HW #5: Client-Server Programming

The goal of this assignment is to practice writing networked programs. As always, you are welcome to adapt any of the code from the text or lecture to help you on the assignment. You are to build a server than lets humans play the game ‘Mastermind’ (and a test client). Your server must accepts certain commands:

Command	Usage	Purpose
newgame	newgame	Tell the server to start a new game
cheat	cheat	Display the correct answer (but do nothing else)
guess	guess color1 color2 color3 color4	An attempt to guess the secret code
history	history	Display all moves and responses in this game
quit	quit	Client wishes to disconnect

The server should start and listen for a connection. Once a connection is established the server must parse each single-line command and return textual output which might consist of multiple lines. The server must always return at least one line of text in response to a command. The command *newgame* tells the server to create a new random code to be solved (and to reinitialize any needed data structures). The server should respond to the *cheat* command by returning a string with the solution to the secret code, however, the game continues unaffected. The *guess* command is how a user attempts to guess the secret code; the server responds with a string such as “2 black and 1 white”. The *history* command should return multiple lines containing the status of the current game with one turn per line. *Quit* instructs the server to close the socket connection immediately. Error checking is required -- in particular, the server must not crash when given improper input, instead, an error message should be returned and the improper command should be ignored. The game ends when either the client guesses the secret code or has made 10 incorrect guesses; after the tenth or final guess the server should print a response, but must not terminate the connection automatically. More than 10 guesses should not be permitted by the server, but the newgame/cheat/history/quit commands are each valid after 10 incorrect guesses. When guessing, colors should be specified using the letters: {RGBYWO; red, green, blue, yellow, white, and orange}.

Your code should be clear and concise. Unlike other assignments for this class, you must write a text-only standalone Java application, not an applet.You should hand in the source code for both client and server programs and screen-shots or scripts of the running programs. You should also install your *.class* files in a publically readable location on the university Unix system so that I may test your program. On the first page of the assignment you should legibly specify command lines that invoke your programs. I strongly suggest letting your programs take a port number argument. To be eligible for full credit (read for an ‘A’), the server should be multithreaded using appropriate constructs in Java. In particular this implies that while running, your server should accept connections from multiple clients -- each can play a separate game. The server should not halt after a client finishes a game (or quits), since other games may still be in progress. I suggest you get a single-threaded version running first before writing a multithreaded server

The rules for the game of Mastermind are available online at:

http://www.pressmantoy.com/instructions/instruct_mastermind.html
http://www.centralconnector.com/GAMES/mastermind.html (no images)

You do not need to keep track of a ‘score’, or number of games and guesses, since there is only a single player.

The following method reads in a line of text from the keyboard:

```
public String getLine() {
  try {
    DataInputStream dis = new DataInputStream(System.in);
    return dis.readLine();
  }
  catch (IOException ioe) { System.out.println(“Problem reading from keyboard”);}
}
```