Assignment #1B - C Programming Basics

CMPSC 311 - Introduction to Systems Programming

Fall 2024

Prof. Shagufta Mehnaz

Due Date:09/27/2024,

In this assignment, you will write simple functions in C. The purpose of this assignment is to assess your basic programming skills. You should have no difficulty in completing this assignment. If you experience difficulty, you should take some time to brush up on programming.

Like all assignments in this class, you are prohibited from copying any content from the Internet or discussing, sharing ideas, code, configuration, text, or anything else or getting help from anyone in or outside the class. Failure to abide by this requirement will result in dismissal from the class, as described in our course syllabus.

Below are the files in this assignment and their descriptions

1. student.h: a header file with the declarations of the functions you will implement.

2. student.c: a source file in which you will implement the functions whose declarations appear in student.h. In other words, you will provide the definitions of the functions declared in student h. (Your changes are required only in this file)
3. reference.h: a header file with the declarations of the functions identical to those defined in student.h except they are prefixed with reference. These are the reference functions against which your implementations will be tested.
4. reference.o: an object file that contains the reference implementations of the functions declared in reference.h. This is a binary file that contains compiled reference implementations of functions. 5. tester.c: unit tests for the functions that you will implement. Each unit test passes input to your implementation of a function and to the reference implementation of the same function and compares the outputs. This file will compile into an executable, tester.o, which you will run to see if you pass the unit tests.
6. Makefile Makefile: instructions for compiling and building tester used by the make utility

Note: Although you only need to edit student.c to complete the assignment successfully, you can modify any file you want if it helps you in some way. When testing your submission, however, we will use the original forms of all files except student.c. So make sure you revert all the changes and still check all your tester pass. Do not forget to add comments to your code to explain your implementation choices.

Below are the functions you need to implement:

1. Swap - Takes a pointer to two integers and swaps the values of integers. (1 point)

2. Absolute Number - Takes an array of integers and applies the absolute value function to each integer (1 point).

3. N-th Tribonacci Number - Takes an integer n as input and returns the nth Tribonacci number. The nth term in the Tribonacci Series is defined as

T(0) = 0, T(1) = 1 , T(2) = 1
T(n) = T(n - 1) + T(n - 2) + T(n-3), for n > 1. (1 point).

4. Mean - Takes an array and the length of the array as input and calculates the mean of the array. (Length of array is at least 1) (1 point)

5. Insertion Sort - Takes an array and their length as input and sort the array using Insertion sort in increasing order (1 point)

6. Reverse a number - Takes an integer n as input and returns the reverse of the number. For example, if 371 is passed to the function as an argument, it should return 173. In this case, since the digits of the number were 3, 7, and 1, reversing the digits would change the order to 1, 7, and 3. For the sake of simplicity of the assignment, no number would be given having trailing or leading zeroes, or zeroes present anywhere in the number. (1 points)

7. Palindrome - Takes a string (character array) and the string length as input. Returns 1 if the given string is a Palindrome and returns 0 otherwise. If the string “hello” is passed to the function, it should return 0 since the given string is not a palindrome (the reverse of this string is not equal to the original string : “olleh” = “hello”). But if i pass in string “racecar” , it should return 1 since the given string is a palindrome (2 points)

You are encouraged to write helper functions to simplify the implementations of the above functions. You should, however, not use a library function and implement all helper functions yourself.

The sample test cases present in the tester.c are only for your reference in this assignment. During grading, your program will be evaluated against multiple test cases. Each test case that passes will carry 0.2 points. Each 1 point question will have 5 test cases and each 2 point question will have 10 test cases.

To start working on the assignment, start by implementing any of the above questions in student.c and execute the following commands in the project directory to build and test your implementation. Do not forget to build the project after you have made any changes.

```
make clean
make
./tester
```