Assignment 1

Objectives

- Practice implementing classes
- Review arrays and Strings
- Practice reading and understanding specifications
- Exposure to more complete testing

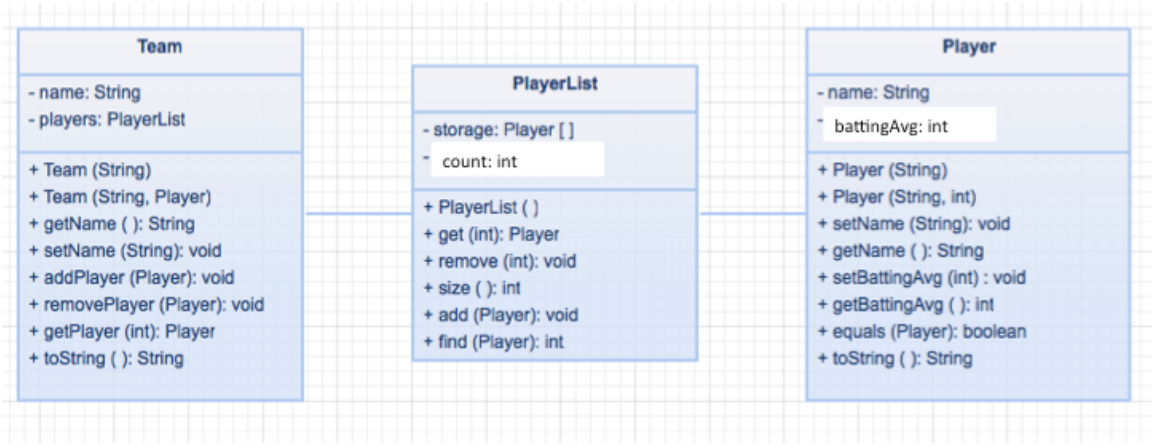
Introduction

You are creating a set of classes that can be used in a new "Fantasy Baseball Team" application. You are creating the infrastructure to allow a user to create their own *Fantasy Baseball Team* and add players to that team. In this assignment you will implement the classes that will be used by a person creating and managing their player roster.

You will create a Team class, which will store the team name and a list of players associated with the team. A player will store both a name and a batting average associated with the player (ie. Jose Bautista's batting average is 350).

Don't worry if you know nothing about baseball! This is the great thing about programming to a specification, you don't NEED to be an expert in the area, just follow the spec!

The following UML diagram provides the specification for the three classes involved, showing the attributes contained in each class and a list of the methods you will need to implement in each class. The `Team` class contains a `PlayerList` object, which is subsequently made up of `Player` objects.



Quick Start

1) Download `a1tester.java Player.java PlayerList.java Team.java`

2) Read the comments in Player.java and repeat the following until no errors are reported

   a) Implement ***one*** of the methods in Player.java, PlayerList.java, or Team.java

   b) Compile and run the test program a1tester.java

   c) If no errors reported by test program see the Grading section of this document

You should implement your solution in this order: `Player.java`, then `PlayerList.java` and finally `Team.java`.

Understanding the test program: a1tester.java

`a1tester.java` will test your implementations of `Player`, `PlayerList` and `Team`

One of the first things you should do after downloading the source code files is to compile and run the test program.

Compile the test program by typing:

```
javac a1tester.java
```

Run the test program by typing:

```
java a1tester
```

You should see the following output:

```
Player testing.
Failed test: 0 at line 53
```

The tester is reporting that your implementation is failing the very first test. This is hardly surprising, since you haven't written any code yet!

There are 31 tests and your goal for the assignment is to pass them all.

More details on what to do

You should start working in Player.java

The comments above each method to be implemented provide an unambiguous description of exactly what the method is to accomplish. Read the comments carefully before implementing the method.

First, implement the constructor that takes a single parameter:

```
public Player (String name)
```

Then implement `getName` and `getBattingAvg`.

Now, run the tester, you should have passed tests 0 and 1.

Then implement the second constructor:

```
public Player (String name, String battingAvg)
```

Run the tester again and you should have passed tests 2 and 3.

Now implement `setName` and `getName`. Run the tester and you should have passed test 4.

Now implement `setBattingAvg` and `getBattingAvg`. Run the tester and you should have passed test 5.

Now implement the `toString` method. Run the tester and you should have passed test 6.

Finally, implement the `equals` method. Remember that we don't consider the `battingAvg` when comparing `Players`, only the `name`.

Run the tester and you should have passed tests 8 and 9. Congratulations you've completed about 1/3 of the assignment!

Now move on to PlayerList.java. I suggest you implement the methods in the following order: `constructor`, `size`, `add`, `get`, `find`, and finally `remove`.

Finally, work on `Team.java`. This is the easiest to implement, since you will primarily be calling methods on the instance of `PlayerList` you allocate in the constructor. You shouldn't have to write more than a few lines of code in each method.

Submission

Submit your `Player.java` `PlayerList.java` and `Team.java` using conneX. **Please be sure you submit your assignment, not just save a draft**.

A reminder that it is OK to talk about your assignment with your classmates, and you are encouraged to design solutions together, but each student must implement their own solution.

We will be using plagiarism detection software on your assignment submissions.

Grading

If you submit something that does not compile, you will receive a grade of 0 for the assignment. It is your responsibility to make sure you submit the correct files.

Requirement	Marks
You submit something that compiles	1
1 mark for each test case you pass	Up to 31