Homework 6

80 total points
Due: Sunday, October 9, 2011 at 9pm Pacific
Complete this assignment with a partner (see page 2).

I. Overview
In this assignment, you will use the machine library from class to build part of the bases-ball semantics. Specifically, you will design and/or build the following machines:

PitchCountMachine: This machine keeps tracks of balls and strikes. It should handle all Pitch events and need not handle any other kinds of events. It should be a CompositeMachine, composed of:

- balls — a CountingMachine of length 4
- strikes — a CountingMachine of length 3

The machine should have a way to compute the total number of pitches handled. The machine should get stuck if it receives more than four balls or more than three strikes or fewer than three strikes and fewer than four balls.

BasesMachine: This machine keeps track of which bases are occupied. You will design and implement this machine yourself. Your design and implementation is subject to only three constraints: (1) BasesMachine should be a Machine, (2) the machine should have eight states (one for each possible configuration of runners on base), (3) the machine should handle all Play events and need not handle any other kinds of events. The machine need not keep track of the score; it only updates the configuration of the bases according to plays. For example, if your machine is in the all-bases-occupied state and it receives a Double event, then the machine should transition to the second-and-third-bases-are-occupied state.

II. Working in Teams
It’s more difficult to split this work than with the previous assignment. True pair programming may be best. I’ll leave it to you to split the work fairly. You will be asked to provide feedback about the division of work.

III. Extras (for the group of three)
If you are in the group of three, then you must obey the following additional constraints:

- You should enhance the definitions of events to include fouls, and the PitchCountMachine should handle these events appropriately.

- You should enhance the definitions of events to include stealing and caught-stealing (which should be a GotOut), and the BasesMachine should handle these events appropriately. The machine should get stuck if a player tries to steal an occupied base.

IV. Evaluation
Your submission will be evaluated according to its documentation and its code.

Documentation (40 total points)
Provide the following documentation of your design and implementation:

(a) Provide good documentation in your code. (10 points)
(b) Answer the questions in evaluation.txt (30 points)
(c) Answer the questions in feedback.txt. (Participation points)

Code (40 total points)
Your code will be judged on accuracy (25 points) and elegance (15 points).
Materials
If you are in a new group, you should create a directory for your group, as described in HW4.

One partner should now copy the assignment files to their local repository:

```
cd dir/
group
svn copy https://svn.cs.hmc.edu/cs181b/fall11/hw/6 hw6
```

This should immediately commit their files:

```
cd dir/
group
svn commit -m "initial import" hw6
```

The other partner should now be able to check out the files, in their local repository directory by running

```
cd dir/
group
svn update
```

Each partner can also check out the machine library with the following commands:

```
cd dir
svn co https://svn.cs.hmc.edu/cs181b/fall11/hw/machine
```

Each partner should build the machine library:

```
cd dir/machine
scalac *.scala
```

Now you can compile your code against the library\(^1\):

```
cd dir/group/hw6/
scalac -classpath dir/machine basesball/*.scala
```

You should be able to run your code for the PitchCountMachine:

```
cd dir/group/hw6/
scala -classpath dir/machine basesball.PitchCountMachine
```

If you get a NullPointerException, you’re doing it right. You should also be able to run your code for the BasesMachine:

```
cd dir/group/hw6/
scala -classpath dir/machine basesball.BasesMachine
```

If you get no errors, you’re doing it right.

Both team members should now be able to collaborate, using their local repositories. Modify the files to complete your submission.

**Submitting Your Solution**
To submit, make sure you’re in the hw6 directory of your local repository, and type the following command:

```
svn commit
```

Committing is like voting in Chicago: you should do it early and often.

**Collaboration and Honor Code**
I expect you to abide by the Harvey Mudd Honor code. Your solution to this assignment should be produced only by the members of your team. You may discuss concepts and language design at a high level with any student in the class — I encourage you to do so! However, you may not copy solutions from anyone. If someone strongly influences your language design, be sure to cite that person.

If you have any questions about what behavior is acceptable, it is your responsibility to come see me before you engage in this behavior. I will be happy to answer any of your questions.

---

\(^1\) If the code doesn’t compile, make sure your classpath is accurate. Try using the full path to the machine code.