Homework 1 (25 points)

cs1007 - Object-oriented programming and design in Java
Prof. Shlomo Hershkop
Dept of Computer Science
Columbia University
Spring 2006

DUE: February 12 at 11pm electronically.

It sometimes seems that some people would find 1007 a little boring. In order to put such worries to rest we will be programming a game for your first homework.

Mastermind™.

you read that right!!

This is a game you will program the computer to play against yourself or any of your friends. In this game, the computer randomly picks X numbers (example X=3), each in the range 1 through N (example N=6). Think of this as 3 slots, with 6 possible values per slot. You, in turn, keep guessing what the 3 values are until you get them correct. Each time you guess, the computer gives you hints about how close you came by telling you two things:

1. How many slots you got the correct value for. (a single number)
2. How many of the values you guessed appear somewhere in the computer's choice. That is, how many correct values you got, even if in the wrong slot.

For example, if the computer picks (1,2,3) and you guess (1,4,2), then the computer would tell you that you're WRONG, however it would also respond with "1" and "2", respectively. Note that the 1 you guessed counted in both parts of the hint.

Each of your guesses can only count once in the second part of the hint. For example, if the computer picks (2,2,2) and you guess (1,2,3), the hint is "1" and "1". However, if the computer picked (1,2,3) and you guess (2,2,2), the response should be "1", "3".

There is a limit to how many guesses you can guess (call this G). WE can choose G = 20 to make it a reasonable game. You will need to generate new games using a random number generator. Use the java.util.Random class to get random numbers.

Hint:
```
Random generator = new Random();
int n = generator.nextInt(7);
```
Note: the above call gives you a random number between 0-6, how would you adopt it to give you a random number on the range of 1-7?

Here are requirements and suggestions for your implementation:

1) The game ends when the user guesses correctly all the numbers in addition your program should tell the user how many guesses it took (the less guesses the user took, the more likely it is that the user is either really smart, or cheated somehow). You need to end the game if

the number of guesses are used up, and the user has not correctly guessed the sequence.

2) The parameters of the game (G,X,n) should be set to some default settings, but should be changed based on command line arguments.

   It must be easy to change the game to any number of slots and any number of possible values per slot and any number of maximum guesses, by simply passing the following command arguments:

   Java MindGame G=20 X=4 N=5
   that is you get 20 guesses, with 4 mystery numbers, and will be using 1-5.

3) Code for all assignments should be well organized and well commented, using basic JAVADOC conventions.

   Hint: Eclipse can automatically generate comments for any method, if you program the method and the type /** and hit enter, it will create a minimum javadocs required for the particular method.

4) You should have the minimum following program structure (can add as much as you would like)

   MindGame.java - The main program where you read in parameters (if necessary, and print out error messages on weird parameters or bad command line arguments. In addition the main game should be started here.

   Guess.java - Class to represent a game move (that is X random numbers)
   It should define the following methods:
   public Guess(int X, int N);
   public boolean isMatch(Guess othergame);
   public int getCorrectPositions(Guess othergame);
   public int getCorrectGuesses(Guess othergame);

   Hint: don't forget when comparing 2 Guess objects to make sure they are the same number of guesses (Will need to define more methods for this)
   Hint Hint: getXSize() maybe?

   GuessException,java - Exception for doing something illegal in the game, easy example: not inputting numbers for a guess, or the wrong number of numbers. Remember to throw and catch exceptions correctly.

5) Each class in your code needs to be part of a MasterMind package.

6) At the conclusion of a game, you should have a choice of playing again or quitting.

7) **EXTRA CREDIT**: After you get it working, incorporate graphics in some way, using colors instead of numbers for slot values. The graphics window can show the user's guesses, and finally the correct answer.

WARNING: we can not be held responsible if you somehow get addicted to playing MasterMind™

You will be graded on the following points:

1. The program compiles and executes successfully and correctly (10 points).
2. The program contains all of the required programming elements listed in the program specification above. (3 points)
3. The program is well organized, as far as object oriented design. That means classes, are well defined and divided by task. It will help you if you create some documentation in the class files to sketch out your ideas before starting. (2 points)
4. A README file is included with a list of all files and their purposes and instructions on how to run your program. (2 points)
5. The program is well commented, and documented in javadoc. (5 points)
6. The code is nicely formatted (easy to read visually). For eclipse you can format the code, for emacs there is a format command to format the code. (3 points).

Submission instructions are on the class website. Please post general questions to the class web board, and learn to use any IDE Debugger.

Hint: Start Early and have fun!!!