Project 1

For our first project we are going to work on a shell script game to play Mastermind.

Details

Mastermind is a game of code creation and code breaking. The basic games is played this way.

The code maker selects 4 symbols out of 6 and sets the code. The symbols can be repeated, but no spaces may be left blank. The code breaker then has 10 tries to try and break the code. They do this by guessing what the code it. On each guess the code maker indicates how many symbols have been guessed correctly and are in the correct location or how many symbols are guessed correctly but not in the correct location.

For example, we are going to use the symbols: #, $, %, &, *, and @. If the code maker has selected this as the code: # & * @. This is how the game might be played:

First guess: # $ % & So then the code maker would indicate the following: X O
Second guess: $ % & * So then the code maker would indicate the following: O O

In this example, I've chosen to use X to indicate that a correct symbol was placed in the right place, that was the # for the first guess. Then the O is indicating that a correct symbol was chosen but it was not in the correct place, that was the & in the first guess.

For the second guess since the $ was removed, we lost the X but now the * has been guessed but both it and the & are in the wrong locations so two O's have been shown.

If a symbol is just plain ole wrong, then nothing is displayed, hence only two symbols being displayed for both the first and second guesses.

The game ends when either 10 turns are up or when the code breaker guesses the code.

You aren't required to use X and O but some symbols should be chosen and there should be a key or a help that indicates what they mean.

Requirements

- You are required to use the symbols shown above and here: #, $, %, &, *, and @. They have been chosen because many of them are special symbols to the shell and therefore are a real pain to work with. You'll learn how to escape them to make the game work right.
- You must display each guess each time and for each guess display how many choices where both right and in the right spot and just right.
- Your game needs to be able to play with the computer as the code maker, i.e. a 1 player game.
- Your game must be ten turns by default.

Other than that you can do what you want. You may display the moves in any fashion you feel looks appealing and you may be as verbose or terse as you choose during the game play.

- You may make the game a two player game.
- You may allow for 8 or 12 turn games, or any other number of turns for that matter.

Design

You can and should use all the good programming design you have learned in your courses so far. You are encouraged to try arrays, loops, selection and functions in your script. I encourage you to make your script as modular as you can. It's my opinion that breaking a problem down into functions allows you to make for more robust programs.

Technical Requirements

Your program must be able to run in a Linux shell. Unless otherwise stated, the bash shell will be assumed