Programming Assignment 2

Problem Statement:

- You are to develop a program to read Player objects from an input text file and store them in a list data structure implemented with an array. You may reuse your player object from program 1. See that assignment for the requirements for a single player.
- Each line of the data file will contain a single player's name and stats in the same order or form as we read them in program 1. You may assume there are no errors in the data. I will not leave data out or put non-numeric data in numeric fields. You must read until the end of the file is found.

New Requirements:

- You are to implement a **PlayerList** data structure/data type that stores players using an internal array in the class. Add players to the List as they are read in from the data file. You may assume that no more than **20** players are to be stored in a PlayerList. However, you must ensure that the list is not exceeded if there happen to be more than 20 in the input file.
- 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 player to encapsulate i/o operations to the class.

Operations for PlayerList:

- Default Constructor
- Add a player to the List - add items in the order they were found in the file. No sorting is necessary in this version of the program.
- Iterate through the List so that you can get each item out for printing
    - You will implement hasNext() and getNext() on your list for this to work.
- Clear out the list to make it empty
- Test if a list isFull
- Get the size of the list
- Any other useful methods you might require to solve the problem.

Summary of Operation:

- Prompt the user for the input and output file names. DO NOT hardcode file names into your program.
- Open input file
- Read each player and add them to your List.
- Keep track of the number of items in the list.
- Open an output file.
- Write a report summary line to the output file. The summary should include the number of players found in the file.
- Then, write each item from the list into the output file, along with any other output required by the assignment. SEE EXAMPLE FOR FORMAT OF REPORT.
- Do not continue adding players to the list, if it is full.

Other Requirements:

- Include in the comments at the top of the main program file the version of the compiler you are using. Don't forget your name/date/etc.
- Make sure when you begin, you create an empty project, with no pre-compiled header.
- Name your project as Firstname_LastName_Program2. For example if your name is Manaswini Maddu the name of the file should be as Manaswini_Maddu_Program2.
- You may never use built-in C++ template libraries in this course unless instructed to.
- Name your project's main file Program2_main.cpp and appropriate names for all the class files.

Turn In:

- Submit the electronic version of your project as a zip file to canvas for this assignment. Submit only the single zip file to canvas without including large database files.
- The zip file name format should be firstname_lastname_Program2. For example, as **Manaswini_Maddu_Program2.zip**

Grading Requirements:

- 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. (Internal documentation on programs in my courses counts for up to 20% of credit.)
- Always use good variable names. If I don't understand the program I cannot grade it.
- Use good and consistent naming conventions for class members.
- 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.
- Do not work with other students on the solution to this program. Copying code and unauthorized collaboration falls under plagiarism.

Sample Execution:

```
Welcome to the Cricket Player Stats Program.
This program reads the data from an input file, stores them in a list and then writes the result which stores the players data and stats in the output file.
The input file name and output file name taken from the user by prompting.

Enter input file name: playerinput.txt
Enter output file name: report.txt

Reading players data from: playerinput.txt
The result has been written to output file: report.txt

Program 2 is completed
```

Sample Input File and Corresponding Output File:

Sample input text file:

```
Russell Andre 56 1034 794 10
Maxwell Glenn 121 3390 2689 40
Kohli Virat 256 12471 13408 54
Buttler Jos 152 4150 3429 36
Warner David 141 6007 2048 28
```

Note – There should be NO blank lines after the last line of data in an input file.

Sample output text file:

```
Player Report - - - 5 Players found in the input file
Highest Batting_Average     -     Kohli, Virat     <---------- 10% Bonus for getting this line

PLAYER NAME                 :     Dismissals     Batting_Strike_Rate     Batting_Average
-------------------------------------------------------------------------------------------------------------------
        Andre, Russell      :          46              130.23                  22.48
        Glenn, Maxwell      :          81              126.07                  41.85
        Virat, Kohli        :         202               93.01                  61.74
        Jos, Buttler        :         116              121.03                  35.77
        David, Warner       :         113              293.31                  53.16
```