CS250 Assignment 4: Pacific University Courses – Take Two

Date assigned: Monday, March 11, 2019
Date due: Wednesday, March 20, 2019
Points: 40

Goals for Assignment 4

1. Build an ADT.
2. Break up a program into well-defined functions. It is important that your functions be small and focused on a particular task.
3. Implement operator overloading functions in a class.
4. Begin using pointers.

This assignment takes the same input and output as assignment 1, but will now be implemented using a class with operator overloading, and using pointers.

You've been hired by the University to work on part of the course enrollment system. Your task will be to read course enrollment data from a file, print this information on request, and enroll students in courses.

You have an input file with the following information about the courses

- Prefix (only CS or MATH will appear in the file)
- Course Number
- Max Capacity
- Current enrollment

Now, you need to provide a menu of functions for users to interact with your software:

1. Print all courses
2. Print a single course
3. Add a student to a course
4. Quit

The program should continue to allow the user to enter menu choices and receive information until the user chooses to quit.

Class Course: Create a class named *Course*, which represents the information for one course. You must provide the following functionality for the class *Course*:

- An appropriate constructor
- Determine if a course is full or if it has space for a student.
- Determine if a course matches a passed in prefix (CS) and course number (250).
- An overloaded ++ operator that will increase the enrollment of a course by 1. The prototype for the function is: `Course& operator++();` The ++ operator can only be used as a prefix operator. The function must return `*this`.

You must also provide two friend functions:

- `operator>>` will read in information for one course. This can come from standard input (cin) or a file.
- `operator<<` will print out the information for one course. The destination is either the standard output (cout) or a file.

Notes on class Course:

- None of the functions in the class Course should print to the screen, except for `operator<<`.
- `operator++()` should never cause a course to go over capacity, but it should also never print out an error to the screen. It should just fail silently.

Driver: In the driver you must do the following:

- Create an array of pointers to Course. The maximum size for the array is 10.
- Read the contents of the input file (see below) and store them in the array.
- Write the code necessary to provide the interaction and output on the following page.

Input ("courses.txt"): The file will contain lines as follows. The last two integers on a line are capacity and current enrollment, respectively. You must be able to read a maximum of 10 courses from the file. For example:

```
CS 250 48 36
CS 460 24 9
CS 494 24 9
MATH 121 25 23
MATH 240 30 20
```

Error Handling:

- If the user provides an invalid selection for the menu, redisplay the menu with no error message
- If the file contains a prefix other than MATH or CS, print an error message and terminate the program.
- If the file contains more than 10 courses, print an error message and terminate the program.
- If the enrollment is larger than the capacity, print an error message and terminate the program.
- If the class is full, no student can be added.
- For options 2 & 3, if the user inputs an invalid prefix, keep asking until they input a valid prefix.
- For options 2 & 3, if the course number does not exist, print a message and then show the main menu again.

UML:

Use UMLet to create the UML for the class Course. Turn in the UML diagram (hard copy) by the start of class on Wednesday, March 13.

Your **output** is to look exactly like the following:

```
*************************
PACIFIC UNIVERSITY COURSES
*************************

---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 1

CS 250 Cap: 48 Available: 12
CS 460 Cap: 24 Available: 15
CS 494 Cap: 24 Available: 15
MATH 121 Cap: 25 Available: 2
MATH 240 Cap: 30 Available: 10

---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 2

Prefix: MATH
Number: 121

MATH 121 Cap: 25 Available: 2

---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 3

Prefix: MATH
Number: 121

MATH 121 Cap: 25 Available: 2

Student was added to the course.
```

```
---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 3

Prefix: MATH
Number: 121

MATH 121 Cap: 25 Available: 1

Student was added to the course.

---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 3

Prefix: MATH
Number: 121

MATH 121 Cap: 25 Available: 0

The course is already full.

---------------------------
1. Print all courses
2. Print one course
3. Add a student to a course
4. Quit
---------------------------

Option: 4
```

Notes:

1. Your main function is to be mostly variable declarations and function calls.
2. Test your program one function at a time.
3. Use object-oriented design.

To complete this assignment you must submit the following:

1. An electronic copy of your program on Grace

a) Add a project named 04_PUCourseRevisited to your assignment solutions folder. It is vital that you name your solution and your project correctly!
b) Type your program (fully documented/commented) into the project. You need to follow the coding standards from the CS250 class site. Please be sure to read the coding standards.
c) Pay attention to the example output. Your program's output must look **exactly** like the sample output. The spacing and newlines in your output must match exactly.
d) Make sure that your program builds without errors & warnings and runs correctly. If you get any errors or warnings, double check that you typed everything correctly. Be aware that C++ is case-sensitive. You will lose 10% if there are any warnings and 40% if your program does not build successfully.
e) Once you are sure that the program works, it is time to submit your program. You do this by logging on to Grace and placing your complete solution folder in the **CS250-0X Drop** folder.
f) The solution must be in the drop folder by the time class starts on the day the assignment is due. Anything submitted after that will be considered late.

2. A hard copy of your program

a) The hard copy must be placed on the instructor's desk by the time class starts on the day that it is due.
b) The hard copy must be printed in color, double-sided, and stapled in the upper left corner if your solution contains multiple pages.
c) Your tab size must be set to 2 and you must not go past column 80 in your output.

Remember, if you have any problems, come to me straight away with your project on Grace. Good Luck!!!! ☺