1. User Manual

Paragraph Form

The program is inspired by the Mastermind game, which involves two players namely the code maker and the code breaker. In this case, the computer would be the standard code maker, while the player would be the code breaker. The objective of the game is for the player to guess the correct digit pattern and their exact positions as generated by the computer. Like the mastermind game, the computer generates a digit pattern of n length. Notably, the digit pattern may contain multiple instances of the same digit. It then provides information to the player such as the pattern length and the total number of guesses. The code length is the number of digits in the digit pattern that the player will attempt to guess. The total number of guesses is the number of times that the player can enter their guess in the program. Our program uses a pattern length of 4 to 8 digits and a total of 10 guesses. Each pattern length is associated with a specific level of difficulty with 4 being the easiest and 8 being the hardest. When the player enters a guess, the computer responds with the guess board, a table entailing in 3 columns the number of tries, the player's guessed digit pattern, and the status of the guess (R -W) respectively.

If the player enters a guess that does not have the same length as the computer-generated pattern, the program considers this input invalid and thereby prompts the player to enter a guess of the correct length. When the player enters a valid guess, they receive points according to the following protocol: if one of the digits is right, but it is not in the correct digits place, the computer adds one point to W. Alternatively, if the digit is correct and it is in its correct place, the computer adds one point to R. For instance, 0R - 1W means that the player guessed one digit correctly but not its exact position in the pattern.

The game also has lifelines. The lifeline serves as a hint as it helps the player to figure out the correct pattern. Our program has two lifelines available as follows: Lifeline#1 and Lifeline#2. In elaboration, Lifeline#1 reveals a digit, which consequently decreases the player's available number of guesses by 1. Lifeline#2 then reveals the correct number in its correct location, which consequently decreases the number of guesses by 2. Lifelines can only be used once during the game, and they can't be used to diminish the number of guesses available.

The player will win the game when they manage to guess the correct digit pattern. The player will lose the game if they use all attempts without guessing the number generated by the computer. If the player exceeds the guess limit of 10 tries, the player loses the game and the hidden digit pattern is revealed. The computer then asks the player if they want to restart the game. If the player enters ‘yes,’ the program would repeat from the start. If the player enters ‘no,’ the program then stops.

Rules of the Game

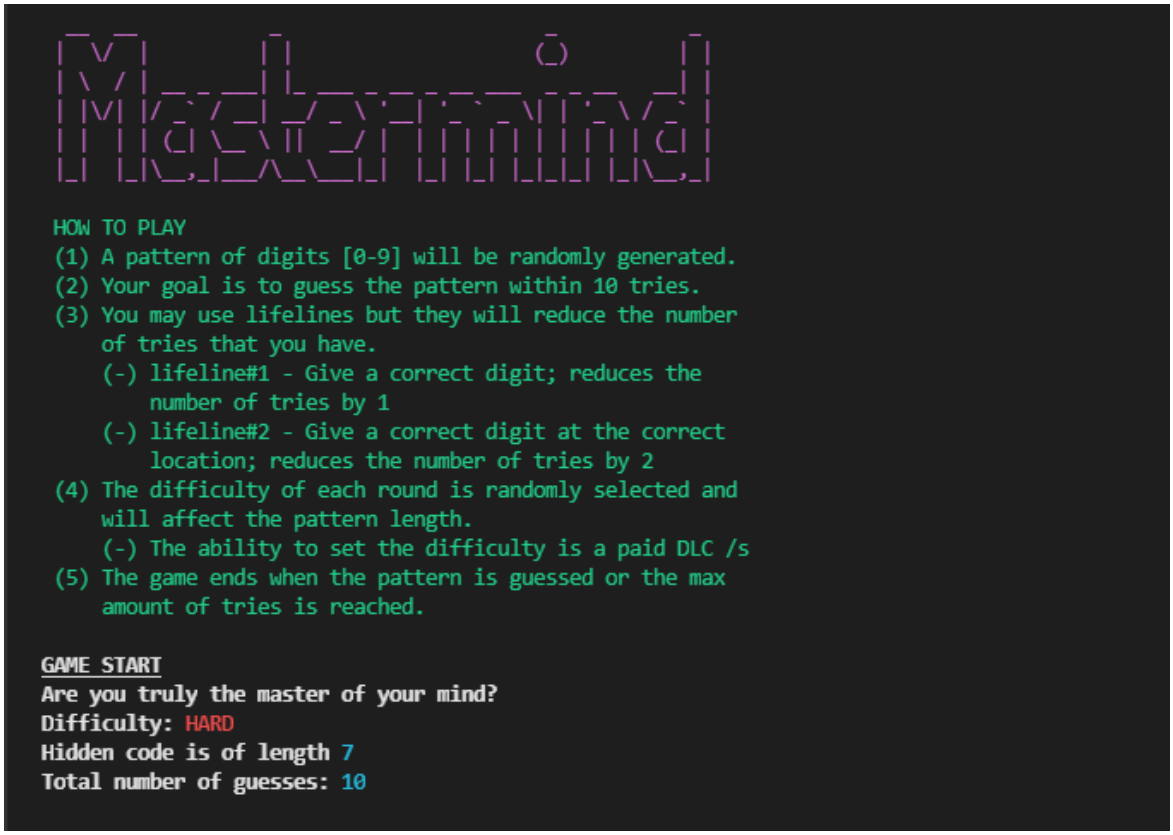
The program is a one player game; the user serves as the codebreaker. They play against the computer (CPU) which serves as the code maker. The rules are as follows:

- ●The CPU shall randomly generate a digit pattern with a length of 4 to 8 digits. The CPU shall display information about the digit pattern such as instructions on how to play the game, pattern length, number of guesses, and the difficulty level. Note that the pattern may contain multiple instances of the same digit (e.g. 10114).
- ●To win, the player must guess the pattern correctly within 10 tries. Otherwise, they lose.
- ●For every valid guess the player enters, the CPU shall return a response hinting the status of the player’s guess, which is evaluated as follows:
  - ○Red indicator (R) - Number of correct digits that are also in the correct digits place
  - ○White indicator - Number of correct digits that are not in the correct digits place
- ●The player may use a lifeline by entering one of the following: ‘lifeline#1’ or ‘lifeline#2.’ If the player avails a lifeline, the number of guesses are decreased by increments of 1 or 2. Lifelines may only be used once throughout the game. Below entails the details of the lifelines:
  - ○ Lifeline#1: A digit in the correct pattern is revealed. The number of guesses is reduced by 1.
  - ○Lifeline#2: A digit and its location in the correct pattern are revealed. The number of guesses is reduced by 2.
- ●The game ends when the player correctly identifies the pattern. The player is then declared as the winner and thus, they are crowned the Mastermind!

2. Implementation

Steps

1.At the start of the game, the program or the CPU will display instructions, level of difficulty, length of the hidden pattern, and the total number of guesses the player can use.



2.The player will then input the guess or the possible pattern on the input field as shown below:

3. Based on the input, the computer will then feedback the result of the guess hinting at the number of digits in their correct position and the digits that are in the pattern but not in the correct position with the indicators below:

a.Red indicator (R) - Number of correct digits that are also in the correct digits place

b.White indicator(*W) - Number of correct digits that are not in the correct digits place

```
Guess #1
Enter guess > 4567890
```

##	Guess	Status
1	4 5 6 7 8 9 0	2R - 1W
2	_ _ _ _ _ _ _	
3	_ _ _ _ _ _ _	
4	_ _ _ _ _ _ _	
5	_ _ _ _ _ _ _	
6	_ _ _ _ _ _ _	
7	_ _ _ _ _ _ _	
8	_ _ _ _ _ _ _	
9	_ _ _ _ _ _ _	
10	_ _ _ _ _ _ _	

In this example, **2R** indicates that there are two correct numbers in correct position and **1W** indicates that there is a single number in the guess that is also in the pattern given by the CPU

4.If the player inputs an invalid length of digits, it will print an error and will ask the player to input a guess again with the correct length.

```
Guess #1
Enter guess > 456789
Incorrect guess length! The pattern length is 7 (received: 6)
```

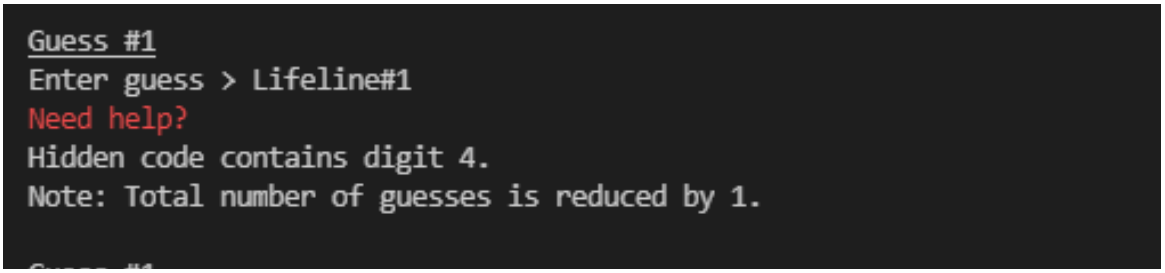
5. If the player inputs and invalid characters aside from numbers, it will print an error and will let the player input a guess again with a correct input containing only numbers.

```
Guess #8
Enter guess > yuio876
Invalid guess! The guess must consist only of digits or lifelines.
```

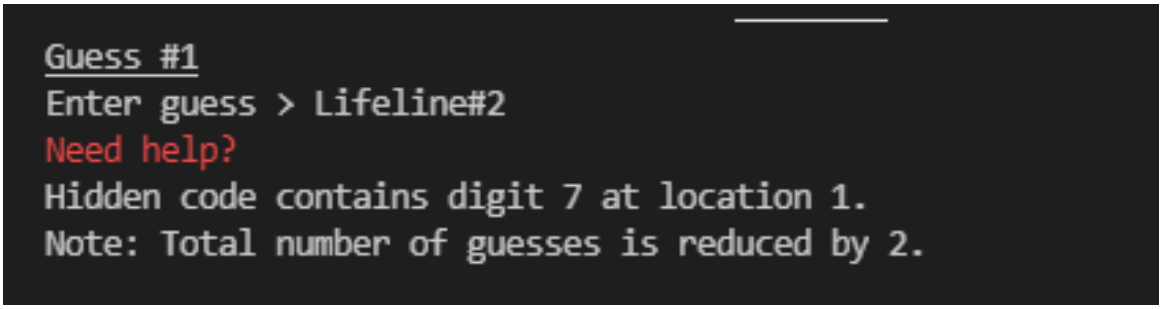
6.The player may also use lifeline to help break the pattern given by the CPU. These lifelines may ONLY BE USED ONCE during the whole duration of the game. If another lifeline is requested by the player, an error message will be displayed and will ask again for the player's input to

provide guesses only. Lifelines can be accessed by entering the following in the input field:

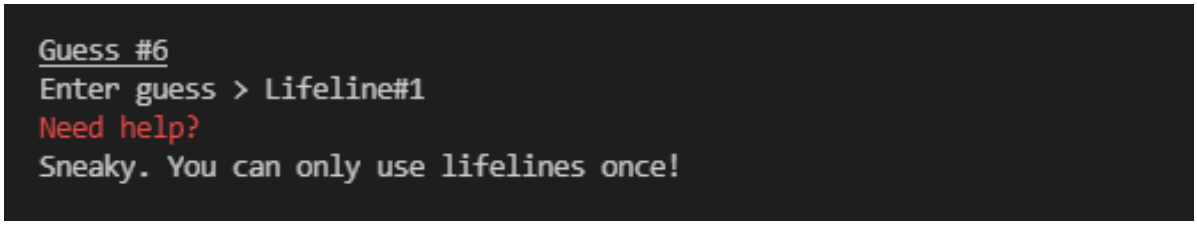
a.lifeline#1 : The player can ask the program to reveal a number in the generated code. As a consequence, the available number of guesses for the player will be decreased by ONE.



b.lifeline#2 : The player can ask the program to reveal a correct number and its position. As a consequence, the available number of guesses for the player will be decreased by TWO.



Given the following lifelines, the player is NOT allowed to use a lifeline when the lifeline will exhaust the number of available guesses. For example: The player is not allowed to use the lifeline#2 during guess #9 and #10 since it will already exhaust the number of guesses. An error message “Sneaky. You can only use lifelines once” will be displayed if the user requested for a lifeline more than once.



7. The game ends when the player successfully guesses the pattern before the number of guesses runs out and wins the game or if the number of guesses run out before the player successfully guesses the pattern. If the pattern is not successfully guessed, the CPU will display the correct pattern generated.

8. At the end of the game, the CPU will ask the player if the player wants to play again with the game. If Yes, it will generate another pattern and will then resume the game. If not, then the program ends and exits the game

Paragraph Form

The program uses the `random`, `typing`, and `colorama` libraries to work. It has a number of functions, namely `generate_pattern`, `compare_guess`, `draw_board`, and `start_game`.

The `generate_pattern` function generates a random pattern of digits of a specified length. It takes in `length`, an integer that indicates the number of digits of the hidden pattern and returns a string of digits. To generate the pattern, it creates the list `digits` and then loops `length` times, each loop adding a random digit to `digits` through the use of the `random` module's `randint` method. Afterwards, it returns `digits` as a single string.

The `compare_guess` function is used to compare two strings, the computer-generated digit pattern and the pattern entered by the player, after which information regarding the player's status is returned. It takes in two string arguments: `guess` and `pattern`. The parameter `guess` is the player's guess and the parameter `pattern` is the correct pattern. The function returns a tuple containing whether the player has won, whether the player has consumed a guess (that is, the guess is valid but incorrect), and the status message.

First, the function will check some guard clauses. If `guess` is a lifeline, it will return `False`, `False`, and `"Need help?"`. If `guess` isn't numerical, it will return `False`, `False`, and `"Invalid guess! …"` telling the player that the guess must only consist of digits or lifelines. If `guess` is of incorrect length, it will return `False`, `False`, and `"Incorrect guess length! …"` notifying the player of the problem. If it passes the guard clauses and is the correct answer, the function will return `True`, `True`, and `"YOU WON"`. However, if the guess is incorrect, it will compute the number of Rs and Ws. To get the number of reds, it will loop through

guess and pattern starting from the end and count how many of the digits match in specific indices. The matched digits will then be removed from the lists. To get the number of whites, it will loop through guess and count how many of the remaining digits are also in pattern. Afterwards, it will return False, True, and the number of Reds and Whites in the format of "*R - *W".

The draw_board function is used to provide a visual representation of the player's guesses so far by displaying a table with three columns including the number of guesses, the player's entered digit pattern, and the status of the entered digit pattern indicated by the "*R - *W" format. The function has three parameters, namely guess, max_guesses, and length. Guess is a list of tuples containing all guesses that the player has made and their respective statuses. Max_guesses refers to the allowable number of guesses, and length indicates the length of the pattern. The function does not return anything, but it prints out the table to the console.

The last function is start_game which takes in keyword arguments, namely pattern_length and max_guesses. The argument pattern_length is the length of the pattern of digits to be generated. By default, it generates randomly from 4 to 8. The argument max_guesses is the maximum number of allowable guesses the player has; it's 10 by default. This function is the main game cycle and it returns nothing.

The game starts by invoking the start_game function. Using pattern_length, it generates a pattern for the player to guess using generate_pattern. Other game variables, including the list guesses, guess_count, lifeline_count, and the flag is_playing is initialised. After that, the game cycle starts. The difficulty level, length of hidden code, and the total number of guesses are presented to the player, and then a loop using the flag is_playing starts.

When the player guesses, their guess is sent to compare_guess along with the correct pattern. The returned values from compare_guess are labeled as has_won, consumed_guess, and status.

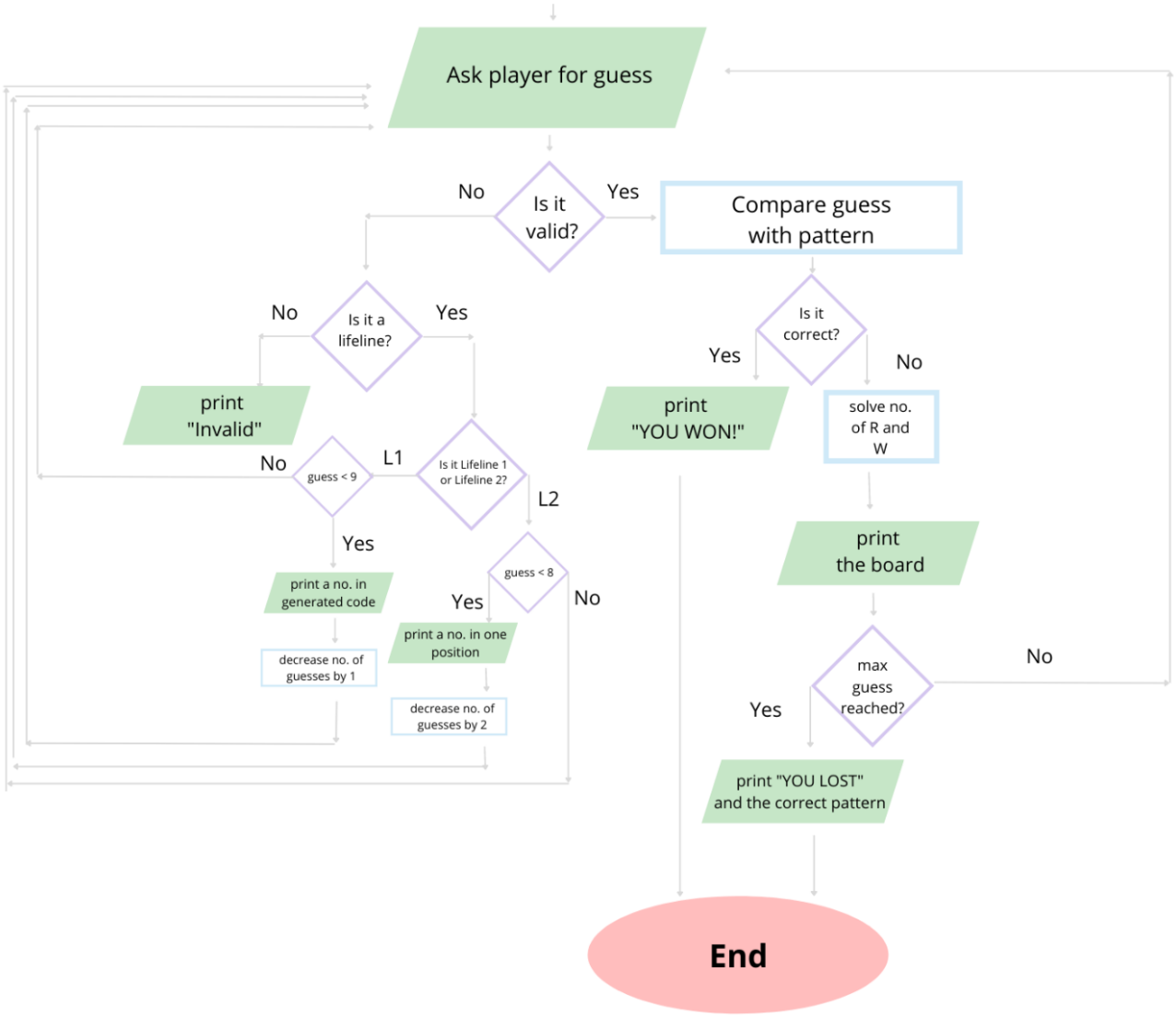
If consumed_guess is True, guess will then be added to guesses as a tuple along with the status message. Then the guess_count will be incremented and the board will be drawn. However, if consumed_guess is False, it will just print the status message to the player.

Next comes the win or lose logic. If `has_won` is `True`, it will print `"YOU WIN!!"` and set the `is_playing` flag to `False`. Otherwise, if the `guess_count` is greater than `max_guesses`, it indicates that the player has used all of their allowable guesses, prints `"YOU LOST!!"` along with the correct pattern, and sets the `is_playing` flag to `False`.

The `start_game` function also includes the logic for the lifeline which evaluates the player's input. If the player sent `"lifeline#1"` as input, `lifeline_count` which determines the number of times the player has asked for a lifeline, is incremented by `1`. If the player uses a lifeline and the `lifeline_count` is already `1`, the program tells the player that such an act is not possible. It also checks if the player is still allowed to use the lifeline or if the act of using it will make them exceed `max_guesses`. Under `lifeline#1`, a digit from the pattern to be guessed is revealed through the `choice` method from the `random` module and `max_guesses` is decremented by `1`. Under `lifeline#2`, a digit from the pattern will be received as well as it's index in the pattern. In this case, `max_guesses` will be decreased by `2`.

After the course of the game, the computer will ask the player if they want to play again with the game. If yes, it will generate another pattern and will then resume the game. If not, then the program ends and exits the game

Map/flowchart



3. Resources

Play Mastermind Online. (2022). Retrieved from https://webgamesonline.com/mastermind/rules.php

Play Mastermind Online. (2022). Retrieved from https://webgamesonline.com/mastermind/