PROGRAM 10-1 BINARY SEARCH TREES

PROBLEM STATEMENT

You are to develop a program to use a Binary Search tree to store your sorted collection of baseball players. See the earlier assignments for the requirements for the Player object. That has not changed.

You must implement the following operations on your List along with any other utility functions you might need. You may also need to add operations to your Baseball Player class, as well.

Your Tree class must implement at least the following interfaces, and any supporting methods you need. *Reminder: Because many tree operations are recursive using a node pointer to a root or subroot as input, many operations have a private version that is called with the correct pointer and a public version presented to the user who has no view into the internal implementation of the tree.*

- default **constructor** and **destructor**
- **insert** or **add** - to add a new player to the tree
- **clear** - to clean up a tree and free all of the nodes in the tree, may be called by the destructor
- **getSize** - to return a count of all players in the tree
- **getTeamAverage** – to return the overall batting average for all of the players in the collection
- **print** - to output all of the players, in the appropriate traversal order.

Your program should:

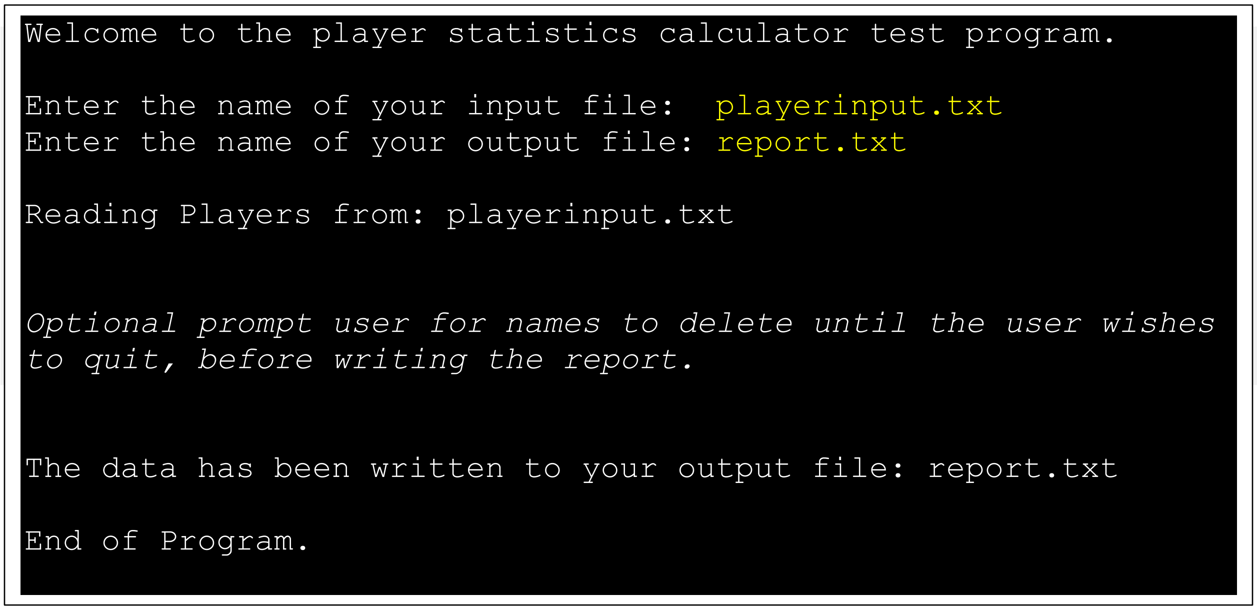
- Open the input and output data files specified by the user.
- Read in the players into the binary tree structure.
- Print the contents of the tree to the report file in sorted order (ascending). Use the same report format as our prior programs. Also,
- Print the contents of the tree again in reverse order (descending) to the output file.

CHALLENGE (INSTRUCTOR MAY ADD THIS AS AN ADDITIONAL REQUIREMENT)

- Implementing **remove(player)** to find and remove players using their name as the key. You should test it with a loop that prompts for player names until the user wishes to quit testing the remove feature. After removing some data, print the tree to the screen to ensure it is still organized correctly.

Other notes: It is very difficult to write an external iterator for a binary tree. You need a companion data structure called a Stack to manage it. Therefore, we will only use our ***traversal*** *method(s) to visit and print each of the nodes in the tree, rather than have a loop in the main program try to do it.*

SAMPLE EXECUTION



```
Welcome to the player statistics calculator test program.

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

Reading Players from: playerinput.txt


Optional prompt user for names to delete until the user wishes
to quit, before writing the report.


The data has been written to your output file: report.txt

End of Program.
```

SAMPLE INPUT FILE

```
Hank Aaron     13941 12364 2294 624  98 755 1402 32
Chipper Jones  10614  8984 1671 549  38 468 1512 18
Ty Cobb        13099 11434 3053 724 295 117 1249 94
Jonny Bench     8674  7658 1254 381  24 389  891 19
Tony Gwynn     10232  9288 2378 543  85 135  434 24
John Smoltz     1167   948  118  26   2   5   79  3
```

Note – there should be NO blank lines after the last line of data in an input file, or extra blanks at the end of lines.

SAMPLE OUTPUT TEXT FILE (NOTE SUMMARY AT TOP!)

```
BASEBALL TEAM REPORT --- 6 PLAYERS FOUND IN FILE
OVERALL BATTING AVERAGE is 0.290

    PLAYER NAME      :    AVERAGE    OPS
---------------------------------------------
         Aaron, Hank :     0.305    0.928
        Bench, Jonny :     0.267    0.817
            Cobb, Ty :     0.366    0.934
         Gwynn, Tony :     0.338    0.810
      Jones, Chipper :     0.303    0.930
        Smoltz, John :     0.159    0.406

For testing, list in reverse order is:
    PLAYER NAME      :    AVERAGE    OPS
---------------------------------------------
        Smoltz, John :     0.159    0.406
      Jones, Chipper :     0.303    0.930
         Gwynn, Tony :     0.338    0.810
            Cobb, Ty :     0.366    0.934
        Bench, Jonny :     0.267    0.817
         Aaron, Hank :     0.305    0.928
```

REFERENCES

I am using the online baseball reference (www.baseball-reference.com/players) to look up batting statistics. Please note that the on-base percentage I get is slightly off of the actual percentages due to not using quite ALL of the plate appearance data in my calculations.