Programming Assignment 1

Problem Statement:
You are to develop a C++ program to define and test an object class whose instances will be used to store the data for a Cricket Player. In this first version of the program, a cricket player will have a first name, a last name (you may use string type where appropriate) and an array that stores 4 values for the passing stats. The data in the array contains: Number of Matches Played, Runs Scored, Total number of Balls Faced, Number of matches player remained not out **[MATCHES, RUNS, BALLS, NOT_OUTS]**

Your class needs to provide all the methods needed to:

- Provide default constructor for your **Player class**
    - default first and last name are "unknown"
    - default numeric values are 0
- Read or parse a Players data from a single line of text input
    - These will be used later when we start reading lines of data from an input file. The next iteration of this program will read a Player from a single line of input file, so we want to set this up to make that easier to reuse.
- **get** methods for any data the test driver might need to display on the screen later – your get operations should include gets to retrieve:
    - Player name
    - Dismissals
    - Batting Strike Rate
    - Batting Average
- **set** methods to copy values into the members of your Player object, where needed.
- And, any other methods you need to solve the problem.

TEST DRIVER:

- Once you have developed a good class definition, you need to write a "test" main program to test it out. Your test program should declare a variable of your new class type, and then prompt the user for the data to be set into the object. Then, display the characteristics of the Player for the user.
- Note that all interactive prompts are coming from the main program. The Player object itself is passive - it is not prompting for values. All values are placed into the object by using its methods.
- The pages at the end of this document have a sample test. Yours should behave and look the same way. (My sample input is highlighted in yellow).

NOTE:
Note that the value of this program is 20 points (out of a projected total of 100 programming points). The other programs in the course will be follow on programs to this one. As they

become more difficult/complex/time-consuming and follow on, please be careful for every programming assignment from the beginning.

Computing Statistics:
MATCHES - Number of Matches Played
RUNS - Runs Scored
BALLS - Total number of Balls Faced
NOT_OUTS - Number of matches player remained not out

- **Dismissals:**
  Dismissals = MATCHES - NOT_OUTS
- **Batting Average :**
  Batting Average = Runs/Dismissals
- **Batting Strike Rate:**
  Batting Strike Rate = (Runs/Balls)*100

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_Program1. For example if your name is Manaswini Maddu the name of the file should be as Manaswini_Maddu_Program1.
- You may never use built-in C++ template libraries in this course unless instructed to.

Turn In:

- Submit the electronic version of your project as a zip file to canvas for this assignment. DO NOT INCLUDE any large database file (extension .sdf) or the DEBUG folder in your zip file. The later versions of Visual create a hidden folder in your project directory named .vs, please do not allow that to be included in your zip. It is very large. Submit only the single zip file to canvas.
- The zip file name format should be firstname_lastname_Program1.
- For example, if your name is Manaswini Maddu your zip file should be named as **Manaswini_Maddu_Program1.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 of Code:

```
Welcome to the Cricket Player Stats Program.
When prompted, please enter the player's data in the form of
        firstname  lastname matches runs balls not_outs

Enter Player Data: Russell Andre 56 1034 794 10
        Russell Andre - Dismissals: 46 Batting_Strike_Rate: 130.23 Batting_Average: 22.47
Do you wish to test another [yes/no]? yes
Enter Player Data: Maxwell Glenn 121 3390 2689 40
        Maxwell Glenn - Dismissals: 81 Batting_Strike_Rate: 126.07 Batting_Average: 41.85
Do you wish to test another [yes/no]? yes
Enter Player Data: Kohli Virat 256 12471 13408 54
         Kohli Virat - Dismissals: 202 Battin_Strike_Rate: 93.01 Batting_Average: 61.73
Do you wish to test another [yes/no]? no

Program 1 is completed
```