The State Universtiy of New York, Korea
Computer Science

CSE 114 Handout 10: PS 7 April 26, 2019

This problem set is due at **11:55pm on Wednesday, May 1, 2019, KST**. Note that I make this one due on Wednesday so that we can have the next exam on the following day, May 2. Don't go by the due date that you see on Blackboard because it is in EST. Go by the one given in this handout.

- Please read carefully and follow the directions exactly for each problem. Files and classes should be named exactly as directed in the problem (including capitalization!) as this will help with grading.
- You should create your programs using emacs.
- Your programs should be formatted in a way thats readable. In other words, indent appropriately, use informative names for variables, etc. If you are uncertain about what a readable style is, see the examples from class and textbook as a starting point for a reasonable coding style.
- This problem set assumes you have installed Java and emacs on your computer. Please see the course web for installation instructions.
- You must use the command-line interface to run your programs. That is, you must use the `javac` and `java` commands to do this problem set. Do not use Eclipse yet.
- **Be sure to include** a comment at the top of each file submitted that gives **your name** and **email address**.

Submit the following files as a **single zip or tar file** on **Blackboard**. (Mac users usually create a tar file.) Multiple submissions are allowed before the due date. Please do **not** submit `.class` files or any I did not ask for.

```
Player.java
Team.java
UseTeam.java
(And others if you added more)
```

What Java Features to Use

For this assignment, you are not allowed to use more advanced features than what we have covered in Lectures 1 through 14 (April 24 lecture).

Partial vs. Complete Solutions

Please read what I said in PS 1 on this issue.

Naming Conventions In Java And Programming Style In General

Refer to the ones given in PS 1.

Problem (75 points)

In this problem we will model a couple of baseball teams. As you will see below, the entire program will consist of at least three classes: `Player`, `Team`, and `UseTeam` in `Player.java`, `Team.java`, and `UseTeam.java` respectively.

I am providing an almost complete implementation of `UseTeam.java` and your job is to complete the remainder of `UseTeam.java` and create `Player.java` and `Team.java` in such a way that `UseTeam.java` will work without modifying `UseTeam.java`. By that I mean that you may add more to `UseTeam.java` but you are *NOT* allowed to modify the methods that I have already included. In other words, you are not allowed to change the signatures of the methods that I have already used.

In the `[given]` link next to this handout, you will see two files:

- `UseTeam.java`: the use code of the classes that you will provide. Read my comments in this file carefully. *Additional problem requirements are specified in this file.*
- `output.txt`: a sample output generated by running the `main` of `UseTeam` in my model solution. Your completed program will have to generate the output that is *identical* to my sample output, except possibly the last several digits of a floating point number output. If you want to add other features *in addition to* my sample output, you are welcome to do so, but try to be brief for readability.

Restrictions: Do not use *public fields* in the classes that you design. Use *private fields* and provide *getters* and *setters* where appropriate so that accesses can be made to the fields through the getters and setters.

Hand in your `Player.java`, `Team.java`, and `UseTeam.java`. If your solution uses additional files, hand in the source files for them as well.