Homework 7

Software Design **Allen B. Downey**
Spring 2008

Due: Tuesday 11 March
The reading for this assignment is Chapters 16–18 of *How to think...*.

7.1 Poker!

The following are the possible hands in poker, in increasing order of value (and decreasing order of probability):

pair: two cards with the same rank

two pair: two pairs of cards with the same rank

three of a kind: three cards with the same rank

straight: five cards with ranks in sequence (aces can be high or low, but other than that, there is no wraparound).

flush: five cards with the same suit

full house: three cards with one rank, two cards with another

four of a kind: four cards with the same rank

straight flush: five cards in sequence and with the same suit

The goal of this homework is to use Monte Carlo simulation to estimate the probability of drawing these various hands.

1. Navigate into the local copy of your repository and run the following commands to download and unpack the files for this homework:

   ```
   wget http://wb/sd/code/poker.tgz
   tar -xzf poker.tgz
   ```

   It should create the following files:

   `Card.py` : A complete version of the `Card` and `Deck` classes in the book.

   `PokerHand.py` : An incomplete implementation of a class that represents a poker hand, and some code that tests it.

   `Poker.py` : An example program that shows a graphical representation of playing cards on a "table".

   `cardsets` : A directory containing images of playing cards in a variety of styles.

   As always, you should take some time to familiarize yourself with the code I give you before you start writing your own code.

2. If you run `PokerHand.py`, it deals six 7-card poker hands and checks to see if any of them contains a flush. Then it uses Lumpy to generate a UML object diagram and a UML class diagram for the program. These diagrams might help you understand how the program works.

3. `Poker.py` contains code that displays a table full of 7-card hands and uses the `has_flush` method to check for flushes. If you run it and press Deal a few times, you should see at least one hand with a flush.

4. Your job is to modify `PokerHand.py` and `Poker.py` to classify all poker hands. Add methods to `PokerHand.py` named `has_pair`, `has_twopair`, etc. that return True or False according to whether or not the hand meets the relevant criteria. Your code should work correctly for "hands" that contain any number of cards (although 5 and 7 are the most common sizes).

5. Write a PokerHand method named `classify` that figures out the highest-value classification for a hand and sets the `label` attribute accordingly. For example, a 7-card hand might contain a flush and a pair; it should be labeled "flush".

6. Modify `Poker.py` to use your classify method to set the label on each hand.

7. When you are convinced that your classification methods are working, the next step is to estimate the probablities of the various hands. Write a function in `PokerHand.py` that shuffles a deck of cards, divides it into hands, classifies the hands, and counts the number of times various classifications appear.

8. Print a table of the classifications and their probabilities. Run your program with larger and larger numbers of hands until the output values converge to a reasonable degree of accuracy.

9. As a JFFE, you can investigate the claim that certain hands tend to occur in clusters. For example, if you are at a table with four other players, and you have a flush, is it more likely that someone else at the table also has a flush? What about a full house?

10. As another JFFE, modify your classify methods to handle wild cards and see what effect wild cards have on the probabilities. Are there any changes in the probability ordering?

11. To turn in this assignment, create a directory named `hw07` in your repository and put a copy of `PokerHand.py` into it. When `PokerHand.py` is run as a script, it should print the card classifications and their probabilities.