CS160 Computer Science I
Project 3

Objective:
Work with dictionaries
Work with files
Work with functions

Assignment:

Part 1
Create a data file. You can do this using any text editor. You DO NOT need to write a program to do this part. The layout of the text file will have one line per class, with the class name, letter grade, and number of credits (an integer) separated by a colon (":"), so be sure not to include a colon in the class name.

Part 2
Requirements:
The only possible grades received are A,B,C,D, or F
The letter grades can be in upper or lower case
You can set up a separate initialized dictionary for the grades, with the grade as the key and the associated number of honor points as the value – this will be of help when determining the GPA. An A is worth 4 honor points, a B is 3 honor points, a C is 2 honor points, a D is 1 honor point, and an F receives no honor points.

Write each of the following functions. The function headers **MUST** be written as specified and each must have a comment block.

Required functions:

`readClassInfo(fileName)`
This function creates a dictionary, and fills it by reading in the information stored in the `filename` variable. Each line in the file will contain a class name, grade for the class, and the number credits for the class. Each piece of information will be separated by a '\t' (tab). This must call the function `addClass` to add the information to the dictionary.

`writeClassInfo(filename, classes)`
This function saves the dictionary `classes` to the file name stored in the `filename` variable. Each line in the file will contain a class name, grade for the class, and the number credits for the class. Each piece of information will be separated by a '\t' (tab).

`addClass(classes, className, grade, credits)`
This function will add a class with `className` to the `classes` dictionary and return `True` if it was added or `False` if it is already in the dictionary (meaning the information for the class was updated). Note that the `classes` dictionary will have `className` for key. You do not have to worry about invalid letter grades, if will always be A,B,C, D, or F, but there is no guarantee about the case.

`attemptedCredits (classes)`
This function will return the number of attempted credits (NOT classes) – the total of all classes taken.

`passedCredits (classes)`
This function will return the number of credits where the students received a passing grade (A, B, C, or D).

`printClasses (classes)`
This function will print out all the classes in a neat table, with headings, followed by the GPA, with 3 places after the decimal point. Display the letter grade as uppercase. An example, with completely made up data, might be:

```
   Class     Grade   Credits
1. CSci 160:   A           4
2. CSci 161:   B           4
3. CSci 289    A           3
4. …

Overall GPA: 3.456
```

`getGPA (classes)`
This function will return the student's GPA. Remember, GPA is calculated by determining the total honor points divided by the total number of credits.
The honor points for a class are the number of credits multiplied by 4 for an A, 3 for a B, 2 for a C, 1 for a D and 0 for an F.
To find the GPA

- Sum the honor points for each class.
- Divide the total number of honor points by the total number of credits.

For example, if a student has a 4 credit A and a 3 credit C, the total number of honor points would be 4 * 4 (for the A) + 3 * 2 (for the C), or 16 + 6, or 22.
The total number of credits would be 4 + 2, or 6.
The GPA would be 22 / 6, or 3.667 (rounded to three places).

`updateGrade (classes, className, newGrade)`
If the class exists in the dictionary this function will update the grade of the specified class to `newGrade` and return `True`. If the class is not in the dictionary return `False`.

`classStatus (classes)`
Returns the class status of the student, “Freshman”, “Sophomore”, “Junior”, or “Senior”. The class status is determined by using the number of passed credits. Use the standard criteria of 24 (sophomore), 60 (junior), and 90 (senior) credits.

`main()`
This function will test all the other functions. THERE IS NO NEED TO USE INPUT STATEMENTS OR A MENU. You can hardcode all the values when testing. You can ask questions in your main program, but it is not a requirement.

REMEMBER TO COMMENT YOUR CODE!