

## General structure of the program:

### Mastermind

Get secret code length **and** range of numbers from the user

Create arrays for storing amount of specific **number** in secret code, history queue **and** set the **number** of guesses to 0

Generate random secret code

black pegs = white pegs = 0

Has user ran out of available guesses

T

F

break

incorrect guesses = 0

Display previous guesses

Get the guess combination from the user

black pegs = amount of correct guesses

incorrect guesses = amount of numbers that were **not** guessed

white pegs = length of secret code - incorrect guesses - black pegs

Display result of the guess

Add the guess to history queue

If black pegs == length of secret code

T

F

game won = true

Ø

black pegs = white pegs = 0

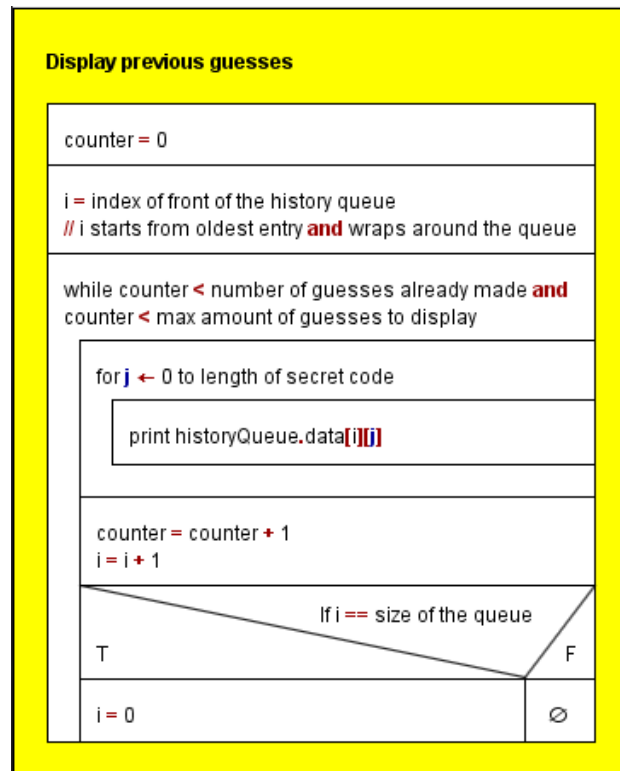
**number** of guesses = **number** of guesses + 1

While user has **not** guessed secret code correctly (game won == false)

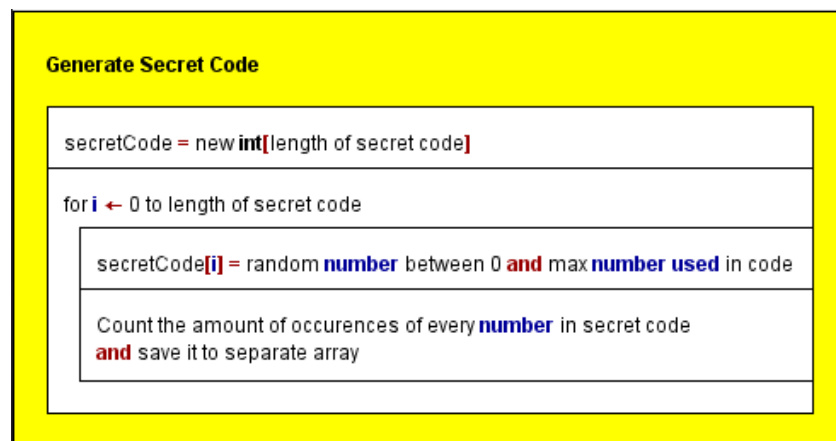
Display message telling the user that he either won **or** lost

While user chooses to play again

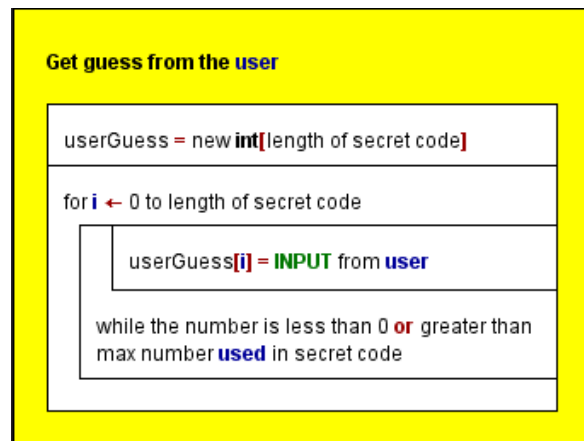
### DisplayPreviousGuesses method:



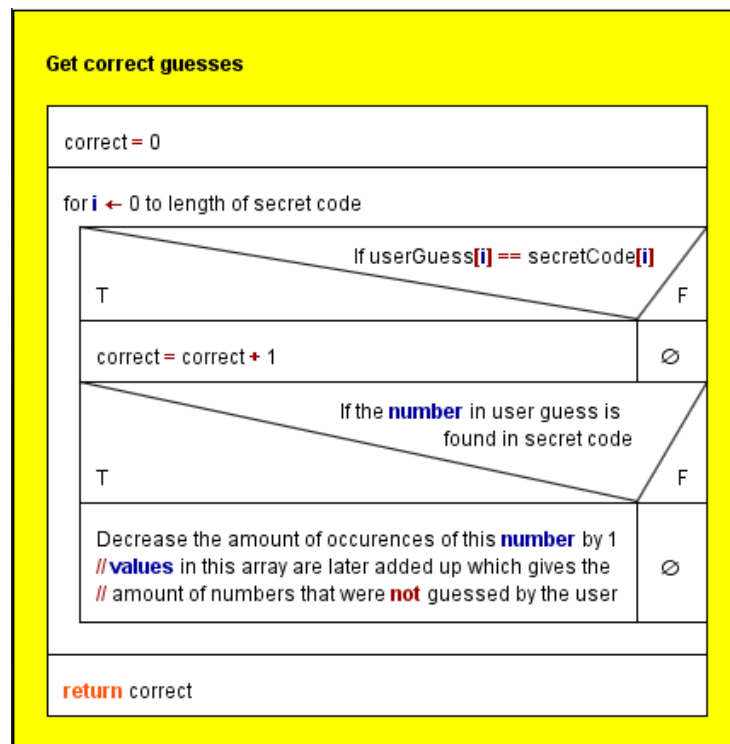
### GenerateSecretCode method:



### GetUserGuess method:



### GetCorrectGuesses method:



## GetConfiguration method:

### Get game configuration

length of secret code = user **INPUT**

While user **input** < 3 **or** user **input** > 6

max number **used** in code = user **INPUT**

While user **input** < 3 **or** user **input** > 9

amount of guesses available to user =  
 $((\text{max number } \text{used} \text{ in code} + 1) / 2 + \text{length of secret code} / 2) * 2$

length of array storing **guess** in history = length of secret code + 2  
// 2 extra elements for white **and** black pegs

## AddGuessToHistory method:

### Add user guess to history

temp = new **int**[(length of guess from history)]

for **i** ← 0 to length of secret code

temp[**i**] = userGuess[**i**]

temp[length of secret code] = black pegs  
temp[length of secret code + 1] = white pegs

If queue is full

T

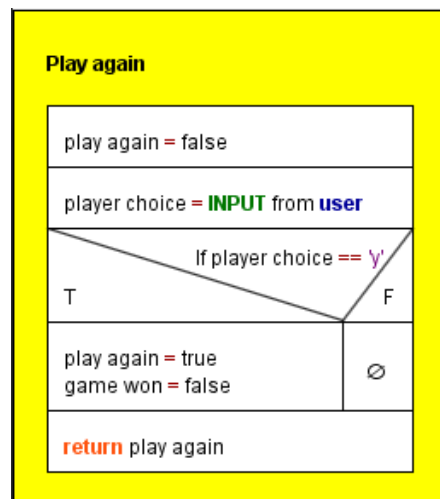
F

Remove oldest **entry** in queue

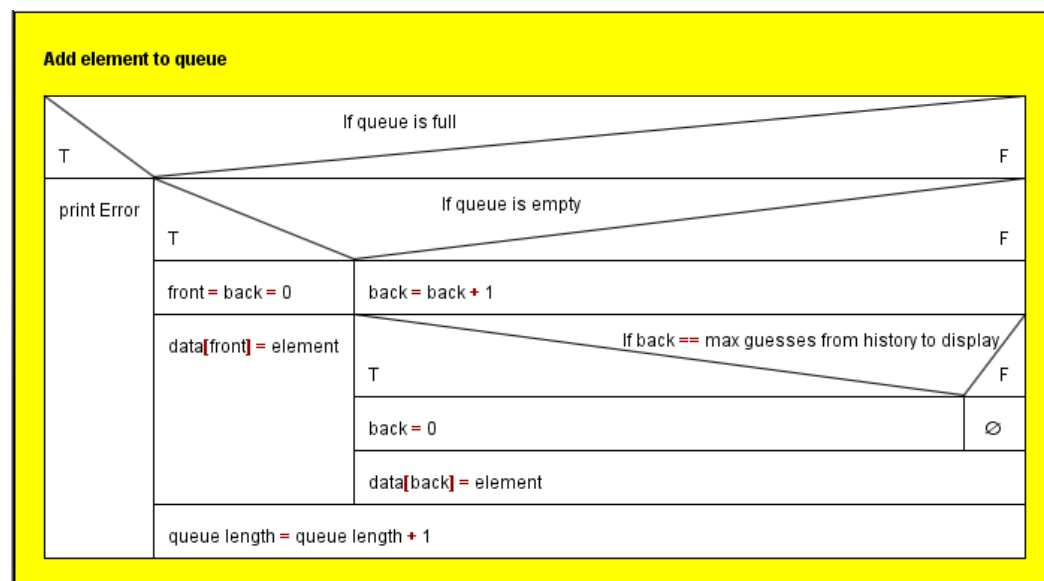
∅

Add temp to history queue

## PlayAgain method:



## Add method:



## Remove method:

### Remove element from queue

T	If queue is empty	
		F
print error	temp = data[front]	
	front = front + 1	
	If front == max guesses to display	
	T	F
	front = 0	∅
	queue length = queue length - 1	
	return temp	