CSCI 2010 Assignment 9

OBJECTIVES
This assignment covers Chapter 17 Linked Lists.

INSTRUCTIONS
Create a project in Visual C++ Express named `<my name>Pass9`.
Add `eg141_NmberList.h` and `eg141_NumberList.cpp` files to your project. These are included in the zip file with this document. The files implement a linked list of numbers.
Create and add `<my name>pass9_demo.cpp` to your project. This will have `main()` and use the `eg141_NumberList` class.

The purpose of this assignment is to add data members and methods to the `eg141_NumberList` class in order to give it more functionality.

Add the following to the `eg141_NumberList` class:

1. Add a method **<my name>empty()**
    a. It should take no parameter.
    b. It should return true if the list has no node, and false if there are 1 or more nodes.
2. Modify the **<my name>displayList()** method
    a. If the list is empty, print "The list is empty".
3. Add a method **<my name>find()**
    a. It should take one parameter, a value.
    b. It should go through the linked list. If the value is found, it should return the position of the value in the list.
        i. If the value is in the first node, return 0. If the value is in the second node return 1, and so on.
        ii. If the value is not found in the list return -1.
4. Add a method **<my name>sum()**
    a. Return the sum of all the values in the list.
    b. If the list is empty, return 0
5. Add a data member **<my name>size**.
    a. The size represents the number of nodes in the linked list. Think of it as the length.
    b. It should be an integer.
    c. Add a method **<my name>getSize()** that returns the current value of size.
    d. Whenever a new node is added to the linked list, the size should increase by one.
    e. Whenever a node is deleted from the linked list, the size should decrease by one.
6. All methods should have prototypes in the `eg141_NmberList.h` file and implemented in the `eg141_NmberList.cpp` file.

The `main()` should do the following.

- Create a `eg141_NumberList` class to be used in the rest of the `main()`.
- Print the following menu

```
1.    Append a value
2.    Insert a value
```

```
3.    Delete a value
4.    Print the list
5.    Print the size
6.    Print the list sum
7.    Find a Number
8.    Check if the list is empty
9.    Exit
```

- Read the user's choice and do the following. Use the appropriate methods in the `eg141_NumberList` class.
    - Option 4. Print all the existing nodes in the list.
    - Option 1. Ask the user for a value, and append the value to the end of the list.
    - Option 2. Ask the user for a value, and insert the value in the list.
    - Option 3. Ask the user for a value, and delete it from the list.
    - Option 4. Print all the nodes in the list.
    - Option 5. Print the size of the list.
    - Option 6. Print the sum of all the values in the list.
    - Option 7. Prompt the user to enter a number, return the position if it is found.
    - Option 8. Check if the list is empty.
    - Option 9. Exit the program. The program should not quit until this option is selected. Use a loop that keeps printing the menu and asks for user menu selection.

You need to submit the following:

1. You need to **zip the entire project folder** which should contains the project files and the following source files:
    1. `eg141_NmberList.h`
    2. `eg141_NmberList.cpp`
    3. `<my name> pass9_demo.cpp`
2. Name the ZIP file `<my name>Pass9.zip`. Upload the ZIP file into the DropBox in D2L.

Important: If you do not put `<my name>` to the above mentioned fields, **you will get 0 point.**

Estimated time: **6 hours**

Submission instructions:
You need to compile the above program, and provide **one test case**. Do a screen capture of the input and related output for the 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_assignment09.pdf**.
2. lastname_firstname_Pass9.zip (<my name>Pass9.zip).

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

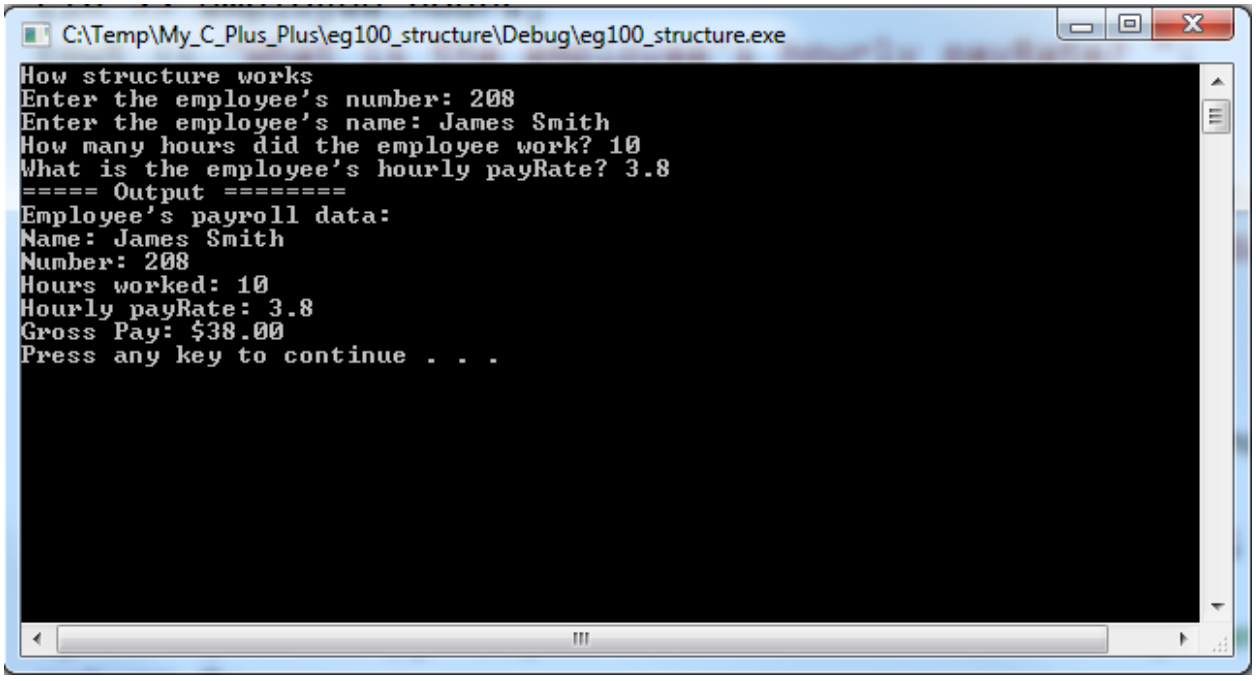
Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- 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.