Algorithms and Data Structures Coursework Report - TicTacToe Game

Gregor Kelly
40324671@live.napier.ac.uk
Edinburgh Napier University - Algorithms and Data Structures(SET08122)

1 Introduction

The assignment given was to make a working game of Tic-Tac-Toe, playable in the command line using the C programming language. It must be a playable game created by our understanding of algorithms and data structures that have either been self-taught or taught through classes and lectures.

2 Design

For my design I decided to create my game based around the layout of the number pad on a regular keyboard. This may not have been the best way to design a Tic-Tac-Toe game, however using the design of the number pad helped me visualize the game board. for example if a player wanted to place an 'X' in the top left corner of the game board, they would enter 7 when prompted. I decided that the best way to do that was to use a two-dimensional array.

2.1 Two-Dimensional Arrays

Using a two-dimensional array is much like using a matrix. Instead of using a single number to store a value like in a regular array, 2D arrays use coordinates to store its values. For example, a regular array would store values using single values seen in figure 2 and a 2D array would store values using rows and columns to define the location seen in figure 1.

[0][0]	[0][1]	[0][2]
[1][0]	[1][1]	[1][2]
[2][0]	[2][1]	[2][2]

Figure 1: 2D array that contains 9 values.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Figure 2: 1D Array that contains 9 values.

In memory, using a two-dimensional array stores its values in the same way as a regular array would, in a linear fashion, with the first row stored first, then second row etc. This is quite important to remember when using pointers as forgetting the order of rows first columns second may point to a different address and that will cause complications in coding.In my program I used two 2D arrays, one to create the board and a second to initiate the win conditions.

2.2 Pointers

Getting the base game working I had originally placed the input, if validated correctly, straight into the game board but when it came down to deciding on who won, I had no way of recording any of the win conditions. That is when I learned about pointers and how they can be used to reference into different functions and be used in more than one way.
In my game I used pointers to point to a location in the game board 2D array and the win conditions array. For example, when location 1 was filled with a players piece, it would go through the win condition loop and see if there was a winner by counting the number of player pieces in a line, if there were three in a row then the checkWin function would return true(1) and the player would win.

3 Enhancements

If I were to continue working on this project I would've liked to create a score feature that would keep count of each game won and which player was winning overall.
I would also have liked to fix the bug in which my game exits when a letter is entered when prompted.
feature I was hoping to add would be a history of play to show the earlier games played.

4 Critical Evaluation

Overall, my game works however I feel there are parts that I should have spent more time on. For example, the validation of the players input could be vastly improved. When prompted with an input for a turn, if you type in a letter, the game will show the error message prompting to re-enter as it was an incorrect input, but instead of it letting you re-enter the game exits and I couldn't figure out how to fix it.
Another part of the validation process is when multiple numbers are entered, the game plays all the numbers in sequence playing multiple turns. For example it is player X's turn, if 159 is entered it will place an X in location 1, and O in location 5, and an X in location 9. If I had given more time to debugging these parts I would probably have managed to fix them in time for the hand in.

5 Personal Evaluation

Working on this project has really helped in my understanding of using pointers, two dimensional arrays and furthering

my knowledge of C, I am really used to using object oriented programming languages such as C++ or C Sharp so using a non object oriented design process proved quite the challenge for me.
Another big challenge for me in the coursework was understanding how to use pointers instead of placing values straight into an array, thankfully the in class support staff/students were great at explaining it from the ground upwards.
Overall I don't think I have done as well as I could have with this project as I have had plenty time to work on it but did not take advantage of that time. It is not my proudest bit of work as I really struggled to find the motivation to work on this and other projects over the last month. I feel like if I was in a better frame of mind over the past couple of months I may have been able to achieve a higher mark but I cannot turn back time.