CS2050: Computer science 2
Project 3 **Same-Number Subsequence**
Total Points (50 pts)
Due: March 13, 2022 @ 11:59

Analysis of algorithms
Analysis of algorithms is the determination of the amount of time and space resources required to execute it. Usually, the efficiency or running time of an algorithm is stated as a function relating the input length to the number of steps, known as **time complexity**, or volume of memory, known as **space complexity**.

However, the main concern of analysis of algorithms is the required time or performance. Generally, we perform the following types of analysis −

- **Worst-case** − The maximum number of steps taken on any instance of size **n**.
- **Best-case** − The minimum number of steps taken on any instance of size **n**.
- **Average case** − An average number of steps taken on any instance of size **n**.

Program Testing
Testing a program is simply checking by means of actual execution whether a program behaves in the desired manner. The program is executed and supplied with **different test data**, and the way in which the program responds to this test data **is analyzed**.

Problem Description

Write an O(n) program that prompts the user to enter a sequence of integers ending with 0 and finds longest subsequence with the same number. Here is a sample run of the program:

<Output Sample>
```
Enter a series of numbers ending with 0: 2 4 4 8 8 8 8 2 4 4 0
The longest same number sequence starts at index 3 with 4 values of 8
```
<End Output Sample>

Tasks:

1. [10 pts] Design Algorithm: (Describe how your algorithm works)

2. [15 pts] Analysis of the Algorithm: (Describe how you analyze the algorithm – read the algorithm analysis paragraphs above)

3. [15 pts] Coding: (submit a working .java program to canvas), name the program Project_03, your submission must include screenshot.

4. [10 pts] Testing: (Describe how you test your program – read the Program Testing paragraph above)

5. Fill in self-evaluation:
   a. Can your program find the largest same number sequence? _____
   b. Can your program find the correct start index of the largest same number sequence? _____
   c. Is your algorithm O(n)? _____

Submissions: Upload three files to canvas as listed below:

- Document file contains your descriptions for the points 1, 2, 4, and 5 as stated in **Tasks** above.
- Submit the complete implementation of Project_03.java.
- Screenshot showing a result by entering the list `2 4 4 8 8 8 8 2 4 4 0` as shown in the sample output above.

Grading:

- Task 1 should be an algorithm, missing an algorithm is a minimum of 3 points off.
- Task 2 must include the analysis of two cases, 5 points each.
- Tasks 3 and 5 with the screenshot worth 15 points
- Tasks 3 and 5 without screenshot 15 points off
- Task 4 must show testing results using different data. **Full credit** is given when testing with positive, negative values and with non-numeric values, student should tell how the code behaves in each list entries and why?