# CS221: Data Structures — Programming Assignment #3

Due: see canvas for due dates — (30 points)

## Problem Statement

1. You are to refine your Player List from Program 2 to be implemented with a doubly-linked list instead of single linked list in the class to store them. The input and output file formatting requirements are unchanged from Program 2.
2. Also, you must maintain the list in sorted order (alphabetical by last name, then first name). This means that every time a player is added to the list, you must locate, and place that item in the correct spot. This will require you to be able to compare two players by name. For example, these names are in sort order:

   *Joe Buck*
   *Thomas Jones*
   *Bob Miller*
   *Mark Miller*

   *You may assume that case sensitivity does not matter for this problem.*

3. You must provide the user with the ability to delete a player from the list by providing their firstname and lastname. Only delete the player node with an exact match. In order to test this feature, you will need to add a loop to the main program that prompts the user for the name to remove and then prompt them to see if there are more to remove.
4. Write the list of players contents to the report file both before and after testing the remove operation. Also, at the end of the report, print the player list in reverse order to ensure your back links are implemented properly.

**Other Requirements**

- Make sure your name and the compiler version are noted in the comments at the top of your main program.
- You may not use any built-in or C++ library sort functions.
- IMPORTANT: I will no longer accept programs that have the following non-standard C++ feature in them:
  - You may not initialize members in the class variable declaration section. They must be initialized in the constructor. These will NOT compile in many versions of C++
  - Ex:

```
class Player {
     double average = 0.0;    ← this assignment should not be here
     . . .
};
```

**Helpful Information on how to compare two strings in C++**

http://www.cplusplus.com/reference/string/string/compare/

**TURN IN:**

For this assignment, submit the electronic version of your code to canvas.

Grading Notes:

- Your program must be well-commented. Comment all variables, functions and remember to have a section of comments at the top of your program that includes your name, date, course section and a description of what your program does. See my posted examples.
- Use good variable names.
- Use good and consistent naming conventions for class members.
- YOU MUST Separate the class implementations into their proper .cpp, .h file sets.
- Use proper code indentation to make sure your program is easy to read and understand.
- You will receive no more than 50% credit if your program does not compile.
- If your program compiles but does not execute correctly, you will receive no more than 70% credit.

**Sample Execution [ corrected a typo on screen sample ]**

```
Welcome to the player statistics calculator test program.  I am going to
read players from an input data file.  You will tell me the names of
your input and output files.  I will store all of the players in a list,
compute each player's stats and then write the resulting report to
your output file.


Enter the name of your input file:  playerinput.txt
Enter the name of your output file: report.txt

Reading Players from: playerinput.txt
The sorted list has been written to your output file: report.txt

Would you like to remove any players from your list?  Y
    Please enter the First and Last Name of the Player: Chipper Jones
    Chipper Jones Removed Successfully.

Would you like to remove any players from your list?  Y
    Please enter the First and Last Name of the Player: John Smith
    John Smith was not found in your team.

Would you like to remove any players from your list?  N

Testing Complete. The new version of your list has been added to
the report file.

End of Program 3
Press any key to continue . . .
```

***OUTPUT SHOULD Be the EXACT same format as in program 2. Please note the extra reporting information above (Print the sorted team before and after testing the deletions. Additionally print the team in reverse order.)***

**TIPS:**

**I only tested this with one test case, removing a player from somewhere in the middle of the list. You should test other scenarios such as removing players from the ends of the list.**

Note that the points for this assignment are higher than the first two programs. It usually takes considerably more time to complete the assignment.