Lab Instructions

Objective: At the end of this lab, you will be able to define some simple recursive functions in Python.

In class exercises

You should have time to finish at least the following exercises in class

1. In mathematics, the harmonic series is the divergent infinite series:

   $$\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots.$$

   Its name derives from the concept of overtones, or harmonics in music.

   Create a function that, given a precision parameter, returns the value of the harmonic series.

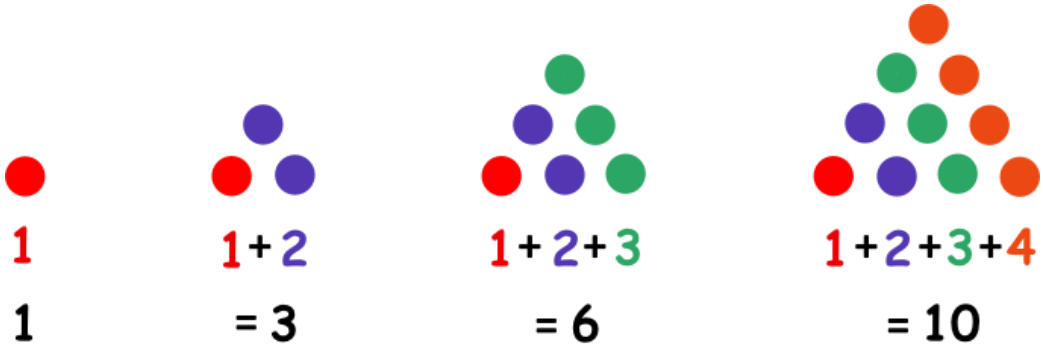
   ### Examples

   ```
   harmonic(3) → 1.833

   harmonic(1) → 1.0

   harmonic(5) → 2.283
   ```

2. Write a function **print_triangular_numbers(n)** that prints out the first n triangular numbers (n is a positive integer). A call to **print_triangular_numbers(5)** would produce the following output:

   ```
   1       1
   2       3
   3       6
   4       10
   5       15
   ```

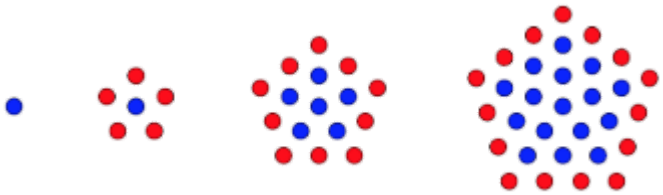
   (Hint: search in Google to find out what a triangular number is)



Then apply divide-and-conquer strategy to write a program that asks user for a value of n (n is a positive integer) then calls the function with that value to print out the first n triangular numbers.

3. Write a function that takes a positive integer and calculates how many dots exist in a pentagonal shape around the center dot on the *Nth*

   In the image below you can see the first iteration is only a single dot. On the second, there are 6 dots. On the third, there are 16 dots, and on the fourth there are 31 dots.



Homework

Revisit the Collatz sequence (see Challenge Exercise in Lab 2)

The Collatz sequence is as follows:

- Start with some given integer .
- If it is even, the next number will be divided by 2.
- If it is odd, multiply it by 3 and add 1 to make the next number.
- The sequence stops when it reaches 1.

According to the Collatz conjecture, it will always reach 1. If that's true, you can construct a finite sequence following the aforementioned method for any given integer.

Write a RECURSIVE function that takes in an integer and returns the **highest integer** in the corresponding Collatz sequence.

Examples

```
max_collatz(10) → 16
# Collatz sequence: 10, 5, 16, 8, 4, 2, 1

max_collatz(32) → 32
# Collatz sequence: 32, 16, 8, 4, 2, 1

max_collatz(85) → 256
# Collatz sequence: 85, 256, 128, 64, 32, 16, 8, 4, 2, 1
```