CSCI 2010 Assignment 2

OBJECTIVES
This assignment has you work with structs and arrays of structs.
It is also a review of CSCI-1010 topics such as arrays, iomanip functions, and searching arrays with linear search.

INSTRUCTIONS
Write a program that stores the data about the profit made of a team of bank traders.

Create a project in Visual C++ Express named **<my name>Assignment2** and add a .cpp file named **<my name>pass2.cpp**.

The code for <my name>pass2.cpp should read the following information from the keyboard:

- Trader number → integer
- Trader name → string
- Trader unit → string
- Trader Profit → double

The information for each trader should be stored in a record defined by a `struct`. All the traders' records should be stored in an array of `struct`s. Assume that **up to 12** traders could be entered.

Trader names can have 1 or more spaces so be sure to use the `getline()` function to read in a name.

Input should stop when the sentinel value `-999` is typed in the trader number or the array is full (**12 traders**).

After the sentinel is typed, display to the monitor in columns with column headings, trader number, trader name, and trader profit. List the traders in **reverse order** (first one listed is the last element in the array) of the order they were input.

Your columns should line up nicely. In order to get the name column to line up properly, you will need to find the name with the longest length and use that as an argument to `setw()`. Use string's `length()` method to check the length of names.

After the data is printed, output the total number of traders, the total profit, the average profit, the lowest profit and the trader with the lowest profit (he/she would be issued a warning letter for poor performance).
Write **functions (or methods)** to:

- Read the list of traders
- Print the table of traders
- Compute the total profit, the average profit, the lowest profit and the trader with the lowest profit
- Display the total profit, the average profit, the lowest profit and the trader with the lowest profit

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: 5 hours

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

```
Welcome to the trader account manager!
Enter trader 1 number (or -999): 123
Enter trader 1 name: Wang Lim
Enter trader 1 unit: Asia
Enter trader 1 profit: 100000.00

Enter trader 2 number (or -999): 5555
Enter trader 2 name: Luna Lovegood
Enter trader 2 unit: Europe
Enter trader 2 profit: 31000.00

Enter trader 3 number (or -999): 987
Enter trader 3 name: Justin Fletchley
Enter trader 3 unit: USA
Enter trader 3 profit: 58000.00

Enter trader 4 number (or -999): -999

TRADER REPORT
------
NUMBER    NAME                  UNIT            PROFIT
00000987  Justin Fletchley      USA          $58000.00
00005555  Luna Lovegood         Europe       $31000.00
00000123  Wang Lim              Asia        $100000.00

Number of traders: 3
Total profit: $189000.00
Average profit: $63000.00
Lowest profit: Luna Lovegood, $31000.00
```

Submission instructions:
You need to compile the above program, and provide **two test cases** (if applicable). 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_assignment02.pdf**.
2. A .cpp file **<my name>pass2.cpp** (e.g. lastname_firstname_pass2.cpp).

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

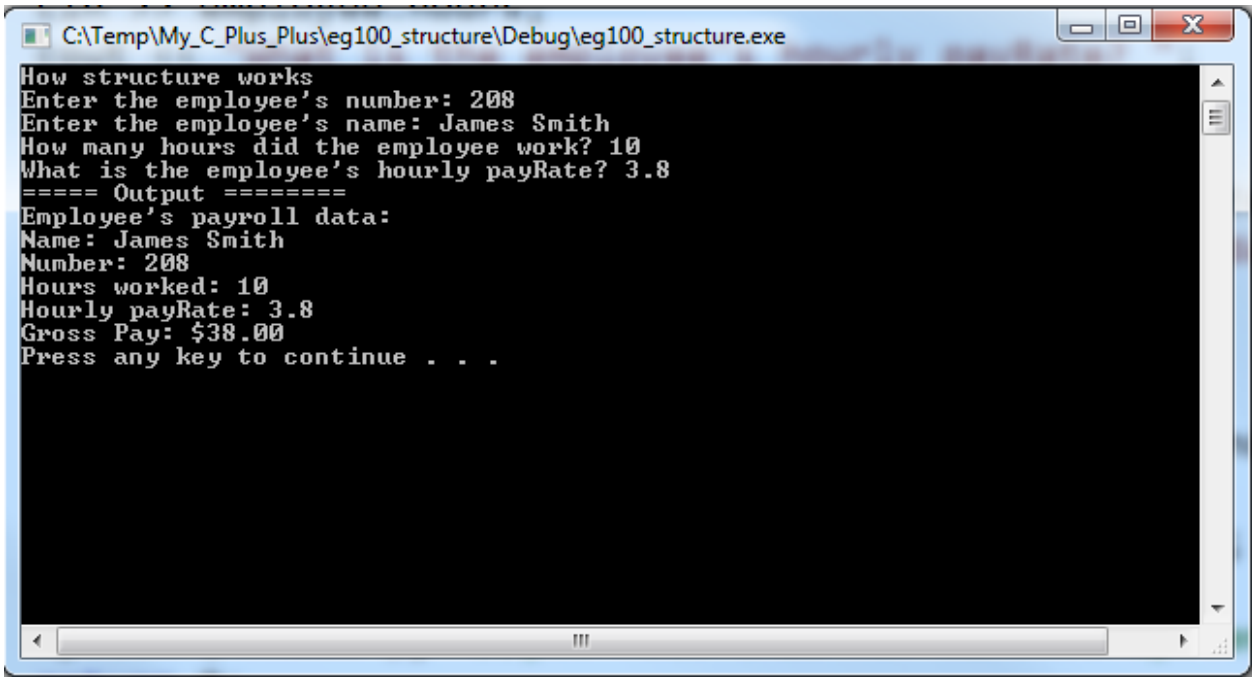
Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- 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.