.$$

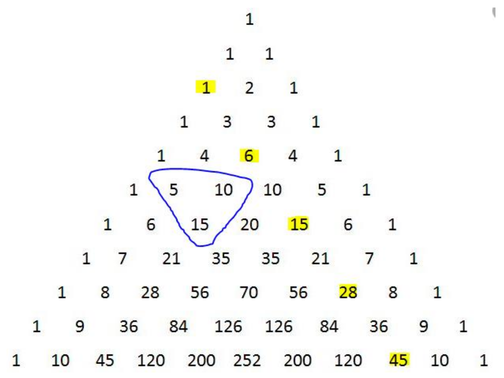
For a square n = 2, so this equation gives us: (2 + 2)/2 = 4/2 = 2

For a pentagon n = 3, so this equation gives us: [(3 + 2)/2 * (3 + 3)/3] = (5/2) * 2 = 10/2 = 5

Pascal’s Triangle

Pascal’s Triangle is a triangular array of binomial coefficients. There are a number of interesting characteristics:

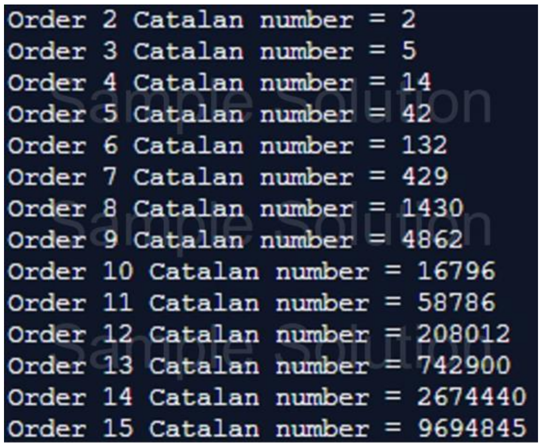
1. The outer left (and right) diagonal consists of all 1’s
2. The second diagonal is formed by adding 1 to the previous digit (counting numbers)
3. The third diagonal consists of triangular numbers
4. The sum of elements in each row forms powers of 2: 1, 2, 4, 8, 16,…
5. The Fibonacci sequence is present in the triangle too!



You may also notice that each element in the triangle is formed by adding together the two adjacent values in the row above (show in blue). We will be using this fact to recursively build a Pascal Triangle.

Programming Assignment Part 1: Catalan Numbers

Create a file called ***catalan.py*** that has a ‘main’ method that calls a function named ‘*catalan*’ which takes an integer argument p (the number of sides of a polygon) and returns the Catalan number of order p-2 using the formula above. Your code only needs to compute the 14 Catalan numbers show below. Recall that for n (order) = 2, p (sides) = 4, for n (order) = 3, p (sides) = 5, and so on. Your ‘main’ method should call your function each time through the loop and output the results to the console in the following format:

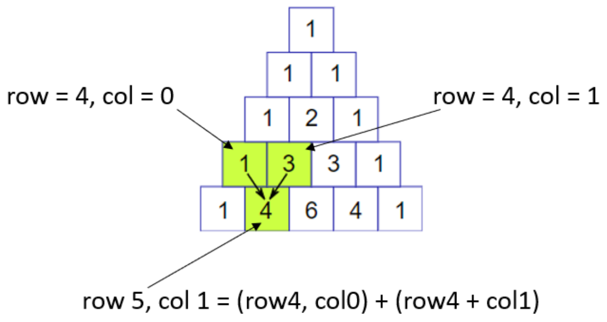


```
Order 2 Catalan number = 2
Order 3 Catalan number = 5
Order 4 Catalan number = 14
Order 5 Catalan number = 42
Order 6 Catalan number = 132
Order 7 Catalan number = 429
Order 8 Catalan number = 1430
Order 9 Catalan number = 4862
Order 10 Catalan number = 16796
Order 11 Catalan number = 58786
Order 12 Catalan number = 208012
Order 13 Catalan number = 742900
Order 14 Catalan number = 2674440
Order 15 Catalan number = 9694845
```

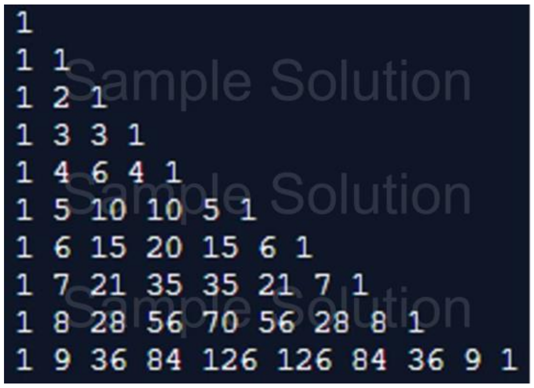
Programming Assignment Part 2: Pascal's Triangle

Create a file named ***pascal.py*** that uses a 'main' method to call a recursive function named 'pascal' which accepts an integer n as a parameter and prints the order n Pascal Triangle to the console. **For this assignment you will generate the Pascal Triangle for n = 10 (the first 10 rows of the theoretically infinite triangle).**

As a hint, your function may choose return a 1 if the current position is at the beginning or end of a row, otherwise it will recursively return the sum of the two adjacent entries above the current position in the triangle. Here's a diagram to help demonstrate the relation between the current element and the adjacent elements above it:



Please note that your 'pascal' function MUST be recursive (it must call itself). There are approaches to this that are iterative, but that's not what's being asked for here. Your final output should resemble a triangle and have each element separated by a space exactly as shown below:



```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
```

Deliverables

readme.txt

So-called “read me” files are a common way for developers to leave high-level notes about their applications. Here’s an example of a README file for the Apache Spark project. They usually contain details about required software versions, installation instructions, contact information, etc. For our purposes, your readme.txt file will be a way for you to describe the approach you took to complete the assignment so that, in the event you may not quite get your solution working correctly, we can still award credit based on what you were trying to do. Think of it as the verbalization of what your code does (or is supposed to do). Your readme.txt file should contain the following:

1. **Name**: Your name and JHED ID
2. **Module Info**: The Module name/number along with the title of the assignment and its due date
3. **Approach**: a detailed description of the approach you implemented to solving the assignment. Be as specific as possible. If you are sorting a list of 2D points in a plane, describe the class you used to represent a point, the data structures you used to store them, and the algorithm you used to sort them, for example. The more descriptive you are, the more credit we can award in the event your solution doesn’t fully work.
4. **Known Bugs**: describe the areas, if any, where your code has any known bugs. If you’re asked to write a function to do a computation but you know your function returns an incorrect result, this should be noted here. Please also state how you would go about fixing the bug. If your code produces results correctly you do not have to include this section.

Please submit your **catalan.py** and ***pascal.py*** source code files along with a PDF file containing screenshots of your output in a PDF called JHEDID_mod6.pdf (ex: jkovba1_mod6.pdf). Please do not ZIP your files together.

Recap:

1. readme.txt
2. catalan.py
3. pascal.py
4. 2 screenshots of your outputs from catalan.py and pascal.py

Please let us know if you have any questions via Teams or email!