APS 105 — Computer Fundamentals

Lab #5: Functions and Arrays

Winter 2022

You must use `examify.ca` to electronically submit your solution by 11:59 pm on **Saturday, February 19, 2022**. Please note that in this lab, you are only asked to implement and submit functions. However, to test your functions, you will need to write a complete `C` program.

Objective

The goal of this laboratory is to solve complex problem involving the usage of functions and arrays.

Preparation

Read through this entire document carefully, and do the work to create the programs that are described below. You are encouraged to:

- Read the class bulletin board on Piazza, to see if others had similar problems. If you do not see anything helpful there, ask a question. Do not ask for, or ever give a full or partial solution to the lab assignment.
- During the lab, ask for assistance from the lab TA assigned to you in your lab subsection.

Start and follow the suggested steps below:

- Write your pseudocode that is in a plain language describing and planning the steps in solving the problem at hand.
- Follow up your plan by writing a code on paper, i.e., stay away from typing on your computer at this time.
- Try to review your code and identify any mistakes that you can identify, i.e., walk through your code.
- Use your computer, type and run your code.
- Run the test cases provided to ensure proper functionality of your code.
- Think if there are corner case by developing yourself difficult examples. Solve the examples you developed by hand and check if your code is giving the expected outcome.

Part 1 — Median of two sorted arrays

Implement a `C` function, named `median`, that receives two arrays as parameters passed and returns the median of the values stored in these two arrays. The two arrays are **sorted**.
The prototype of the function **MUST** be:

```
double median(int a[], int b[], int sizeA, int sizeB);
```

For example, if I have `int a[] = {1, 2, 10, 12, 100};` and `int b[] = {3, 11, 20, 500, 600};`, then the median is $\frac{11+12}{2} = 11.5$.

Part 2 — Longest sequence of increasing numbers

In this part, you are asked to write a C function that finds the longest sequence of increasing numbers in an array. You should implement a function that receives an array passed as a parameter, and prints the longest sequence of increasing numbers.
The prototype of the function **MUST** be:

```
void longestSequence(int a[], int sizeA);
```

For example, we have an int array with values as $1, 3, 10, 1, 7, 8, 10, 30, 0$. Your code should identify the longest sequence of increasing numbers and print

```
Longest sequence is 1, 7, 8, 10, 30.
```

Array size cannot be 0.

Part 3 — Longest Common Sequence

This part requires that you implement a C function that takes in as parameters two arrays and their sizes. This function is required to find the longest common sequence of numbers in these arrays.
The prototype of the function **MUST** be:

```
void longestMutualSequence(int firstArr[], int secondArr[], int sizeA, int sizeB);
```

For example, if you have `int a[] = {1, 2, 3, 4, 1000, 90, 100, 110, 200, 202};` and `int b[] = {1, 1, 2, 3, 90, 100, 110, 200, 202, 600};`, your function should output.

```
Longest Common Sequence is 90, 100, 110, 200, 202.
```

If there are two common sequences of the same size, your function should print out the first common sequence found.

Marking

This lab will be marked out of 10, with 3 marks allocated to each part. 1 mark will be assigned as an evaluation of your coding style (introduced and discussed at length in Plenary Lecture 2). The coding style will be marked in your lab section by your lab TA. Here are the basic requirements for your code to earn the coding style marks:

- Use meaningful identifiers for the variables in your C function. Use either snake case or camel case, but be consistent throughout your entire C program.
- Use named constants when needed, rather than literal constants.
- Use consistent indentation throughout your entire C program.
- Use comments at the beginning of your program before the `main()` function. You may also add comments throughout your program as necessary, but they are not strictly enforced.
- Define and call functions as required in this laboratory.

The automarker may use multiple test cases for each part, and if this is the case, the marks for this part will be evenly split among the test cases. The test cases used for marking may or may not be the same as the test cases that are made available to you. The deadline of **(11:59 p.m. on Saturday, February 19, 2022)** is strictly enforced, so avoid last minute submissions.

Stay Safe and Good Luck!