Homework 4 -- Out: March 14, Monday -- DUE: March 28, Monday, 2:29pm

EC327 Introduction to Software Engineering – Spring 2022

Total: 100 points

Submission: **Failure to follow any of these guidelines **WILL** result in loss of points on the assignment.*

You will use Github Classroom to submit your solution. You can submit as many times as you want until the deadline without incurring in any penalty.

1. Accept the assignment "HW4" in the EC327 github classroom using this link: https://classroom.github.com/a/tXxyVktL and clone the repository on your own local machine.
2. Complete the problems in this assignment, saving your solution for each question in a separate file, named Q1.cpp, Q2.cpp, Student.cpp, Student.h
3. Stage (add the files)
4. Commit your changes
5. *Push your changes to the repository.*

Late Homework Submissions:

Recall the late submission policy and the **fixed penalty of 20%** for submitting up to three days after the deadline. Also recall that we only grade your latest submission (e.g., if you submit both on time and after the deadline, only your late submission is graded).

** You may use any development environment you wish, as long as it is ANSI C++ compatible.* ***Make sure your code compiles and runs properly under the Linux environment on the PHO305/307 (or eng-grid) machines before submitting. IF YOUR CODE DOESN'T COMPILE, YOU WON'T GET ANY CREDIT!***

** Comment your code and use a clean, easily understandable coding convention. A coding style guide is at this link, but many others exist online:*
https://google.github.io/styleguide/cppguide.html#Comments (see "Comments")

**** Please run the following command in terminal BEFORE you try compiling your code (without the quotation marks): "*module load gcc*".   If you run "*gcc --version*" you should see that your compiler version is 5.2.x. This will allow you to use the "--std=c++17" flag when compiling your code with g++.***

IMPORTANT NOTE: YOUR CODE WILL GO THROUGH AN AUTOGRADER SO MAKE SURE YOUR CODE TAKES THE INPUTS AS THE ABOVE EXAMPLE AND ITS OUTPUTS ARE IN THE SAME FORMAT (SPACES, COMMAS, etc.).

Q1. Multi dimensional arrays. [25 points]

Given a NxN matrix of floats, write a function `getNorm` that returns its 1-norm, defined as:

$$\|A\|_1 = \max_{1 \leq j \leq n} \left(\sum_{i=1}^{n} |a_{ij}| \right)$$

The prototype of `getNorm` should be

```
float getNorm(float **matrix, int N);
```

You can use any libraries that are included in the C++ installation on the lab machines, for example cmath. You are given a header file Q1.h and a sample main file Q1main.cpp. You can use them to test your code, which you should include in a file named Q1.cpp.

Files to submit: Q1.cpp

Q2. Dynamic memory allocation. [20 points]

Write a function `reverseString` that, given a C++ string, returns a pointer to a new string whose content is the reverse of the original one. The prototype of `reverseString` is as follows:

```
 string* reverseString(string &s);
```

For example, when given a string with “EC327” as a value, reverseString will return a new string containing “723CE”.

You are given a header file Q2.h and a main file Q2main.cpp. Implement your function in a file called Q2.cpp and submit it.

Files to submit: Q2.cpp

Q3 *Class design.* [55 points]

a) Design a class named **Student** that contains:

- A string data field named `first`, which holds the student’s first name
- A string data field named `last`, which holds the student’s last name
- An unsigned integer data field `id`, which holds the student’s ID
- A float gpa, which holds the student’s GPA

`first`, `last`, and `id` should be declared as public members, while gpa should be made private.

- A no argument constructor, which sets “Jane” as `first`, “Doe” as `last`, 0 as `id`, and 0.0 as `gpa`.

- A constructor that takes four arguments and assigns their values to `first`, `last`, `id`, and `gpa` respectively. The prototype of this constructor should be:
  - `Student(string f, string l, int i, float g);`

- Get and set functions to handle the private data field `gpa`, as follows:
  - `void setGpa(float g); //assigns the value g to gpa`
  - `float getGpa(); //returns the gpa`

- A function display with the following prototype:
  - `void display();`

When executed, display should print the student's first name, last name, id, and GPA on a single line separated by a single space, as follows:

```
John Doe 3921093129 3.5
```

b) Separate the class declaration and implementation into `Student.h` and `Student.cpp`. You can use #ifndef / # define / #endif to prevent multiple class declarations.

We provide a Q3main.cpp file to test your implementation.

Files to submit: Student.cpp, Student.h

IMPORTANT NOTE: YOUR CODE WILL GO THROUGH AN AUTOGRADER SO MAKE SURE YOUR CODE TAKES THE INPUTS AS IN THE ABOVE EXAMPLES AND ITS OUTPUT IS IN THE GIVEN FORMATS. CODE THAT DOESN'T COMPILE ON THE LAB MACHINES WILL NOT BE GIVEN ANY CREDIT.