COLBY COLLEGE
Computer Science Department

CS151 **Allen B. Downey**
Spring 1999

Assignment 11: Arrays of objects

Due: Tuesday 23 November

This assignment builds on top of the Card class from Homework 9. If you are not satisfied with your solution to that assignment, feel free to use my version from the class web page.

`http://rocky.colby.edu/cs151/solutions/hw9/Card.java`

Exercises from the book

In Chapter 11 there are several methods that I sketch but don't fill in completely. You should type them in and clean them up:

1. Type in `sameCard` and test it. Make sure you are clear on the difference between shallow and deep equality. If not, ask.
2. Use the code in Section 11.6 to write a method called `buildDeck` that creates and returns a fully-populated array of cards.
3. Write a method called `printDeck` that takes an array of cards and prints them one per line.
4. Type in and test `findCard`.

Shuffling and sorting

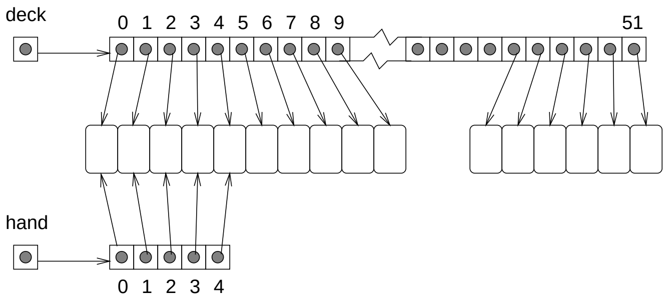
1. Write a method called `swapCards` that takes a deck (array of cards) and two indices, and that switches the cards at those two locations.

   HINT: it should switch the references to the two cards, rather than the contents of the two objects. This is not only faster, but it makes it easier to deal with programs in which cards are aliased.
2. Write a method called `shuffleDeck` that uses the algorithm in Section 12.1. You can use the `randInt` method from the solutions to Homework 10. Look it over carefully, it may be different from what you wrote.
3. Write method called `findLowestCard` that uses the `compareCard` method to find the lowest card in a given range of the deck (from `lowIndex` to `highIndex`, including both).
4. Write a method called `sortDeck` that arranges a deck of cards from lowest to highest.

Histograms of cards

Let's say that you are given a small array of cards. Say it contains five cards. Say it's called `hand`.

1. Write a method called `drawHand` that takes an array of Cards as a parameter and that returns a new array of Cards that contains (references to) just the first five cards in the deck. Here is what the state diagram for this method should look like:



2. Use `shuffleDeck` and `drawHand` to generate and print four random poker hands. Did you get anything good?

3. Now imagine that you want to generate random hands and check, automatically, if you have a good hand. In case you don't play poker, here is what I mean by a good hand (in increasing order of value):

   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
   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

   Write a method called `isFlush` that takes a hand (array of cards) and that returns a boolean indicating whether the hand contains a flush.

4. Write a method called `isThreeKind` that takes a hand and returns a boolean indicating whether the hand contains Three of a Kind.

   The following part of the assignment is optional, but fun:

5. Write a loop that generates a few thousand hands and checks whether they contain a flush or three of a kind. Estimate the probability of getting one of those hands.