**CSCE 2014 - Homework 5   
Due Date - 04/30/2015 at 11:59 PM**

**Problem Statement**

The goal of this assignment is to give students experience with inserting data into binary search trees, and using these BSTs to print a subset of data in order. In particular, we will be reading a quarterback file QB.txt and storing selected information about quarterbacks in our BST. Then we will be searching this BST to print out a subset of the quarterback objects based on a "range query". For example, to print all objects with (80.0 < Rating < 90.0). The detailed requirements are listed below.

* The quarterback file contains 17 fields of information, which is too much for this application. Your first task is to write a function to open and read this file, and select out four information fields (**Player, Att, Yds, and Rating**). The other fields can be ignored. To start, you can simply print these four fields. Later, you will store this quarterback info in your binary search tree.
* Create a C++ class to store the four quarterback information fields above, and implement methods to "set" and "get" these fields, and a method to "print" the data. Your most complicated task is to implement some method to compare two quarterback objects to each other based on ONE of **Att, Yds, or Rating** data fields in the object. Your comparison methods will be called by the BST search, insert and delete to correctly "walk the BST".
  + One option would be to create three methods "less\_than", "greater\_than" and "equal" that return true/false based on the field you choose to compare.
  + Another option would be to create a method called "compare" that returns -1 if (qb1.field < qb2.field), 0 if (qb1.field == qb2.field), and 1 if (qb1.field > qb2.field), where "field" one of the 4 information fields above.
* Modify the binary search tree code tree.h and tree.cpp so the tree can store quarterback objects instead of just integer values. This will require significant changes to the parameters for each method, and also to the data comparisons in these methods. The actual logic of how insert, search, delete, and print work should remain essentially unchanged.
* Create a new method for the BST class that allows you to print out a subset of the quarterback objects based on a "range query". For example, if you inserted QB objects based on their Ratings, you can use this information to traverse the tree to print out just the objects that fall within a range [Low\_Rating .. High\_Rating]. As a side benefit, these records will be printed in order.
* Create a main program that calls your function to read the QB.txt file, and insert this QB information into your binary search tree. Then prompt the user to enter a criteria of **Att, Yds, or Rating**, and then the low and high of the criteria. You program should print out only the objects that fall within the specified range [low .. high] of that criteria.

**Assignment Submission:**

You need to submit 1) C++ code, 2) design document, and 3) test results for this assignment. Put all of them in a zip file, and name it as <last name>\_assignment3.zip. You should use the Moodle system for this submission. When you have completed your C++ program, use the "Programming Report Template" on Moodle site to document your programming project. This report has separate sections to describe the problem statement, your design decisions, your implementation process, and your testing results. Your document should be less than a page long.