UMKC School of Computing and Engineering

Assignment 4

Due: Sunday March 19

For this program, you'll write a class that uses dynamic memory, and also uses a dynamic array of them within the program.

AwsomeCS U. records its grades using the standard 4.0 scale, with plus / minus grades:

A: 4.0	B-: 2.7	D+: 1.3
A-: 3.7	C+: 2.3	D: 1.0
B+: 3.3	C: 2.0	D-: 0.7
B: 3.0	C-: 1.7	F: 0

The grade points (GP) earned for each class are found by multiplying the grade by the number of credits in the course. Earning a B+ in a 4-hour course would earn 3.3 * 4 = 13.2 Grade Points. The Grade Point Average is found by dividing the student's total Grade Points by the total number of credit hours completed.

If a student is enrolled in a course but has not completed it, a dummy flag value of -1 is recorded for the grade. Courses not yet completed are excluded from GPA calculations. If a student has not yet completed any course (e.g. first-semester freshmen), the GPA is -1.0 by definition.

For this program, you will write a Student class. A Student has an ID number, first and last name, and (optionally) a dynamic array containing 1 or more courses the student is enrolled in or has taken. (See below for the definition of the course struct.) A Student has the following public interface:

- Because it uses dynamic memory, it will need a copy constructor, destructor, and overloaded assignment operator.
- Define constructor(s) to take: an ID number and first and last names; a constructor that sets the ID to 0 and the names to empty strings. You can do this by writing two constructors, or writing one and giving default values to all parameters. Both constructors should have an empty course list.
- GetID() and GetName(), both const methods. GetID() returns the student's ID number. GetName() returns a string consisting of the student's last name, followed by a comma and a single space, then the first name.
- bool isEnrolled(const string& CourseName) const: Returns true if the course of the specified name is on the student's list of courses AND does not have a grade recorded for it, otherwise returns false.
- bool enroll(const Course& C): Adds course to student's list of courses, indicating that the grade has not been recorded yet.
- bool recordGrade(const Course& C): If student is enrolled in the specified course, AND the course passed in has a valid grade (range from 0.0 to 4.0), record the grade and return true; otherwise return false without making any changes.
- int hoursCompleted() const: Returns the number of credit hours for courses that have grades recorded.
- float GPA() const: Returns the GPA based on completed courses. Courses where the student is enrolled but has not yet received a grade are in progress and are NOT counted as part of the GPA calculation.
- Overload input and output operators for the Student. The output operator should write the ID number, first name, and last name, separated by whitespace; on the next line, the number of courses taken; followed by a list of the courses, using the output operator for courses. The list of courses taken should be indented by a tab:

```
21045  Calvin Hobbes
3
	CS101  3	3.3
	ENG100 3	3.0
	ARTHIST225	1	4.0
```

For the Course, use a struct, containing the course name (string), credit hours (integer), and grade points (float or double). AwsomeCS U. uses the convention that a Course has a grade of -1.0 if the student is currently taking the course but no grade has yet been recorded. A grade in the range from 0 – 4.0 indicates the student has completed the course. (For this program, we will ignore issues regarding incompletes, withdrawals, etc.) Overload the stream insertion and extraction (output and input) operators for this struct. You don't need a separate set of header/implementation files for this; put it with the code for Student.

The main program has 2 input files: The first, students.txt, contains one or more students. Your program should use a dynamic array to store them. The second input file, courses.txt contains a series of input data. Each consists of a student ID number, followed by a course with a grade recorded. Your program should take the following action:
No student with this ID number: Print a brief error message, go to next input.
Student exists but is not enrolled in this course: Print a brief error message, go to next input.
Student exists and is enrolled in the course: Update grade.

After all input has been processed, produce 2 output files: The first is a listing of the students in the same format as students.txt. The second is a listing of the students, with student name, and GPA, each on a separate line.

Below is a sample piece of the two output files:

1- **New students File:**

```
794024  Virginia    Newman
9
ART426  3   3.7
CE132   3   3.3
CE275   3   3.3
CS165   4   -1
EE201   3   2.7
ENGL121 3   0.7
ENGL175 3   -1
MATH328 3   -1
MUS239  3   -1
```

2- GPA File:

```
Newman, Virginia    2.74
Gross, Kyle 2.88077
Tran, Moses 3
May, Brooklyn   3.33333
Hale, Michael   1.83333
Nichols, Natalie    2.34789
Williams, Katherine 2.95
Cunningham, Casey   2.76552
Chan, Myles 2.85
Jenkins, Tatum  2.59318
Burke, Elise    2.325
Manning, Bryan  3.03333
Moreno, Skylar  1.9375
Payne, Chandler 2.39756
Mccormick, Logan    2.46667
```