CSCI 2010 Assignment 7

OBJECTIVES
This assignment has you work with operator overloading, static member and aggregation.

Q1: Overloading [40 points]
Modify the `FeetInches` class discussed in textbook Chapter 14, so it overloads the following operators:

<=
>=
!=

- For this question, name your class `<my name>FeetInches`. For example, my class would be named: `LeongLeeFeetInches`.
- Do NOT use inline member functions for the three overloaded operator functions mentioned.
- You need to submit the following files:
    1. `<my name>FeetInches.h`: Specification for class should be in this **.h** file
    2. `<my name>FeetInches.cpp`: Member function implementations of the three overloaded operator functions should be in this **.cpp** file
    3. `<my name>FeetInches_Demo.cpp`: You also need to write a separate program to test the class. In other words, main() is here.

Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 point for the question.**

Estimated time: 2 hours

Q2: Aggregation and static member [60 points]
A trading company has three divisions, each responsible for sales to different locations. Design a `DivSales` class that keeps the sales data for a division, with the following members:

- An **array** with four elements for holding four quarters of sales figures for the division.
- A **private static variable** for holding the total corporate sales for all divisions for the entire year
- A **member function** that takes four arguments, each assumed to be the sales for a quarter. The value of the arguments should be copied into the array that holds the sales data. The total of the four arguments should be added to the static variable that holds the total yearly corporate sales.
- A **function** that takes an integer argument within the range of 0-3. The argument is to be used as a subscript into the division quarterly sales array. The function should return the value of the array element with that subscript.

Write a program that creates an array of three `DivSales` **objects**. The program should ask the user to enter the sales for four quarters for each division. After the data are entered, the program should display a table showing the division sales for each quarter. The program should then display the total corporate sales for the year.

- For this question, name your class `<my name>DivSales`. For example, my class would be named: `LeongLeeDivSales`.
- Use inline member functions for all the method functions mentioned.
- You need to submit the following files:
    1. `<my name>DivSales.h`: Specification (and inline member functions) for class and other functions should be in this **.h** file
    2. `<my name>DivSales_Demo.cpp` : You need to write a separate program to test the class. In other words, main() is here.

Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 point for the question.**

Estimated time: 4 hours

SAMPLE RUN (Q2 only)
Your output should look similar to the following. Things in bold are typed by the user.

```
Enter sales data for Division 1
        Quarter 1: 1000
        Quarter 2: 2000
        Quarter 3: 3000
        Quarter 4: 4000
Enter sales data for Division 2
        Quarter 1: 2100
        Quarter 2: 3200
        Quarter 3: 4200
        Quarter 4: 5200
Enter sales data for Division 3
        Quarter 1: 6000
        Quarter 2: 7000
        Quarter 3: 8000
        Quarter 4: 9000
================================
--- Report ---
Sales for Division 1
        Quarter 1: $1000.00
        Quarter 2: $2000.00
        Quarter 3: $3000.00
        Quarter 4: $4000.00
Sales for Division 2
        Quarter 1: $2100.00
        Quarter 2: $3200.00
        Quarter 3: $4200.00
        Quarter 4: $5200.00
Sales for Division 3
        Quarter 1: $6000.00
        Quarter 2: $7000.00
        Quarter 3: $8000.00
        Quarter 4: $9000.00
Press Enter to Continue.
Total Corporate Sales: $54700.00
```

Submission instructions:
You need to compile the above programs (questions) separately, and provide **two test cases** (if applicable) for each program (question). Do a screen capture of the input and related output for each test case. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks) to cut out the program input and output (from the screen capture), paste them into a word document under a related question number, save the document as a pdf file. A sample input/output (screen capture) can be found at the end of this document.

You need to submit the following:
1. A pdf file containing the screen captures of program input and output of all test cases, name the file **lastname_firstname_assignment07.pdf**.
2. All .h and .cpp files (5 files in total for this assignment). Zip your .cpp files into a single zip file (or rar file) **lastname_firstname_assignment07.zip**.

Please submit electronic copy (the above mentioned **two files**) to D2L digital dropbox.
If you cannot follow the above instructions, points would be deducted.

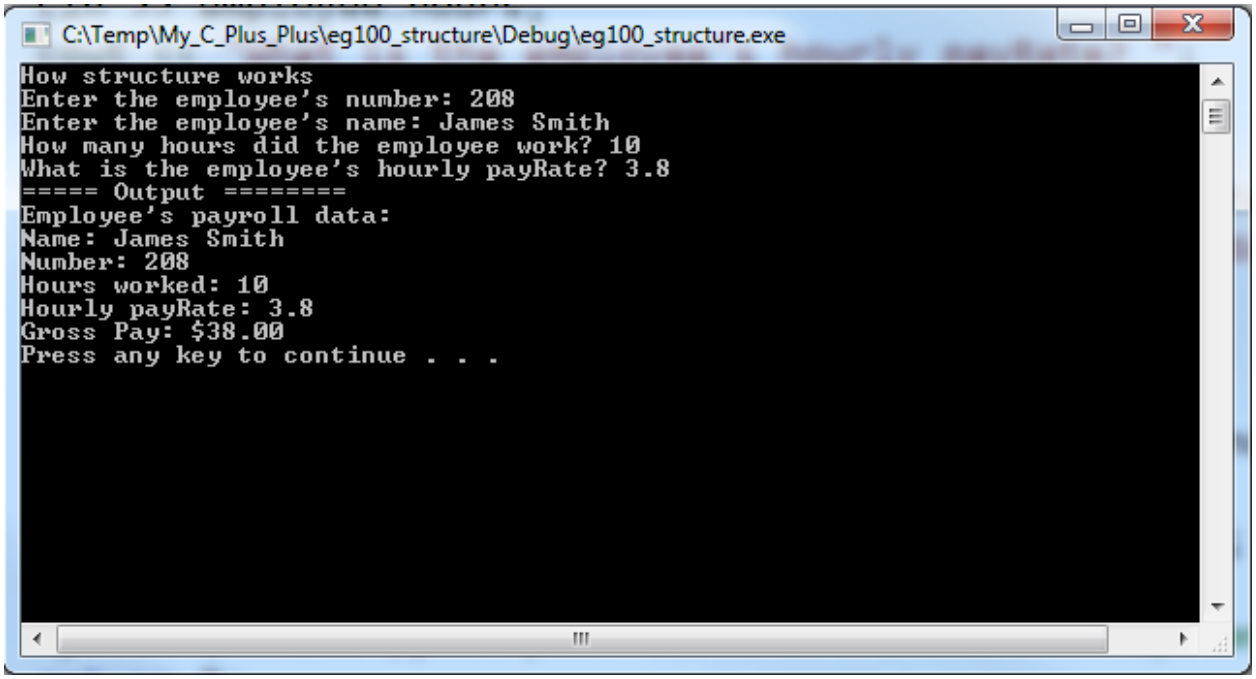
Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- Correctness (50%): Does the program compile correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down in a reasonable way (e.g. classes and methods)?
- Style (10%): Is the program **indented** properly? Do variables have **meaningful names**?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully?
- Documentation (10%): Do all program files begin with a **comment** that identifies the author, the course code, and the program date? Are all the classes, methods and data fields clearly **documented (comments)?** Are unclear parts of code **documented (comments)?** (Some items mentioned may not apply to some languages)

A program that does not compile will get at most **50% of the possible points**.

Sample input/output (screen captures)

Assignment X, test case 1, input/output:



Screen capture must be readable by the instructor, or 0 point will be given.
Please note that you can use more than one screen captures for each test case.