CSC120 Fall 2023 Project: Mastermind

1 Introduction

The goal of the project is to write implementations of Mastermind, a two-player game where each player chooses a secret "codeword" and takes turns to guess the opponent's codeword before the opponent does. The permissible codewords in Mastermind are the 4-digit integers all of whose digits are between 1 and 6 (e.g., 2123 and 6543).

When a player receives a guess for her codeword, if the guess is identical to the codeword, the player loses the game immediately; if the guess is not identical to the codeword, the player provides a hint called "feedback" and the opponent takes her turn. The players can analyze the feedback to determine the next guess.

While in the real-world Mastermind, the two players take turns and provide their guesses, the code you will produce will be one-sided.

This is how we determine the feedback. Let us use 0..3 (from the highest to the lowest) to refer to the four positions of the codewords and guesses. The feedback is given by two numbers, the "hits" and "misses." Both are nonnegative, and their sum is at most 4.

(a) The number of "hits" is the number of positions p at which the digit is identical between the codeword and the guess.

(b) The number of "misses" is the maximum number of additional "hits" that can be produced by reordering the guess's digits where hits are currently not occurring.

For example, if the codeword is 1123 and the guess is 2213, the guess has 1 "hit" (the last digit "3") and 2 "misses" (the permutation 1223 generates two additional "hits"). You can combine the two numbers as `10 * "hits" + "misses"`.

Part 0 (1%) Due at End of Day, Monday, October 24, 2023
Practice submitting an *arbitrary* Java source file and a video file.

Part 1 (10%) Due at End of Day, Monday, November 6, 2023
Write a one-sided program where the user finds the codeword the program selects. The program must check the validity of each guess, provide feedback, permit an indefinite number of games, reveal the codeword if asked, report the round number in each game, and terminate the present game when receiving 0 as the guess.

Part 2 (9%) Due at End of Day, Monday, December 4, 2023
Write a one-sided program where the program finds the codeword the user selects. The program may receive the codeword from the user, only for computing the feedback. The user may enter the feedback herself or let the program compute it. If the user chooses to enter the feedback and some feedback is incorrect, the game may stop finding no more candidates.

You may form a team of **at most three students from the class**. Submission of work will be individual. In the submission, state your teammates.

2 Execution Examples

Here is how Part 1 may run. Presented in blue are the places where the user's input changes the course of action.

```
---- Let's play the game of Mastermind. ----
Reveal the codeword (y/n)? y
The codeword is 5545.
Round = 1, your guess (0 to stop): 1234
0 hits, 1 misses.
Round = 2, your guess (0 to stop): 1235
1 hits, 0 misses.
Round = 3, your guess (0 to stop): 1245
2 hits, 0 misses.
Round = 4, your guess (0 to stop): 5545
You've got it!
Another game (y/n)? y
---- Let's play the game of Mastermind. ----
Reveal the codeword (y/n)? n
Round = 1, your guess (0 to stop): 1111
1 hits, 0 misses.
Round = 2, your guess (0 to stop): 2222
1 hits, 0 misses.
Round = 3, your guess (0 to stop): 3333
0 hits, 0 misses.
Round = 4, your guess (0 to stop): 3111
0 hits, 1 misses.
Round = 5, your guess (0 to stop): 1322
1 hits, 1 misses.
Round = 6, your guess (0 to stop): 1244
2 hits, 0 misses.
Round = 7, your guess (0 to stop): 1255
You've got it!
Another game (y/n)? y
---- Let's play the game of Mastermind. ----
Reveal the codeword (y/n)? n
Round = 1, your guess (0 to stop): 789
An invalid guess.
Round = 1, your guess (0 to stop): 1111
1 hits, 0 misses.
Round = 2, your guess (0 to stop): 0
The codeword was 2514.
Another game (y/n)? n
```

Here is how Part 2 may run. Presented in blue are the places where the user's input changes the course of action. The user provides the feedback in two digits, where the first digit is the hits and the second the misses. In the second game, the user provides an invalid codeword.

```
---- Let's play the game of Mastermind. ----
Enter feedback yourself (y/n)? n
Round = 2, Size = 1296, Guess = 5332
hits=1, misses=1
Round = 3, Size = 230, Guess = 1522
hits=1, misses=1
Round = 4, Size = 37, Guess = 1435
hits=2, misses=1
Round = 5, Size = 4, Guess = 1355
hits=1, misses=1
Round = 6, Size = 1, Guess = 1234
Another game (y/n)? y
---- Let's play the game of Mastermind. ----
Enter your codeword: 4578
Invalid codeword.
Another game (y/n)? y
---- Let's play the game of Mastermind. ----
Enter your codeword: 2122
Enter feedback yourself (y/n)? y
Round = 2, Size = 1296, Guess = 6513
Enter feedback: 01
hits=0, misses=1
Round = 3, Size = 152, Guess = 4266
Enter feedback: 01
hits=0, misses=1
Round = 4, Size = 20, Guess = 2121
Enter feedback: 22
hits=2, misses=2
Round = 5, Size = 1, Guess = 1122
Enter feedback: 30
hits=3, misses=0
Error. No more candidates.
Another game (y/n)? n
```

3 Coding Hints

Codeword and guess representation You can use `int` for representing the codeword and the guess. You can extract the individual digits from both using the quotient and

remainder operations. Alternatively, you can use `String` for representing the codeword and the guess. You can extract the individual digits as `char` data using the `charAt()` method of `String`.

Computing the number of misses An alternate definition of the misses is as follows: Define for $d = 1, \ldots, 6$, let A_d be the number of times d appears in the codeword and B_d the number of times d appears in the guess (or vice versa). Let H be the number of hits. Then the number of misses is:

$$\min(A_1, B_1) + \cdots + \min(A_6, B_6) - H.$$

You can compute the sum using a for-loop with d as the iteration variable.

Finding the codeword For Part 2, you may use an array of candidate codewords, from which to select a guess. Initially, the array has all the `1,296` codewords as its candidates, and you generate all the codewords using a quadruple loop. After receiving the feedback, for each candidate, you can check if the candidate produces the same feedback on the present guess, and remove it if it does not. You can also remove the guess from the candidates.

Playing an indefinite number of games To allow an indefinite number of games (or a huge number of games), you can use a loop like the following, where `moreGames` has `true` as the initial value.

```
while ( moreGames ) {
  // play one game and ask if the user wants another game;
  // moreGames = ( the_answer == yes );
} // loop 1
do {
  // play one game and ask if the user wants another game;
  // moreGames = ( the_answer == yes );
} while ( moreGames ); // loop 2
for ( ; moreGames ;  ) {
  // play one game and ask if the user wants another game;
  // moreGames = ( the_answer == yes );
} // loop 3
```

Creating video recordings The submitted video recordings must demonstrate that your programs are indeed working. For recording, you can launch Zoom, start a personal meeting, share your Desktop screen (or your IntelliJ screen), and start recording with "Record on This Computer." When the recording is complete, Zoom generates a video file. The main folder for the recording can be found when you open Preference and click Recording. The video file (.mp4) appears in a folder named by the date of the recording.

4 Rubric for Evaluation

We will use the following rubric to evaluate your project.

Part	Point of Evaluation	%	Total
0	Submit multiple source Java files and a video	1.0	1.0
1	Submitting a demo video	1.0	10.0
	Submitting a well-commented source code	1.0	
	Playing an indefinite (or a very large) number of games	1.0	
	Playing each game until the guess is correct or the user enters 0	0.5	
	Revealing the codeword when receiving 0 as the guess	0.5	
	Computing and presenting the round number	1.0	
	Checking the validity of each guess	0.5	
	Generating a random codeword	0.5	
	Computing the feedback correctly	4.0	
2	Submitting a demo video	1.0	9.0
	Submitting a well-commented source code	1.0	
	Playing an indefinite (or a very large) number of games	0.5	
	Maintaining and updating a list of candidates, and successfully arriving at the codeword if the feedback is right	4.0	
	Allow the user to let the program to compute the feedback	0.5	
	Rejecting invalid codewords	1.0	
	Computing and presenting the round number along with the array size	1.0	