Omar Alkendi

Mark Lehr, PhD.

19 April 2020

Midterm Project I: Mastermind

Introduction:

Mastermind is a code-breaking game in which the code is broken by the process of elimination. Once the player inputs a guess, the game engine parses the player's guess and informs the player if some digits are in the correct position, incorrect position, or no digits are part of the code. Thus, aids in the development of logical problem solving processes such division into cases process and the aforementioned elimination process. The goal of the game is to break the code in the least possible attempts.

Summary:

Project size: ~350 lines.

Number of variables: 8 variables (only 3 in the main)

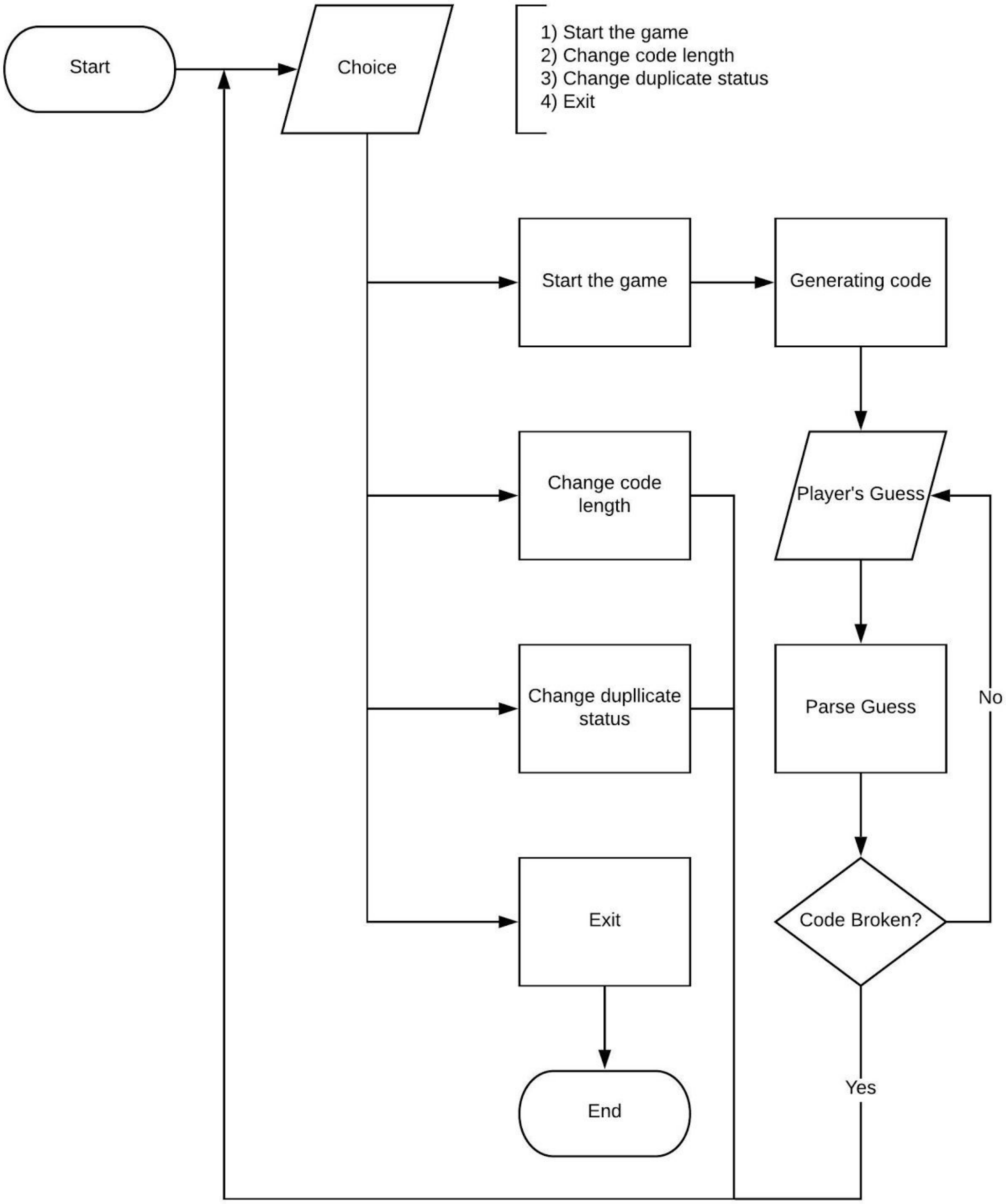
Number of functions: 8 functions (+1 algorithm)

Number of classes: 1 (MM)

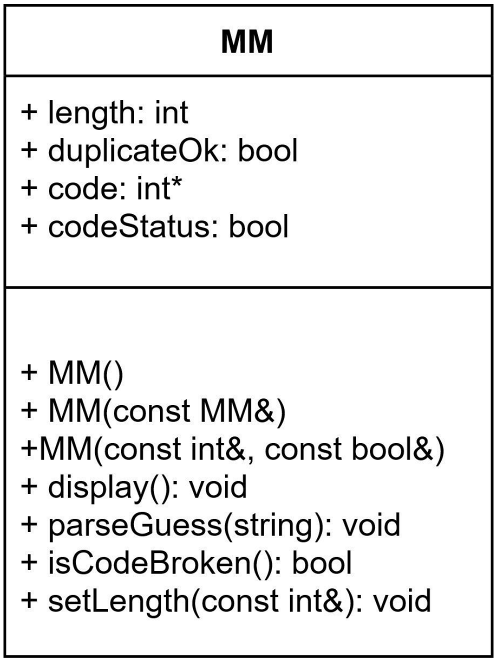
Number of member variables: 4 (All are private)

Number of member function: 5 (4 are public and 1 is private)

Overview of The Game:



The MM (Mastermind) Class:



MM

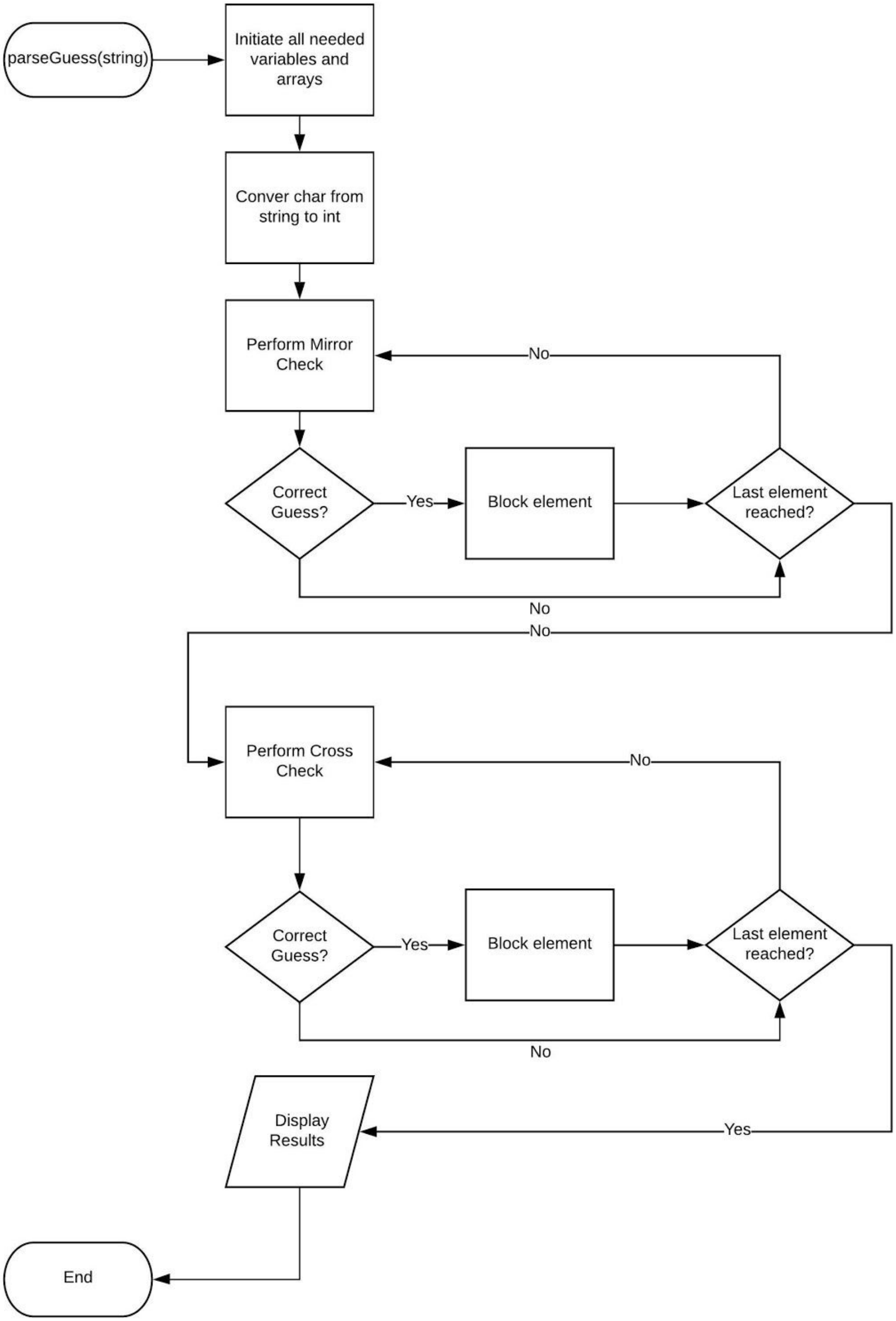
+ length: int
+ duplicateOk: bool
+ code: int*
+ codeStatus: bool

+ MM()
+ MM(const MM&)
+MM(const int&, const bool&)
+ display(): void
+ parseGuess(string): void
+ isCodeBroken(): bool
+ setLength(const int&): void

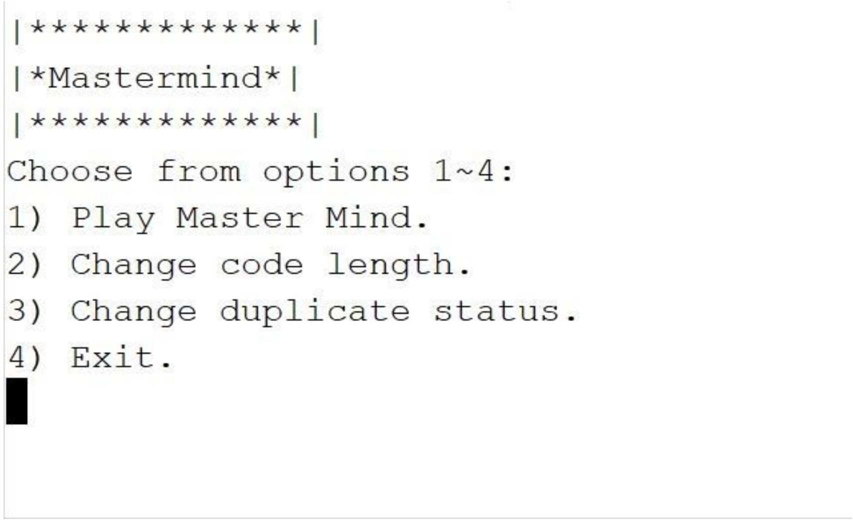
The MM class—short for Mastermind—is the core and the beating heart of the game. All the necessary information to generate a code is implemented in the class and readily available whenever an instance of the class is created. A secondary constructor is available to modify the game parameters. While all input validations are done outside the class scope, once the validations end, all the logical operations are done within the MM class—and mainly inside the **parseGuess(string)** function to be more accurate. The function **isCodeBroken()** is called after every successful attempt in the parse function to update the variable **codeStatus** once the user breaks the code. Since the number of attempts is an additional and non-essential information to the class, it has been defined outside of the class scope.

The parseGuess(string) Function:

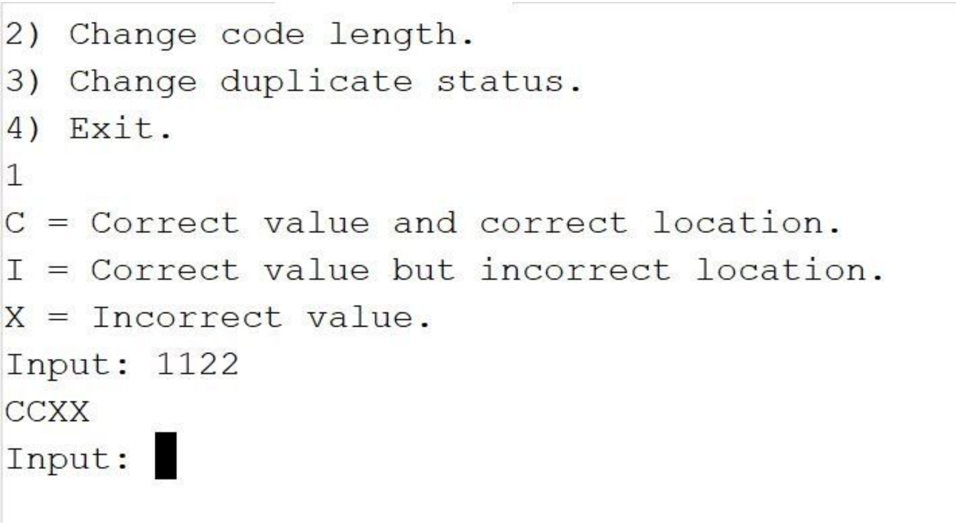
Since the data passed to the **parseGuess(string)** has already been validated, it is guaranteed that the function will analyze a string of integers that has a length equal to the code length. Hence, the function only processes string. It does so by first checking if there are any correct guesses in their correct location.—I called this check a mirror check. The function then checks if there are any correct guesses that are not in there correct location—I called this cross check. Once all checks are done, The function displays the results in the form CIXX—where C means correct location, I means incorrect location, and X means incorrect guess. The block element process in the chart is done through a boolean array.



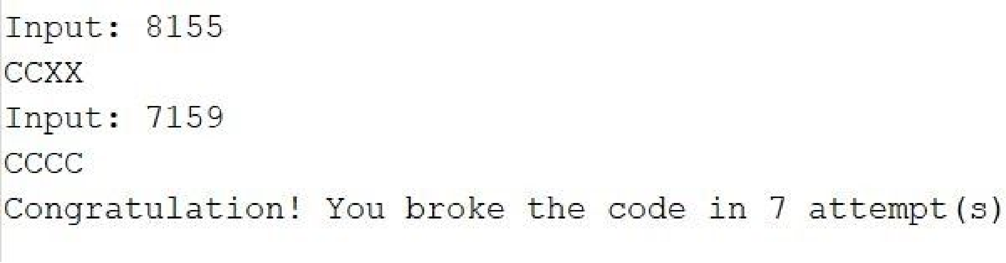
Images from Within:



```
|*************|
|*Mastermind*|
|*************|
Choose from options 1~4:
1) Play Master Mind.
2) Change code length.
3) Change duplicate status.
4) Exit.
```



```
2) Change code length.
3) Change duplicate status.
4) Exit.
1
C = Correct value and correct location.
I = Correct value but incorrect location.
X = Incorrect value.
Input: 1122
CCXX
Input:
```



```
Input: 8155
CCXX
Input: 7159
CCCC
Congratulation! You broke the code in 7 attempt(s)
```