Mastermind

Mastermind: Program Description

Create an Scratch program that will simulate the mastermind game. The actual Mastermind game uses 4 pegs in 6 different colours machine. To start only use 3 pegs and 6 colours. **ONLY IF**, you get 3 pegs working, add a fourth peg.

After the user attempts to guess the answer, the number pegs that have the correct colour and the number of pegs that are in the correct position and have the correct colour is given.

Once you have done this, repeat in AppInventor.

New concepts: (optional)

These concepts do not need to be used to solve this problem, but could greatly reduce the amount of code required to solve the problem.

- Lists

Code will be given to you to help you through this process.

Required Screen Elements

- three sprites to represent each peg that makes up the answer
- three sprites to represent each peg that makes up the user's guess
- display the number of attempts made (allow up to 10 guesses)
- display a win screen - when user’s guess matches the answer in both position and colour.
- a check guess button - to determine the number of correct colours and positions
- a restart button - allows the user to start again.

HINT:
While you are working on your solution,

- display the answer for testing purposes
- be able to select/change the answer, this will allow easier testing.

Suggestions to add:

- show a history of all guesses and their resulting condition
- once your solution works, hide the answer

Suggestions for grade 11 or 12 CS students

- add AI to have the computer select the guess, if done correctly, the computer will always find the solution within 5 guesses.

Mastermind - Version 1 - Phase 1 Three Answer pegs and Three guess pegs, no duplicate colours

- when the game starts or restarts, the answer are three different colours (add the random colours after you have the CheckGuess working
- make sure the answer does not contain duplicate colours

Logic to count correct colour and position

```
if answer1 = guess1
   add 1 to correct colour and position
if answer2 = guess2
   add 1 to correct colour and position
if answer3 = guess3
   add 1 to correct colour and position
```

Logic to count correct colour

```
if answer1 = guess2 OR answer1 = guess3
   add 1 to correct colour
three more statements similar to this
must be added
```

Mastermind - Version 1 - Phase 2 Three Answer pegs and Three guess pegs, with duplicate colour

Be able to show and explain why the above logic does not work when duplicate colours are involved.

You may use your own logic to make solve the duplicate colour problem. If you need an idea on how to fix this you can use the following approach:

- use a variable for every peg (both the answer peg and the guess peg) to keep track of whether or not a match was made using the peg.
- when a match is made, set the value of the variable to zero
- each time user press the Check Guess button, these values must be reset to 1

Logic to count correct colour and position

```
if answer1 = guess1
   add 1 to correct colour and position
   set answer1value to 0
   set guess1value to 0
if answer2 = guess2
   add 1 to correct colour and position
   set answer2value to 0
   set guess2value to 0
if answer3 = guess3
   add 1 to correct colour and position
   set answer3value to 0
   set guess3value to 0
```

Logic to count correct colour

```
if peg1value = 1 and guess2value = 1 and answer1 = guess2
   add 1 to correct colour and position
   set answer1value to 0
   set guess2value to 0
if peg1value = 1 and guess3value = 1 and answer1 = guess3
   add 1 to correct colour and position
   set answer1value to 0
   set guess3value to 0


two more pairs of statements similar to this must be added for the 3 peg solution
```

Mastermind - Version 2 - Phase 1 Four Answer pegs and Four guess pegs, no duplicate colour

- add a fourth peg to your working three peg program

Mastermind - Version 2 - Phase 2 Four Answer pegs and Four guess pegs, with duplicate colour

- add a fourth peg to your working three peg program

Mastermind Problem – version 3 Phase 1 using lists and cloning

This could be done with lists and without cloning
(It also may be able to be done with cloning and without lists.)

- a global variable will be needed to count the number of clones made
  - this will also be used for the pegID of each clone
- clone the answer peg
  - each answer peg will need a local variable for the pegID
- clone the guess peg
  - each guess peg will need a local variable for the pegID
- the pegID is used to keep track of the position of the peg in the answer or guess
- use a list to keep track of each of the following:
  - the colour of each answer peg
  - the colour of each guess peg
  - again, the pegID is used to keep track of the value in the list