**CS2 Final Project Spring 2024**

**Prof. Karen Trovato**

**Objective:**

You will add, modify or extend a class such as University Member, Student or Course. You can have multiple classes in the same program. Please create a menu if you want to demonstrate multiple features.

For each class:

* I/O parsing a new file for new class with new functionality (e.g. including time)
* Inheritance
* Overloading (Any, including: +, << or >>)
* Friend function (does not double count if overloading << or >>)
* New (different) dynamically allocated objects
* Templates
* New object related to current UnivMember & Student with accessors and mutators.
  + Example: Course
* New application functionality

YOUR PROGRAM SHOULD PREFERABLY READ FROM A FILE & NOT REQUIRE ANY USER INPUT.

**Example:**

Extended Student to:

+1 Overload >> (demonstrated by reading from student.txt)

+1 Inherit from UnivMember

1. **Upload your code into the autograder**

**It is listed as 14: Final Project**

**You can make multiple submissions up until the deadline.**

Be sure to comment each file with your **username,** be sure to comment properly, and use proper syntax and style**.**

If you are on a Mac and have to copy a whole directory of files from storm to your Mac, you should consider using the -r flag to recursively copy an entire directory like this (all on one line).

**Examples & ideas:**

You may choose any number of these or define your own.

1. Derive Student and/or Profs from UniversityMember.
   1. Read from a file, reading and writing the resulting Student into a new output file from a vector of Students
   2. Overload << to write the Student object (and demonstrate it) +1
   3. Overload >> to read the Student object (and demonstrate it) +1
2. Read Courses (tab delimited, or you can generate your own from the Courses2023.xlsx file)
   1. Generate professor list along with IDs. Save to a file +1
   2. Show all courses for a given professor or student +1
      1. Create a useful schedule showing a grid of M-F and time blocks with Course# and section +2
   3. Read in: a student file with list of course CRNs
      1. Show all courses for a given student +2
      2. Determine if a student has courses that overlap in time, and output this error to an error file. +2
      3. A function that adds a student to a course. Consider adding by ID +2
      4. Give the roster of students for a particular course, not allowing over 30 students +2
   4. Read in: course listing
      1. Produce a list of courses that overlap based on their timeslot +1

Consider an overload for == indicating time overlap +1

1. IDs Read prof and student names.
   1. Extract Prof names from CoursesFall2023 and generate IDs.
      1. Save a file of Profs with their IDs +1
   2. Generate IDs for Prof and Student and store them into a file that could be subsequently read. +1
   3. Read the Prof & Student IDs, determining the largest value to set as the next available FID. +1
   4. Add new students, generating FIDs and appending to the existing student file. +1
2. Email username:   
   Generate **unique** usernames based on first initial and last name. If a username is already taken, then add an increasing numeral to it.
   1. Output the student name, username and ID to a second file +2

Example input file:

Smith, Mary

and later on there is

Smith, Michelle

Output to a file, maybe ‘existingStudents.txt’

Consider overloading the <<

It might look like:

Mary Smith,A0000001,msmith

Michelle Smith,A0000002,msmith2

(or msmith1 if you want - just as long as they are different. )

* 1. Create a way to read in the file above and set Student values. +2

Here, each student might be read in and the ID would be set explicitly.

Consider overloading the >> operator