Twisted Mastermind

INARI

February 2022

MasterMind is a logic board game from the 70's in which you have to correctly guess a random secret code in a determined number of guesses. This game can be played by two players, the code-maker and the code-breaker.

- The code-maker creates the secret combination, composed by a sequence of 4 colored pegs.
- The code-breaker makes a series of guesses, each guess composed in the same way by 4 colores pegs. After each guess, the code-maker gives feedback to the code-breaker to see how close you got to the real secret code.

Each feedback is composed of two numbers, represented by white pegs and black pegs, and they tell you how your guess and the secret combination compare. Black pegs tell you how many pegs of your guess you have correct in color and position, and white pegs tell you how many pegs of your guess you have correct in color but not in the proper position. Example of a gameplay:

1. The code-maker has created the secret sequence `Red, Blue, Blue, Red`
2. The code-breaker has ten tries to guess the code. In his first attempt, the player guesses `Red, Yellow, Green, Blue`
3. The code-maker compares the guess with his solution, and gives him 1 black peg and 1 white peg. 1 Black peg because he guessed correctly the first peg (The red one) in color and position, and 1 white peg because he guessed a Blue peg but in the incorrect position. The code maker never tells you which color or which position you correctly guessed (which is the real fun of the game after all)
4. The code-breaker makes another guess now, and tries the following code `Yellow, Yellow, Green, Green`.The player receives no pegs as feedback because he matched no color against the solution.
5. The gameplay continues until the code-breaker guesses the code correctly correctly or the code-breaker has reached the maximum number of tries allowed without reaching the solution.

To get a simple idea on how this game is played and how it works, you can check out this link and give it a shot: http://www.webgamesonline.com/mastermind/

Here you have several examples to see how the algorithm to compute the feedback works:

code	guess	black pegs	white pegs
RGGB	RGGB	4	0
RRRR	BYOB	0	0
GBBR	GBRB	2	2
BBBR	RBGG	1	1
RBGG	BBBR	1	1
BBBR	BBRB	2	2
WBWB	BWBW	0	4
OOOW	OWWW	2	0

Legend 1: (R)ed, (B)lue, (Y)ellow, (G)reen, (W)hite, (O)range

What we ask you is to implement a simple REST API in which a secret code is computed, the user introduces a guess code in the form of a string like "RRBB" and then this guess is compared against the secret solution and the feedback is returned.

We need the following endpoints (although you can add more endpoints if you feel so):

1. A POST endpoint to create a new game.
2. A GET endpoint to retrieve the current state of the game, with the information that would be required for a frontend client to render the board appropriately.
3. A POST endpoint to create a new guess for an on-going game.

We require as well that this small API to be "production-ready", meaning you follow the same practices you would follow as if you were implementing and deploying these features in a real company. You can implement this in any language you like (python, Java, .Net, Go, Scale, etc.), with any framework you wish.

Also, to retrospect with the code of the project, some questions:

- What is your favorite package manager and why?
- What architecture did you follow for this project?
- Do you know and understand UML? What would the class diagram of the domain exposed in the mastermind game be like for you?
- What tools are you aware of to handle things like code styling and consistency, formatting, typing...?
- Is there anything you would like to comment about the project, this small exercise, etc?

Feel free to ask as many questions as you require. Here we offer you some tips:

- Be mindful of the concept "production-ready". Our real intention is not only how good you are at algorithms, but how does your code "breathe" as well.
- No need to look at the Knuth's algorithm.
- It is good and recommended to deliver the solution in a git repository. We will like to take a look to the commits.
- Although not required, the use of Docker and Docker-Compose (or other automate deployment tools) will be a bonus for us and will probably ease your life.
- You can reutilize third party libraries that you think tha can be useful.