Objectives:

Learn how to perform analysis

Setup:

All files needed for this lab were created by doing the first task in lab 0. If you didn't do lab 0, do the first task to create your lab repository.
Unless otherwise stated, all writing goes into the file lab2.

Part A Analysis

Follow this guide on how to perform an analysis:
https://seneca-ictoer.github.io/data-structures-and-algorithms/B-Algorithms-Analysis/how-to-do-an-analysis
Perform an analysis of the following functions.

Function 1:

Analyze the following function with respect to number

```
def function1(number):
    total = 0

    for i in range(number):
        x = i + 1
        total += x * x

    return total
```

Function 2:

Analyze the following function with respect to number

```
def function2(number):
    return (number * (number + 1) * (2 * number + 1)) // 6
```

Function 3:

Analyze the following with respect to the length of the list. Note that the function call len() which returns the length of the list is constant (O(1)) with respect to the length of the list.

```
def function3(list):
    n = len(list)
    for i in range(n - 1):
```

```
            for j in range(n - 1 - i):
                  if list[j] > list[j+1]:
                        tmp = list[j]
                        list[j] = list[j+1]
                        list[j + 1] = tmp
```

Function 4:

Analyze the following function with respect to number

```
def function4(number):
      total = 1
      for i in range(1, number):
            total *= i + 1
      return total
```

Part B Pre-Lab Preparation:

While not 100% required, doing this will make it easier to complete the lab during the lab class

Copy the functions sum_to_goal() and fibonacci() from your lab1/lab1.py to lab2/lab2.py. Only copy those two functions.

Part C In-Lab Discussion:

During your lab period, form groups of 3 students.
Your professors may have further instructions on how to form groups.
If you have not completed the pre-lab, do so now. copy the functions sum_to_goal() and fibonacci() from lab1/lab1.py to lab2/lab2.py. Only copy those two functions.
Doing so will trigger a series of tests, which will be timed.
In each group, look at the runs of your lab1 functions (using lab2's set of testers)
In the action tab, expand your successful run of the tester for the lab. You will find the timing results of your functions there.
in lab2.md discussion, fill in the names of your group members.

Example: List the members of your group member below:
* Name
* Dev Soni (Me)
* Ruslan Gofman

Fill in the Timing table in lab2.md with the times for your group, and add or remove rows as necessary based on the number of members in your group

Timing Data

Note, if a groupmate did not complete lab 1, simply put 0.0 in for the times; it is ok if something is missing.

Team member	Timing for fibonacci	Timing for sum_to_number
Dev Soni (Me)	5.659	0.003

| Jack Martin | 2.805 | 0.614 |
| Ruslan Gofman | 5.55 | 0.10 |

Fill in the second table with the following information:
slowest time for each of the listed function
fastest time for each of the listed function
difference in timing between slowest and fastest times

Compare the slowest and the fastest version of each function; what were the differences? Was it a difference in syntax? A difference in approach?... for example, did one solution use recursion while the other did not?### Summary
Example:
function	fastest	slowest	difference
sum_to_number	0.003	0.988	0.985
fibonacci	2.805	6.182	3.377

Reflection

Considering the solutions you saw in the lab 1 code, what differences did you see between the fastest and slowest versions?
Was there a difference in terms of the usage of space resources? Did one algorithm use more/less space (memory)?
What sort of conclusions can you draw based on your observations?

Submitting your lab

To get a mark for this lab, you must submit:

a complete analysis of every function in part A.

Please make sure you follow the steps listed below fully.

Place all your work for this lab into the file lab2.md unless otherwise indicated

When you are happy with the state of your files, submit a link to your repo's lab2 folder on Blackboard.
Note: Submitting a link to Blackboard indicates that your lab is ready for grading in the current state. You can only submit a link once you are ready to be graded.