CS101 Project:
Mastermind with Six Digits

Description of the Game Rules

This is a more complicated version of the well-known game of Mastermind. Here is how it works:

The game is played between two players. One of them is the code maker, and the other is the code breaker. The code maker creates a six digit number, where *leading zeros are allowed,* as opposed to the usual convention. For example, the code could be:

074311

The code is not known to the code breaker. The code breaker then comes up with a guess, and suppose they produce the number:

111000

Now, the code maker compares the guess and the actual code, and provides *feedback*. The feedback consists of two numbers: The first indicates "direct hits", that is, correct digits in the correct position. The second number is "indirect hits", that is, digits that are in the code, but are in the wrong position in the guess. For this example guess, the feedback would be:

0, 3

There are no direct hits, that is, no digits match exactly between the code and the guess. There is a zero in the wrong place, and there are two ones in the wrong place. That makes a total of three indirect hits.

The code breaker tries to take advantage of the information from the feedback, and produces another guess:

223344

Now the feedback is:

1, 1

This is because the 3 is a direct hit, and there is an indirect hit for 4.

The game ends when the feedback is:

6, 0

The Goal of the Project

The goal of the project (probably contrary to your expectations) is to make the computer play as the code breaker, and the user as the code maker.

So, in essence, as the game starts, the computer presents you with a guess immediately, and expects feedback. You compare it to your secret code number (which you never enter into the computer) and give two numbers as your feedback. Then, the computer gives you another guess. At the end, the computer guesses your number.

The conditions are as follows:

- The computer should play efficiently. You may or may not come up with the best algorithm to play the game, but it should not be random guessing by the computer.
- If you give inconsistent answers (after all, you can respond 0, 0 to every guess) the computer should be able to detect this, and tell you you are being inconsistent.
- The input and output should happen on the console. You need not bother programming any fancy user interface. If you do, it will not earn you any extra points.

Required Work

Analysis

You should perform an analysis of the problem, and write it up. Specifically, you should explain what kind of algorithm you are using, and how efficient you think it is. Proofs are not necessary, but would be appreciated.

Design

Once you have analyzed the problem and come up with an algorithm, you should document your implementation design. What components your program will have, why, and how it will be organized.

The Program

You can write your code in any IDE of your choice, but the preferred IDE is IntelliJ IDEA. Clean and clear code is important, as is obeying naming and coding conventions observed throughout the course.

The Presentation

You should be well prepared as a team to present your analysis, design, program and perform a demonstration within 15 minutes. 15 minutes will be a sharp cut-off. Anything you have not presented in the alloted 15 minutes is non-existent.