Introduction to Computer Science Homework 5
Assigned Oct. 2, 2012
Due Oct. 9, 2012

Background:
This homework is intended to give you experience with using classes for data and procedural abstraction as discussed in the Oct. 2 and Oct. 4 lectures. It also has some review questions for the midterm exam.

Problem:
1. Create a Java class, TicTacToeBoard, that implements a tic-tac-toe gameboard of arbitrary size. It should look something like this:

   ```java
   public class TicTacToeBoard {
      public static final int SQUARE_INVALID = 0;
      public static final int PLAYER_NONE = 1;
      public static final int PLAYER_X = 2;
      public static final int PLAYER_O = 3;

      public TicTacToeBoard(int edgeSize) {
         ...
      }

      public boolean makeMove(int x, int y, int mark) {
         ...
      }

      public int getContents(int x, int y) {
         ...
      }

      public int victory() {
         ...
      }
   }
   ```
The argument to the constructor specifies how long each side of the gameboard is. For example, if `edgeSize` is 4, then player X has won in this configuration:

```
0 0 0 0
X X X X
X X X X
```

In general, a player has won if they fill a complete row, column, or diagonal with their mark.

`makeMove` makes a mark on position `(x, y)`, where each coordinate is in the range 0 to `edgeSize`. The mark should be either `PLAYER_X` or `PLAYER_O`. If the position and mark are legal, no move has been made on that position, and no player has won yet, it should return `true` and record the mark. Otherwise, it should return `false` and not change the board state.

`getContents` should return the mark at the specified position: `PLAYER_NONE`, `PLAYER_X`, or `PLAYER_O`. If the coordinates are invalid, it should return `SQUARE_INVALID`.

`victory` should return `PLAYER_X`, `PLAYER_O`, or `PLAYER_NONE` indicating which player, if any, has won.

You can implement the class in a manner of your choosing. Please don’t use individually-named `int` variables, one for each position, though. Something with arrays would be a good choice.

(There are fewer than `Short.MAX_VALUE` possible board configurations for many board sizes. If you manage to solve the problem storing your state in a single `short`, the points you get for cleverness will balance out the points you lose for bad taste.)
2. Test your class. I am not going to tell you how to test your code. Perhaps you want to write a main method that asks for moves and simulates a game. Or, perhaps you want to write a test script that creates boards in various interesting configurations and checks that the required methods behave as expected.

The only thing to be turned in for this homework is a single Java file, submit/hw5/TicTacToeBoard.java.
Review questions

1. What is the value of \( c \)?
   
   \[
   \text{byte} \ a = \text{(byte)} \ 0xFE; \\
   \text{byte} \ b = \text{(byte)} \ 0x10; \\
   \text{byte} \ c = \text{(byte)} \ (a * b);
   \]

2. What does this code print, exactly?
   
   ```java
   public class C {
     public static int func(int k) {
       int a = 1;
       int b = 1;
       for (int r = k; r > 0; r--) {
         k = a + b;
         a = b;
         b = k;
       }
       return k;
     }
     
     public static void main(String[] args) {
       System.out.printf("func(+3) = %d\n", func(+3));
       System.out.printf("func(-3) = %d\n", func(-3));
     }
   }
   ```

3. Suppose you are given a java \text{float} as a sign, exponent, and fraction. Remember that it has the value \((-1)^s \cdot 2^e \cdot 1.f\). Under what circumstances is it a nonzero integer?

4. Write the code for a Java class with a single method, \text{largestNegative}, that takes a non-null array of \text{double} as an argument and returns the largest negative value in the array. If there are no negative values in the array, it should return \text{Double.NEGATIVE_INFINITY}. 