|  |  |  |
| --- | --- | --- |
| **LAB101Assignment** | **Type:** | **LongAssignment** |
| **Code:** |  |
| **LOC:** | **120** |
| **Slot(s):** | **5** |

**Title**

Mastermind.

**Background Context**

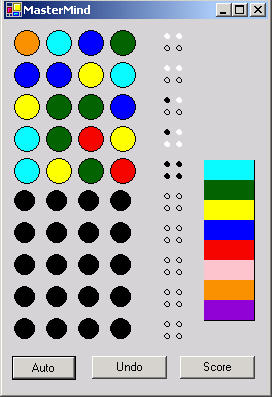
Mastermind or Master Mind is a code-breaking game for two players. The modern game with pegs was invented in 1970 by Mordecai Meirowitz, an Israeli postmaster and telecommunications expert.It resembles an earlier pencil and paper game called Bulls and Cows that may date back a century or more..

**Program Specifications**

In the popular game Mastermind, one player creates a secret code of four pegs, each of which can be chosen from one of six colors. (The number of pegs and colors may be different than this in different versions, but your implementation will use these values, but be extendible, very easily, to other values. Also, a color can be reused, thus there really are six choices for each of the four pegs.) The other player then has to guess the color of each peg, with the order mattering.

The player who made the secret code then has to give feed back to the player guessing. This feedback is in the form of white and black pegs. A black peg means that the guesser has chosen the correct color in the correct slot. These are first “calculated” and awarded. Once these are counted, these pegs are ignored. Then the white pegs are awarded. These are for pegs that are the correct color but are in the incorrect slot. There is no “double dipping” of pegs in the response, so the sum total of white and black pegs the guesser can receive is four, and no one peg in the guesser’s answer may earn them more than one black or one white peg.

Here is a picture of a Mastermind board:



A framework has been written for you to fill in. The main has been completed and nine functions have been specified for you to fill in. Each function comes with a pre-condition and post-condition.

A function pre-condition is the set of specifications that must be satisfied in order to call the function. If these conditions aren’t met, then the behavior of the function is unpredictable. However, if the conditions are met, then the function must uphold the listed post-conditions. (For example, if the sqrt function is given a positive real number, then it is required to return the square root of that positive real number.) The post-condition explains what you must do in the function while the pre-condition explains what you can assume without checking.

The most difficult function in the program is the numWrongPlaceMatches function. In this function, comments have been left to discuss the various steps involved in solving the given task. For the rest of the functions you must simply use the pre-conditions and post-conditions given to write the function.

***Function details:***

* A greeting will be printed in the beginning, exactly like the one printed in each of the samples. From that point the program will respond to each guess with a statement of the following form:

*You have X perfect matches and Y imperfect matches.*

where X is the number of perfect matches (right color, right spot) and Y is the number of imperfect matches (right color, wrong spot with no double counting).

* At the end of the game, if the user wins, the number of guesses they needed as well as the amount of time they took will be printed in the following format:

*You have won the game in X turns and M:SS!!!*

where X is the number of turns taken and M:SS is the amount of time in minutes and seconds. Here is how you can print out a number (num, in this example) so that a zero precedes it if it’s one digit long instead of two:

*printf(“%02d”, num);*

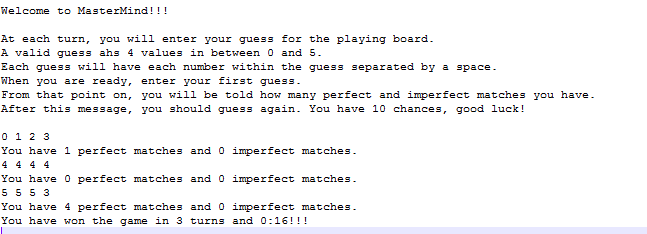
* If the user loses, then a message stating so along with a printout of the correct game board will be given using the following format:

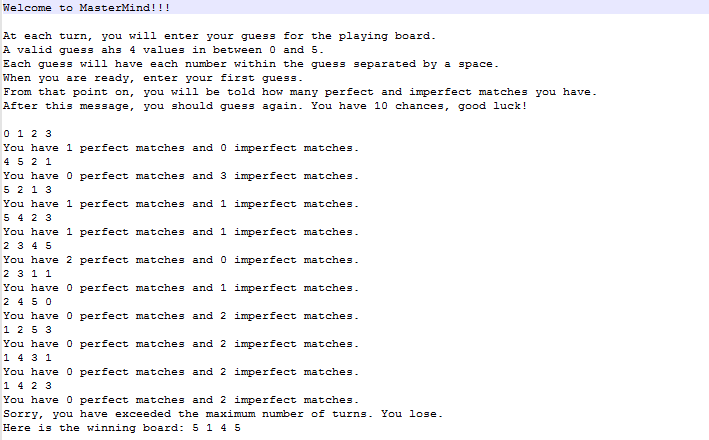
*Sorry, you have exceeded the maximum number of turns. You lose.*

*Here is the winning board: A B C D*

where A, B, C and D are the numbers (in order) of the secret pegs..

***Expectation of User interface:***





**Guidelines**