Course Project
Due: Mon. 11/23/09

Project Description

Implement an algorithm to solve Mastermind. Mastermind is a code-breaking game for two players. The description of Mastermind is given on the wikipedia page http://en.wikipedia.org/wiki/Mastermind_%28board_game%29.

Your task is to write a function that computes a guess (query) in the form of a sequence of K pegs, where each peg can have 1 out of C colors. The query will be answered with a number of pins. Black pins correspond to pegs of the correct color at the correct location, while white pins correspond to pegs of the correct color at the wrong position. This feedback should be used to then generate a new guess etc. until the code is broken. Clearly, you should remember your previous guesses and the responses of the system to improve compute a new guess. Your algorithm will be run for 15 minutes on a preselected collection of problems with varying values for K and C (values typically between 0..9). The algorithm that cracks the most codes in 15 minutes wins the competition.

We will provide the source code of the main function that generates encoded problems and returns feedback to your guesses in the form of black and white pins. You can use this to test your algorithm before you hand it in. For grading purposes we will be using a different set of problems. Note: your code is required to compile and run without issues on the provided test-bed. If not, you will receive a very low grade for your project.

Technical Requirements

1. Programming Languages

You should use Java to implement the CodeBreaker class. We will compile and run your code, therefore, make sure you are not using any non-standard libraries. For example, it should use only objects/function supported by the JRE.

Your algorithm has to run on Windows or, if you write your code under linux/unix/bsd, it should run on ICS unix computers.
2. Code Interface

The class you'll provide should have the following interface so that we can create an CodeBreaker object and call its method guess:

```java
class CodeBreaker{
    public ArrayList<Integer> guess(int K, int C, int nBlackPegs, int nWhitePegs);
}
```

description:

- **K**: the number of code pegs in a row
- **C**: the number of colors
- **nBlackPegs**: the number of black key pegs for the previous guess, e.g., nBlackPegs=2 means there are 2 black key pegs in our feedback to your previous guess. 0<=nBlackPegs<=K. But when starting a new game, nBlackPegs=-1.
- **nWhitePegs**: the number of white key pegs for the previous guess. 0<=nWhitePegs<=K. But when starting a new game, nWhitePegs=-1.

Return value: a new guess. It should be an array of K integers. The color of the peg is encoded by a 0-based integer. Each element in the array specifies your guess about the color of the code peg in the corresponding position. The index is from 0~K-1, and the value of the element is from 0~C-1. E.g., if `returnValue.get(3)=5`, it means your guess of the 4th peg in the code has the 6th color.

In every game, we call your `guess` method to make one guess at a time until codePeg is the same as the code or the number of guesses>N or the running time t>T.

Submission Instructions

Submit all of your code, and a report.

First few lines of your program should include (as comments) your name, student ID, and necessary notes about compilation and execution of your program, if any.

Don’t submit plain code, include some comments.

If you used any non-standard library, please upload it with your code too.

In the report, give a short description of your implementation of the algorithm. Also, provide some overview of results. For example, you can discuss how different values of K,and C influence results. You can also study the time and space complexity of your algorithm. In general, the more effort you invest in this project you higher your grade. Your report should not be longer than 5 pages (including references, figures etc.) using 12pt fonts.
The bare minimum to submit is a working class that can solve simple Mastermind instances and a report that describes the strategy employed by your algorithm. Also, upload some of the examples you used to test your code.

The report is required to be in a doc, txt, or a pdf format.

Zip all of your code, examples, and report in a zip file titled <lastname_studentid>.zip (for example: smith_1234567.zip). Upload your zip file to EEE dropbox named PROJECT, by the deadline.

**Grading**

Your grade consists of two parts: the experiment performance and your report.

We will test your code by randomly generating instances of Mastermind codes with random parameters such as K and C. The grade is based on the success rate, and the average number of guesses for each successful round. Only in the case that there are very good reasons beyond your control that have caused your code to fail to compile on our computers will we give you a chance to change your code, (however the algorithm should not fundamentally change). In all other cases of failure your grade will be very low.

The second part is your report. If you provide us with a solid, well-written report in which you analyze your algorithm intelligently (using graphs and other visualizations) you will receive extra credit. Remember that it is nice to have a fast algorithm, but it is more important that you perform a scientific analysis of your algorithm and show your ability to think academically.

**Working in Groups**

You are allowed to work in groups of no more than 5 students. However, each student will have to implement and submit their own code, and write their own report. Reports and code cannot be copies of each other.

**Academic Dishonesty**

If it is determined that you have copied code either from fellow students or from the web you will receive 0 credit for your project. You must implement your own code, do your own analysis and write your own report (even though you can consult in groups of at most five members to share ideas of how to improve your algorithm).

Have fun!