# GGS 366: Lab 3

## General guidelines

**Please submit the .ipynb notebook containing all the codes.** Make sure the notebook is properly formatted by following the instructions below.

* Separate answers to each question using text cells and markdown headings. See example syntax here: [Markdown Guide - Colab](https://colab.research.google.com/notebooks/markdown_guide.ipynb)
* Write the essay questions in your Python notebook using text cells.
* The function body (where appropriate) must include a docstring to provide a general description of the function.
* Each step of the code needs to be commented.
* The code needs to be properly indented and readable. For more on formatting guidelines: [PEP 8 – Style Guide for Python Code | peps.python.org](https://peps.python.org/pep-0008/) and [PEP 257 – Docstring Conventions | peps.python.org](https://peps.python.org/pep-0257/)

You may use GenAI as a supporting tool. However, directly copying code from GenAI will be considered plagiarism and hinder your learning process. This can negatively impact your performance in course exams and, ultimately, your professional success. Therefore, it is in your best interest to thoroughly understand the fundamental concepts and make a genuine effort to solve the problems on your own.

## Question 1 (5 points)

Here’s a list: *integers = [100, 80, 60, 40, 20, 0]*

Print the list in reverse order using a ***for* loop**. The expected output is:

0

20

40

60

80

100

## Question 2 (5 points)

Here’s a list of numbers: *randoms = [10, -5, 20, 30, -15, 105, 50, -25, 60]*

Tasks:

1. Iterate through the list using a ***for* loop**.
2. **Skip negative numbers** (i.e., do not add them to the sum).
3. **Stop the loop** if a number greater than **100** is encountered.
4. **Calculate and print the sum** of the processed numbers (only positive numbers before encountering a number greater than 100).

## Question 3 (5 points)

Here’s a list of city names:

*cities = ["Austin", "New York", "Amsterdam", "Boston", "Atlanta", "Chicago", "Athens", "Denver", "Anchorage", "Los Angeles"]*

Use **list comprehension** to create a new list that only includes cities that start with the letter **"A"** and have **more than 6 characters**.

Here’s another list of people’s ages:

*ages = [12, 25, 17, 18, 22, 15, 30, 13]*

Use **list comprehension** to create a new list where each age is categorized as either **"Minor"** (under 18) or **"Adult"** (18 and over).

## Question 4 (10 points)

A **prime number** is a number that is only divisible by **1** and **itself**. For example, **7** is a prime number because its only divisors are **1** and **7**.

To determine whether a number **N** is prime, it must meet the following criteria:

1. **N must be greater than 1.**
2. **N should not be divisible by any number between 2 and** √N **(i.e., the remainder of N divided by any of these numbers should not be 0).**

**Tasks:**

1. **Write a function** that takes an integer as input and returns whether it is a prime number.
2. **Create an empty list** called prime\_numbers = [].
3. **Use a while loop** to test all numbers from **1 to 50** and check if they are prime. If a number is prime, append it to the prime\_numbers list.
4. **Print the list** of prime numbers.

## Question 5 (10 points)

Imagine we have defined a **rectangular area** using a **bounding box**. The bounding box is represented as:

*bounding\_box = [-77.528, 38.701, -77.364, 38.792]*

This corresponds to:  
**[longitude\_min, latitude\_min, longitude\_max, latitude\_max]**

To determine if a **geographic location** falls within this bounding box, we need to check:

* If the **longitude** of the location is within the range **longitude\_min ≤ longitude ≤ longitude\_max**
* If the **latitude** of the location is within the range **latitude\_min ≤ latitude ≤ latitude\_max**

**Tasks:**

1. **Write a function** *is\_within\_bbox(longitude, latitude, bbox)* that checks whether a given geographic location falls within the bounding box.
2. **Here’s a list of sample locations.**

*locations = [ (-77.45, 38.75), (-77.52, 38.71), (-77.36, 38.80), (-77.53, 38.79), (-77.40, 38.69), (-77.41, 38.80)]*

1. **Use a for loop** to iterate through the list and check if each location is inside the bounding box. Print an appropriate message (inside or outside) for each location.