Rochester Institute of Technology
National Technical Institute for the Deaf
Information and Computing Studies Department

Name: Edward Riley

NACA.161 Programming Fundamentals II
In-class Exercise Day 8 – Loops

Overview

This lab involves the development of a program to compute the Grade Point Average (GPA) of a student based on the grade and # of credits for each of the student's courses. For each course, the needed information is the final letter grade for the course and the # of credits for the course. A letter grade can be converted to "points" for the GPA calculation using this table:

Grade	Points
A or a 4	4
B or b 3	3
C or c 2	2
D or d 1	1
F or f 0	0

Multiplying the letter grade points by the # of credits gives the "weighted points." For example, a grade of B in a 4-credit course yields 12 weighted points. The formula to compute the GPA for all courses is: as follows:

(sum of weighted points) ÷ (sum of credits)

Develop the class **Gpa** that will compute a student's GPA and a test class **ComputeGpa** to prompt for the grade and # of credits for each course, and print the GPA after all course data have been entered.

Based on the description above, what are the attributes of the class **GPA**?

Grades to Credits

Based on the description above, what are the methods of the class **GPA**?

Exercise #1 – Testing Conversion from Letter Grade to Points

Write a class Gpa that has the no attributes and the following method.

Method	Parameters/Return	Purpose
getPointsForGrade	Letter grade, type char as parameter. Returns the points for the grade	Converts the grade to its equivalent point value.

A switch statement **must** be used in getPointsForGrade to convert the grade to its equivalent number of points. Return a -1 if the letter grade is not valid.

Write a class named ComputeGpa with a main method. Prompt the user to enter the letter grade. Use the method getPointsForGrade to compute the number of points. An infinite loop **must** be used so that the user can enter all of the valid grades, both upper and lower case, and a selection of invalid grades to thoroughly test the method getPointsForGrade.

Notes:

- The return value of -1 will be used in other exercises to validate the letter grade entered by the user; at this point, simply print the -1.
- There is no method in the Scanner class to read a character. Read a string and then select the first character as the grade using the charAt method.
- Assume that the user will enter at least one character for each letter grade.
- The infinite loop is easy to write using a while loop; i.e. use while(true). This allows for testing a large number of inputs without having to re-execute the program. jGRASP provides a button to halt the execution.
- Note: In order to thoroughly test the method getPointsForGrade, the testing must include entry of all valid upper case and lower case letters and some invalid letters.

Sample Output

```
Enter grade (one character): a
Points: 4
Enter grade (one character): B
Points: 3
Enter grade (one character): c
Points: 2
Enter grade (one character): D
Points: 1
Enter grade (one character): e
Points: -1
Enter grade (one character): F
Points: 0
Enter grade (one character): x
Points: -1
Enter grade (one character): ^C
```

Exercise 2 – Computing GPA

Based on the formula above, the class Gpa will need to keep track of the sum of weighted points and sum of the credits. Add the following two attributes to the class Gpa:

Attribute	Meaning
sumCredits	Sum of credits for all courses entered by the user; of type int.
sumPoints	Sum of the weighted points for all courses entered by the user; of type int.

Add the following methods to the class Gpa:

Method	Parameters/Return	Purpose
Constructor	No parameters. No return type.	Set both sums to 0.
addGrade	2 parameters. Param1 is the letter grade as type char. Param2 is the number of credits as type int. No return value.	Add the credits to the sum of credits. Convert the letter grade to points and add the product of the points and # of credits to the sum of weighted points.
calcGpa	No parameters. Returns the GPA as type double.	Compute the GPA by dividing the sum of weighted points by the sum of credits.

To test the class Gpa, add more code to the class ComputeGpa. After prompting for the letter grade, prompt the user for the number of credits.

Add a loop to allow the user to enter the letter grade and # of credits for multiple courses. There are many loops that could be used. For this exercise, assume that each student takes exactly 3 courses.

After prompting for the information for the three courses, print the GPA. Format the output so that only two fractional digits are printed.

The limited amount of output makes it difficult to determine if the sum of points and sum of the weighted points have been calculated correctly. Modify the two classes so that the sum of credits and sum of weighted points are printed after each course is entered. See the two outputs below. With the extra information, it is then possible to determine if both sums are computed correctly.

Notes:

- Assume the user enters a valid letter grade.
- Assume the user enters a non-negative number of credits, that is, 0 or greater.
- Provide access to the two sums by two accessor methods.
- The main method no longer directly calls the method getPointsForGrade; instead, it should now be called by the method addGrade.

Sample Output

```
Enter grade (one character): a
Enter credits: 4
Sum Points: 16  Sum Credits: 4
Enter grade (one character): b
Enter credits: 4
Sum Points: 28  Sum Credits: 8
Enter grade (one character): C
Enter credits: 2
Sum Points: 32  Sum Credits: 10
GPA: 3.20
```

Exercise 3 – Data Validation and User-entered Number of Courses

Now that the GPA is being calculated correctly, validate the user input, and allow the user to enter the number of courses to be averaged.

A programmer needs to assume that any user input can be entered incorrectly. In the previous exercises, the user knew how to enter valid input. Now each input must be validated. For the number of courses, the user must enter a number greater than 0. If the number of courses is invalid, print an appropriate error message and prompt the user to re-enter the number of credits.

For the letter grade, the user must enter exactly one letter and the grade must be one of the valid letter grades (A, B, C, D, F, upper or lower case). Part of the validation is already implemented, since the method getPointsForGrade returns a value of -1 if the letter grade is invalid. If the input is invalid, print an appropriate error message and prompt the user to re-enter the letter grade.

For the number of credits, the user must enter a number for the credits between 0 and 9, inclusive. If the number of credits is invalid, print an appropriate error message and prompt the user to re-enter the number of credits.

Sample Output #1

```
Enter number of courses: 0
Invalid number of courses - must be greater than 0
Enter number of courses: -2
Invalid number of courses - must be greater than 0
```

```
Enter number of courses: 2
Enter grade (one character): a
Enter credits: 4
Enter grade (one character): B
Enter credits: 4

GPA 3.50
```

Sample Output #2

```
Enter number of courses: 2
Enter grade (one character): x
Invalid grade - must enter A,B,C,D,F (upper or lower case)
Enter credits: xx
Invalid grade - must enter exactly one letter
Enter grade (one character): a
Enter credits: 4
Enter grade (one character): b
Enter credits: 99
Invalid credits - must be between 0 and 9, inclusively
Enter grade (one character): b
Enter credits: 4

GPA 3.50
```

When you complete all of the steps successfully and answer all of the questions, contact your instructor to check if your program(s) executes correctly and to review your code. We will initial the line below.

_____Successful execution of code

If you do not finish the program during the class period, contact your instructor to initial below so that you can complete it before the next class period.

_____Code not completed during lab time

You may then have your instructor verify your work at the start of work period in the next class. If you do not have a signature, then you can not receive any points for this assignment.