MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG140 C Programming Assignment 3

**Date handed out**: 2 May 2023, Tuesday

**Date submission due:** 15 May 2023, Monday, 23:55, Cyprus Time

# Programming Assignment 3: MemoME

**Purpose:**
The main purpose of this programming assignment is to revise the topics that we have covered so far including arrays, pointers, functions, repetitive statements, conditional statements and fundamentals of C programming.

In this assignment, you are going to implement a simplified two-player memory game. The main purpose of the game is to have characters hidden behind doors represented on a square board and the players can open doors two at a time to find the matching ones. If they open two doors and they do not match, then they will lose points. Whoever finds the most matching pairs will win the game.

Do not try to compile your entire program in one "big bang". Compile it piece by piece. **Test each function/piece that you have compiled to make sure it works correctly before you add the next function/piece.**

**Game Specifications:**
MemoME is a two player game. The players first need to specify the board size which is nxn (square board). They can play with a board size (n) that can be between 2-32. Please note that the board size can only be multiple of two. Therefore, the board size can be 2x2, 4x4, 6x6, etc. When the game starts, the players will be asked to enter the board size. If they enter it properly, then the program will **dynamically** create the board.

```
Welcome to MemoMe!
Please specify the board size (2-32): 35
Board size can only be between 2-32!
Please specify the board size (2-32): 3
Board size can only be a multiple of two!
Please specify the board size (2-32): 6

A Board is created for you with the size of 6x6!
```

When the board size is decided then the game needs to add the characters to the board and close the doors. The board size will also decide the number of characters that are hidden. If the board size is 2, then 2 different characters will be hidden, if the board size is 8 then 8 different characters will be hidden.
The characters that can be hidden will be the characters starting from ASCII code of 33. Therefore, if the board size is 2, then the characters that can be hidden will be:

| ASCII code | Character |
|---|---|
| 33 | ! |
| 34 | “ |

If the board size is 4, then the characters that can be hidden will be :

| ASCII code | Character |
|---|---|
| 33 | ! |
| 34 | “ |
| 35 | # |
| 35 | $ |

These characters need to be also repeated. For example, if the board size is 2x2, then each character will be repeated twice. If the board size is 4x4, then each character will be repeated 4 times. These characters need to be located on the board **randomly**.

Once the players choose the board size, e.g., a board size of 6, a board can be initialised as follows. Please note that characters that will be hidden will be as follows: !, “, #,  $, %, &

|   | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | ! | “ | # | % | & | % |
| 2 | $ | ! | % | & | “ | $ |
| 3 | ! | “ | & | # | & | “ |
| 4 | # | % | “ | ! | & | # |
| 5 | $ | “ | % | $ | ! | $ |
| 6 | # | % | & | # | $ | ! |

Once the board is created the game can start. When the game starts the board will need to close the doors (hide the symbols) and start the game. When the doors are closed the players will be shown the board as follows:

|   | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | @ | @ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | @ | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | @ | @ | @ | @ | @ | @ |

The game is a two player game. Therefore, the players will have a turn. In each turn, the players will have a chance to enter the locations of two doors, if they find a matching characters, then they score 2 points and the doors will be opened. If they do not find a matching characters, then they lose 1 point and then the doors will be closed. A sample play will be as follows:

```
Player 1 Turn:
```

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | @ | @ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | @ | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | @ | @ | @ | @ | @ | @ |

```
Enter the coordinates of doors to open: (1,2) (6,1)
```

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | @ | “ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | @ | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | # | @ | @ | @ | @ | @ |

```
Unlucky! Your score: -1, total score: -1

Player 2 Turn:
```

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | @ | @ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | @ | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | @ | @ | @ | @ | @ | @ |

```
Enter the coordinates of doors to open: (1,1) (3,1)
```

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | ! | @ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | ! | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | @ | @ | @ | @ | @ | @ |

```
Lucky you! Your score: 2, total score: 2

Player 1 Turn:
```

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | ! | @ | @ | @ | @ | @ |
| 2 | @ | @ | @ | @ | @ | @ |
| 3 | ! | @ | @ | @ | @ | @ |
| 4 | @ | @ | @ | @ | @ | @ |

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 5 | @ | @ | @ | @ | @ | @ |
| 6 | @ | @ | @ | @ | @ | @ |

```
Enter the coordinates of doors to open:….
```

Your program needs to also consider three special cases:

1. If the user enters door numbers that are not valid then you need to make sure that you give the error message "`Sorry, wrong coordinates!`" and ask them for the coordinates again. For example, if the user enters (7,7) (6,1) then your program should give an error message for the board above.
2. If the user enters the same door coordinates then your program needs to give an error saying "`Sorry, two door coordinates are same!`" and ask the user to enter the coordinates again.
3. The users cannot open a door already opened. For example, for the board above, if the user enters: `(1,1) (3,1)` again, then your program will say the following and ask the user to enter again.
   ```
   (1,1) already opened!
   (3,1) already opened!
   ```

The game will finish when all the doors are opened. The user can start to play the game again at the end. Your program needs to ask:

```
Would you like to play again (Y/N)? N
Byee
```

If the user enters "Y" then the program will start again. If the user enters any other character, your program will stop.

**Programming Requirements:**

In order to implement this game, you will need to write at **least** the following functions, but if you need more functions, please add them.

| Function | Explanation |
|---|---|
| `main` | This function will call the `boardSize` function to get a proper board size from the user, then it needs to dynamically create the board with the given board size, and maintain the current total points of both players. This function needs to also maintain the rounds between the players and ensure that the players can continue to play the game until all doors are opened. This function also uses the `displayBoard` function to show the status of the board to the user after a user opens two doors. See the sample run! |
| `boardSize` | This function takes a value from the user and decides if it is a valid size. If it is not then it will continue to ask the user until the user enters a valid one. The board size can only be between 2-32 (including 2 and 32), and it has to be a multiple of 2. **Input**: No input **Output**: A valid integer number representing the board size. |
| `initializeBoard` | This function will get the board, its size (n) and randomly populate it. This function will generate a random character based on the given n value. |

| | |
|---|---|
| | Please see the game specifications section on how the characters will be generated.<br>**Input**: an array of board (nxn), and the n value.<br>**Output**: Returns true (1) if the operation is successfully completed. Returns false (0) otherwise. |
| `getCoordinates` | This function will ask the user to enter the coordinates in the format of "(row1 column1) (row2 column2)". This function will check if the coordinates entered are within the boundaries of the board, if the coordinates entered are different (users cannot enter the same two doors), and if the coordinates entered are not entered before (users cannot open a door already opened).<br>**Input**: two row and two column variables (row1, row2, column1, column2), and the board size.<br>**Output**: Returns true (1) if the operation is successfully completed. Returns false (0) if the coordinates are not appropriate. |
| `checkCharacters` | This function will take the coordinates of the hidden characters to be checked and then will return true (1) if they include the same characters.<br>**Input**: row1, row2, column1, column2, board, and the board size n<br>**Output**: Returns true (1) if they include the same character. Returns false (0) if they include different characters. |
| `displayBoard` | This function will clear the screen and display the current status of the board. |

**Sample Run:**

```
Welcome to MemoME!
Please specify the board size (2-32): 1
Board size can only be between 2-32!
Please specify the board size (2-32): 35
Board size can only be between 2-32!
Please specify the board size (2-32): 3
Board size can only be a multiple of two!
Please specify the board size (2-32): 4

A Board is created for you with the size of 4x4!

--Player 1 Turn:--
      1      2      3      4
  1   @      @      @      @
  2   @      @      @      @
  3   @      @      @      @
  4   @      @      @      @
Enter the coordinates of doors to open: (1,5) (1,1)
Sorry, wrong coordinates!
Enter the coordinates of doors to open: (1,1) (2,1)
```

```
       1      2      3      4
  1    !      @      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    @      @      @      @
Unlucky! Your score: -1, total score: -1

--Player 2 Turn:--
       1      2      3      4
  1    @      @      @      @
  2    @      @      @      @
  3    @      @      @      @
  4    @      @      @      @
Enter the coordinates of doors to open: (1,2) (1,2)
Sorry, two door coordinates are same!
Enter the coordinates of doors to open: (1,2) (2,1)
       1      2      3      4
  1    @      “      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    @      @      @      @
Lucky you! Your score: 2, total score: 2

--Player 1 Turn:--
       1      2      3      4
  1    @      “      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    @      @      @      @
Enter the coordinates of doors to open: (1,1) (4,1)
       1      2      3      4
  1    !      “      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    !      @      @      @
Lucky you! Your score: 2, total score: 1

--Player 2 Turn:--
       1      2      3      4
  1    !      “      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    !      @      @      @
Enter the coordinates of doors to open: (1,1) (2,2)
(1,1) already opened!
Enter the coordinates of doors to open: (2,3) (2,2)
```

```
       1      2      3      4
  1    !      “      @      @
  2    “      #      !      @
  3    @      @      @      @
  4    !      @      @      @
Unlucky! Your score: -1, total score: 1

--Player 1 Turn:--
       1      2      3      4
  1    !      “      @      @
  2    “      @      @      @
  3    @      @      @      @
  4    !      @      @      @
Enter the coordinates of doors to open: (1,1) (4,1)
(1,1) already opened!
(4,1) already opened!
Enter the coordinates of doors to open: (1,3) (2,2)
       1      2      3      4
  1    !      “      #      @
  2    “      #      @      @
  3    @      @      @      @
  4    !      @      @      @
Lucky you! Your score: 2, total score: 3
….
….

--Player 2 Turn:--
       1      2      3      4
  1    !      “      #      “
  2    “      #      !      #
  3    “      #      $      !
  4    !      $      @      @
Enter the coordinates of doors to open: (4,3) (4,4)
Lucky you! Your score: 2, total score: 8

***Winner is Player 2!***

Would you like to play again (Y/N)? N
Byee
```

**Grading Policy:**

**If your code does not compile, you will automatically get zero. If your code compiles, you will then be graded based the following scheme:**

| **Grading Point** | **Mark (100)** |
|---|---|
| `Main` | 25 |
| `boardSize` | 10 |
| `initializeBoard` | 20 |
| `getCoordinates` | 15 |
| `checkCharacters` | 15 |
| `displayBoard` | 10 |
| Code quality (Appropriate comments, variable names, formulation of selection statements and loops, reusability, extensibility etc.) | 10 |

Please make sure that you follow the restrictions for the assignment as follows:

- Strictly obey the input output format. Do not print extra things.
- You are not allowed to use global and boolean variables.
- Add your name/surname and ID at the top of your code as comments and name your source file “Name-Surname-StudentID.c”
- Only submit your C code.