**SFWRENG 2MP3 – Programming for Mechatronics**

**Fall 2018**

**Assignment 4: Arrays and Functions**

Note: Weekly course assignments account for 20% of the final course grade. **This assignment is due October 21st, 2018 at 11:59pm.**

**Objectives**

The purpose of this activity is to develop a greater understanding of arrays and functions within the C programming language.

**Tasks**

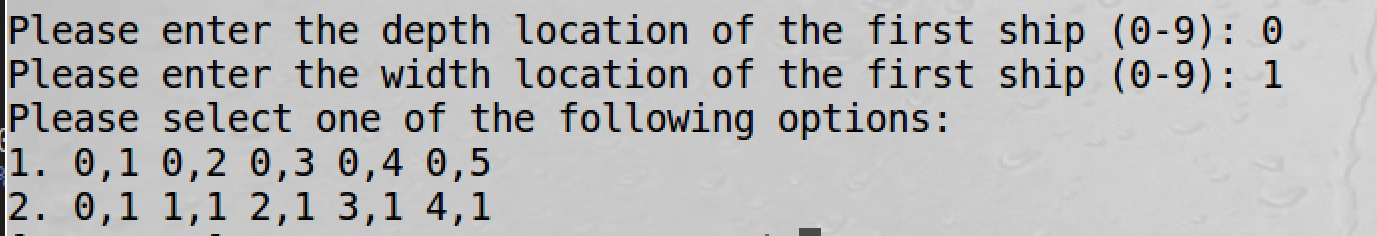
In order to complete this assignment, you must submit a written report of the C code and output for each of the following questions.

**Question 1a:** Create a program that enables users to play a game of battleship on 10x10 boards shown below:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | O | O | O | O | O | O | O | O | O | O |
| 1 | O | O | O | O | O | O | O | O | O | O |
| 2 | O | O | O | O | O | O | O | O | O | O |
| 3 | O | O | O | O | O | O | O | O | O | O |
| 4 | O | O | O | O | O | O | O | O | O | O |
| 5 | O | O | O | O | O | O | O | O | O | O |
| 6 | O | O | O | O | O | O | O | O | O | O |
| 7 | O | O | O | O | O | O | O | O | O | O |
| 8 | O | O | O | O | O | O | O | O | O | O |
| 9 | O | O | O | O | O | O | O | O | O | O |

In the main function, introduce the game to the players and display the above board. Each board is made up of a 2D array that is initially empty, the players will take turns placing their ships.

**Question 1b:** Create a function to place the first ship, a 1x5 piece within the 2D array, the function should ask the user to choose the first coordinate, both the width and height on the grid. The function should then create a list of options of the potential orientation for the ship, for example:



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | O | O | O | O | O | O | O | O | O | O |
| 1 | O | O | O | O | O | O | O | O | O | O |
| 2 | O | O | O | O | O | O | O | O | O | O |
| 3 | O | O | O | O | O | O | O | O | O | O |
| 4 | O | O | O | O | O | O | O | O | O | O |
| 5 | O | O | O | O | O | O | O | O | O | O |
| 6 | O | O | O | O | O | O | O | O | O | O |
| 7 | O | O | O | O | O | O | O | O | O | O |
| 8 | O | O | O | O | O | O | O | O | O | O |
| 9 | O | O | O | O | O | O | O | O | O | O |

Upon receiving the user’s response, fill the 2D array with a reserved character to indicate the location of a ship.

**Question 1c:** Create a function to place a second ship, a 1x3 piece within a 2D array, ensuring that the proposed location does not overlap with a piece that has already been placed. Again, the function should ask the user to choose the first coordinate, both the width and height on the grid. The function should then create a list of options of the potential orientation for the ship.

**Question 1d:** Create a function to place the third and final ship, a 1x4 piece within a 2D array, ensuring that the proposed location does not overlap with a piece that has already been placed. Again, the function should ask the user to choose the first coordinate, both the width and height on the grid. The function should then create a list of options of the potential orientation for the ship.

**Question 1e:** The first user has now placed all of their ships, the program should now be turned over to the second user and should clear the command prompt to ensure there is no cheating. The program should now execute the functions detailed in 1b, 1c and 1d to place the ships for the second user, clearing the screen again once they are finished executing.

**Question 1f:** The game will now commence, displaying the grid each time, and asking for depth and width coordinates for each guess. Within the displayed grid, a space that has been guessed should be marked, as well as the spaces that are a confirmed location for a ship, for example (X denotes a guess, H denotes a piece of the ship that was hit):

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | O | X | O | O | O | O | O | O | O | O |
| 1 | O | O | O | O | O | O | X | O | O | O |
| 2 | O | O | O | O | O | O | O | O | O | O |
| 3 | O | O | O | O | O | O | O | O | O | O |
| 4 | O | O | O | O | X | O | O | O | O | O |
| 5 | O | O | O | O | H | X | O | O | O | O |
| 6 | O | O | O | O | H | O | O | O | O | O |
| 7 | O | O | O | O | H | O | O | O | O | O |
| 8 | O | X | O | O | O | O | H | H | H | H |
| 9 | O | O | O | O | O | O | O | O | O | O |

Each time a guess has been made the user should be told if they hit a ship or not before the other player takes their turn. When a user has successfully sunk all 3 of their opponents’ ships (12 pieces total) they should be told that they have won the game.

**Additional Requirements:**

* The program should include necessary error handling (ex. A user should not be able to enter a coordinate that is not valid)
* Suitable prompts should be provided throughout for the understanding of the user

**Submission Requirements:**

* You have to submit the code in one single “.c” file.
* Also, you have to submit your report in the following format/template. Please create a PDF file.
  + Add one screen shot showing the error free compilation of your code.
  + Display a screenshot of each part of the code executing properly, including portions of a game being executed