Matthew Matar UCP ASSIGNMENT

CONTENTS PAGE

UCP ASSIGNMENT REPORT

Section 1

File: main.c

This is a very simple and short file. Its main job is to analyse the arguments when the program is started. It does this by first checking if there are 2 and only 2 args (Executable + settings file). It then checks if there is a .txt extension because this is all that the assignment says to be able to handle, therefore I should not need to handle or accept anything else other than that. The program will exit if any of these happen and will let the user know if it is that arguments that are the problem with them starting the program.

File: inputFiles.c

This is either the file I spent the most or the second most time on. The way I approached this was to try and handle every possible problem I could think of. This includes limiting the data to fit the terminal screen right, not scaling, I just printed widths and heights until I got a good max of 25. I also disallowed the use of a K value that was not between M or N (inclusive) as this does not work well in the game, meaning that if played in a way a player will always win no matter what player 2 does. It also checks all 5 points in the assignment to be able to handle invalid settings files. This file scans the settings file for just 6 chars which would be the minimum, SOH-(M/N/K)-=-00-\n. Any lines that are longer will be invalid, making the whole settings file invalid.

File: controller.c

Basically, this file's job is to control what happens each time a menu option is selected and to follow the route back. It loops through the menu until the exit button is selected, hence stopping the program, so the name controller is appropriate for this one. It also makes sure all the allocated memory is freed before the program terminates, disallowing memory leaks. One of the notable things that this file and the menu file share is how they deal with the different compilations (either normal, SECRET, EDITOR or EDITOR+SECRET) very elegantly (to the user, not to me). The edit settings option is the most confusing and annoying option in this all.

File: menus.c

The menu file is a lot shorter than I imagined when I planned this out, I came to realise that there are not as many menu options as I thought there would be. However, despite this I feel like it does its task well. I used Strings as menu codes, but on further thought I should have used Integers since they are faster to check. I should have used a macro definition for this, but I only think of that now since with programming I am not used to doing things like this and it does not come to mind. Much of this file validates user input by using fgets then atoi to convert it into an Integer. Atoi is not good if you need the user to enter in the number 0 however all the menu stuff starts at 1 so I'm good there.

File: grid.c

Grid takes the cake with being the most annoying of all the files for me, it was good until I had to try and figure out a way to determine a winner. And thinking of it now, I really think I missed a case that I have in my head, but oh well. It basically is responsible for playing the game (menu option 1). It loops through asking each player for their turn until one of them win or they fill up the board completely, since I couldn't figure out how to determine if there are no more empty tiles for someone to win. It also selects the first player at random, so that's cool. My method for finding a winner just basically came around by thinking how each row/col/diagonal needs to have K amount of matches so when the matchnum reaches this then someone wins, and every time the opposite or an empty tile shows up this gets reset back to 1 since, one tile = 1 match. The checkWinner method caused me much grief and I almost felt like giving up, but I think I have programmed a solution that fits most cases to find a winner.

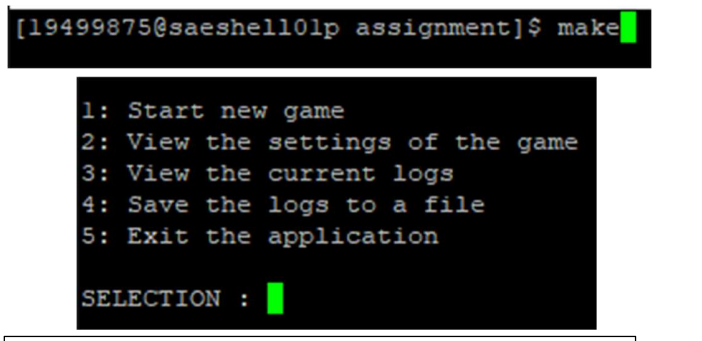
File: linkedList.c

This is my linked list and it's not that much special, other that it's a doubly ended doubly linked list. However, I just copied over my doubly ended doubly linked, linked list from DSA to this so it's not like it took me very long. The way it saves files I thought was cool, using the sprint to format all the data bits from the date/time and settings, and I remembered to pad zero's so I'm happy about that. The saving is the same as the print, but just with fprintf instead of printf. Other than that, there's not much more to say about this file.

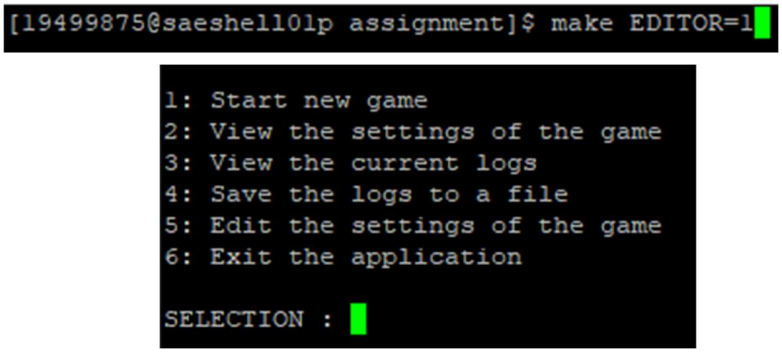
Section 2

All my logs are created within the grid.c file, within the startGame method. This was straight forward for me to thing about, since you get all the data you need, right there and then. I just place these logs right into the list straight after I create them. I shove them right at the end where they need to be, since that is very logical. My design of my linked list helps a lot and I don't really have to think about it, the logs just go where I tell it. When I did DSA, linked lists were something I got a good grasp in, so the whole storing of data in a linked list didn't really cause much complication for me. As for the printing of the logs, I just logically thought about printing the head, and then looping through the rest of the list until it was at the last node. The print and save methods for my Setting and my Log structs were enjoyable to make and actually didn't require as much mental capacity as a lot of the other things in this assignment. I am glad I managed to pretty much print out the logs exactly as it is on the assignment sheet.

Section 3



Normal compilation



Editor compilation

```
[19499875@saeshell01p assignment]$ make SECRET=1
```

```
1: Start new game
2: View the settings of the game
3: View the current logs
4: Exit the application

SELECTION :
```

Secret compilation

```
[19499875@saeshell01p assignment]$ make SECRET=1 EDITOR=1
```

```
1: Start new game
2: View the settings of the game
3: View the current logs
4: Edit the settings of the game
5: Exit the application

SELECTION :
```

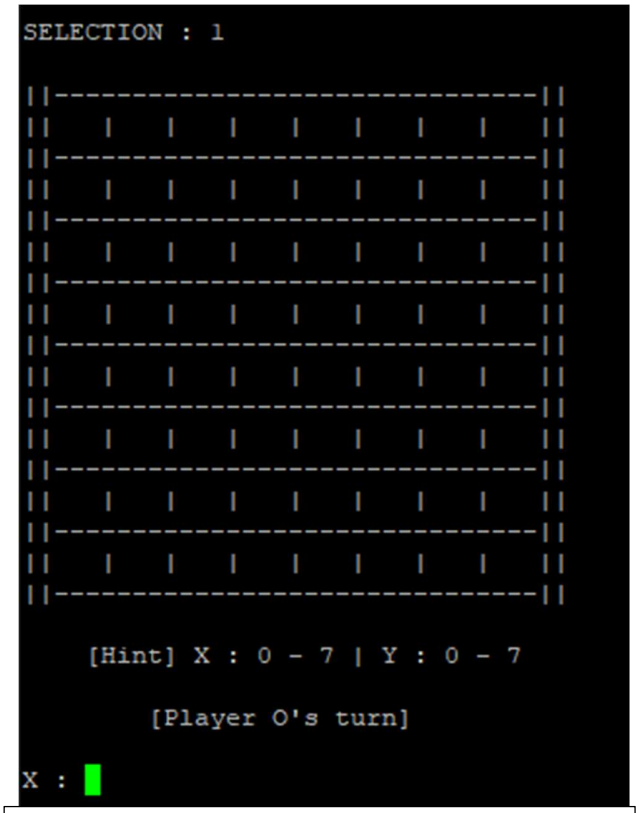
Secret and editor compilation

```
[19499875@saeshell01p assignment]$ ./TicTacToe settings.txt
```

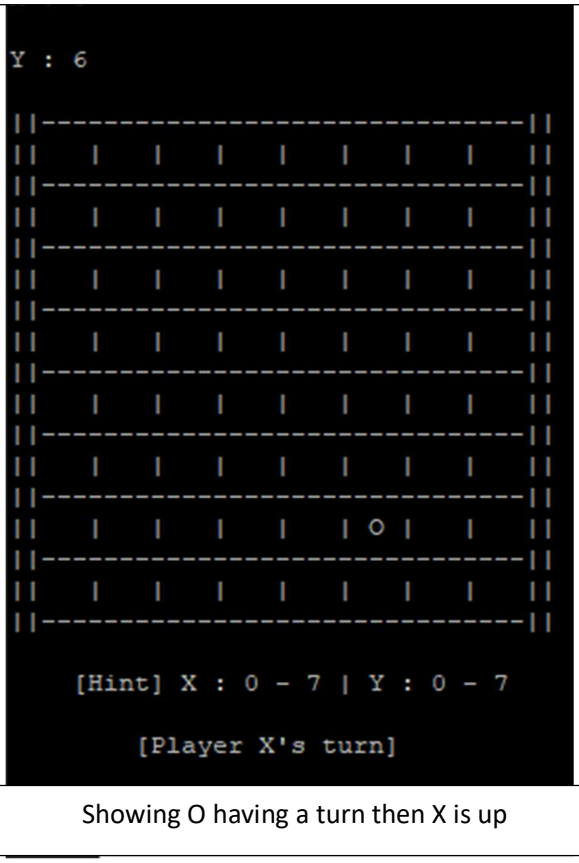
```
1 M=8
2 K=8
3 N=8
```

Settings shown in file and settings shown in game

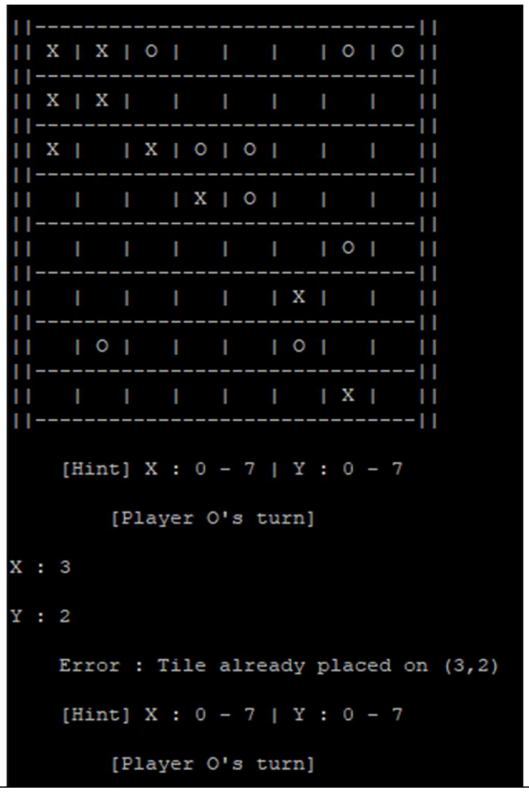
```
SELECTION : 2

    Height          : 8
    Width           : 8
    Matching tiles : 8


1: Start new game
2: View the settings of the game
3: View the current logs
4: Save the logs to a file
5: Exit the application

SELECTION :
```



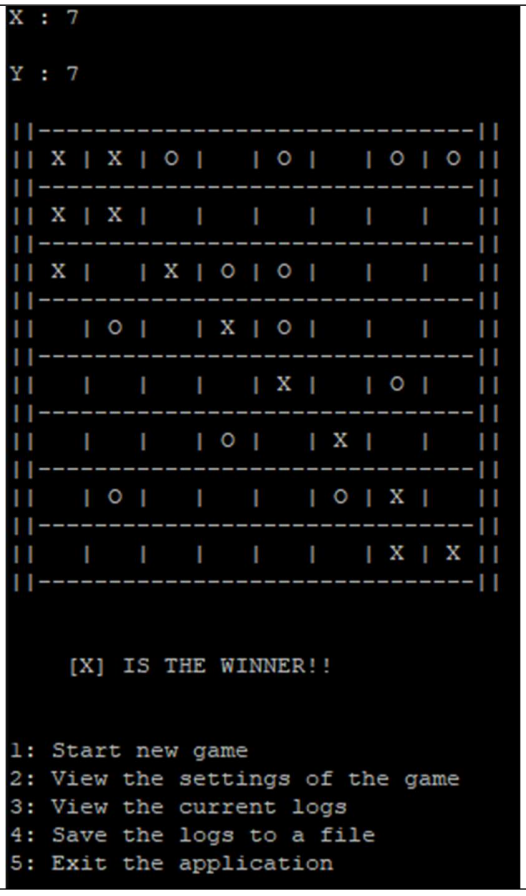
Showing initial map with random player up first



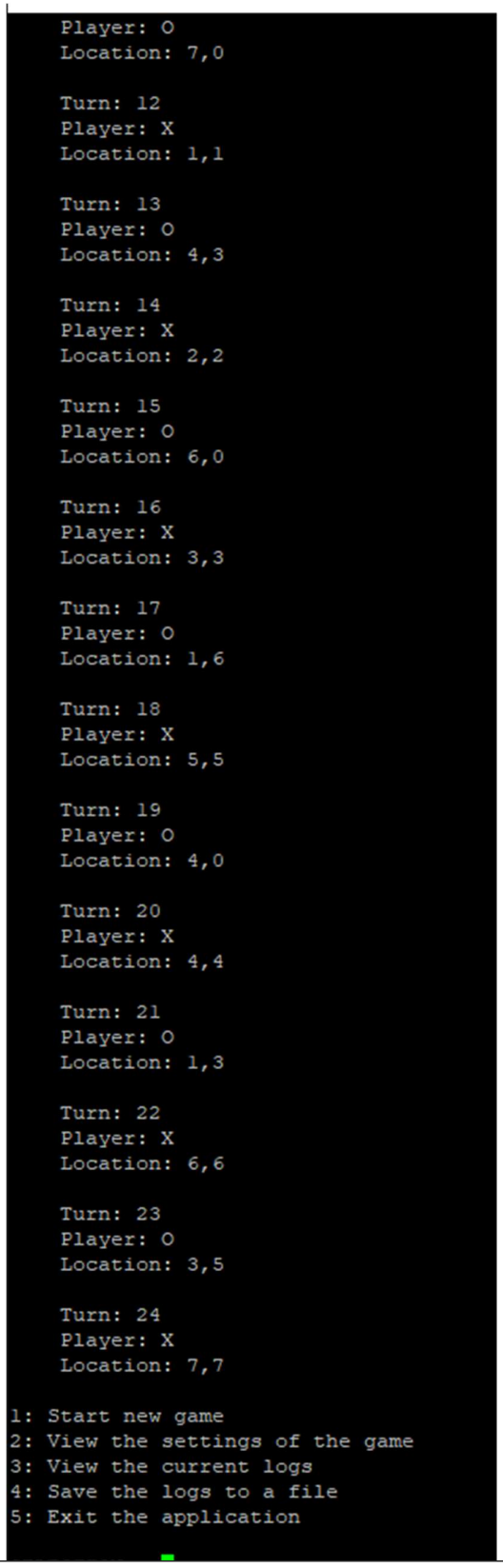
Showing O having a turn then X is up



Showing trying to place on a non-empty tile



Showing X winning (Diagonally)



Logs as seen in terminal without scrolling up

Logs seen in file that it saved to

```
 1
 2 SETTINGS:
 3     M: 8
 4     N: 8
 5     K: 8
 6
 7 GAME 1:
 8     Turn: 1
 9     Player: O
10     Location: 5,6
11
12     Turn: 2
13     Player: X
14     Location: 0,0
15
16     Turn: 3
17     Player: O
18     Location: 4,2
19
20     Turn: 4
21     Player: X
22     Location: 6,7
23
24     Turn: 5
25     Player: O
26     Location: 3,2
27
28     Turn: 6
29     Player: X
30     Location: 1,0
31
32     Turn: 7
33     Player: O
34     Location: 2,0
35
36     Turn: 8
37     Player: X
38     Location: 0,1
39
40     Turn: 9
41     Player: O
42     Location: 6,4
43
44     Turn: 10
45     Player: X
46     Location: 0,2
47
48     Turn: 11
49     Player: O
50     Location: 7,0
51
52     Turn: 12
53     Player: X
54     Location: 1,1
55
56     Turn: 13
57     Player: O
58     Location: 4,3
59
60     Turn: 14
61     Player: X
```

```
61     Player: X
62     Location: 2,2
63
64     Turn: 15
65     Player: O
66     Location: 6,0
67
68     Turn: 16
69     Player: X
70     Location: 3,3
71
72     Turn: 17
73     Player: O
74     Location: 1,6
75
76     Turn: 18
77     Player: X
78     Location: 5,5
79
80     Turn: 19
81     Player: O
82     Location: 4,0
83
84     Turn: 20
85     Player: X
86     Location: 4,4
87
88     Turn: 21
89     Player: O
90     Location: 1,3
91
92     Turn: 22
93     Player: X
94     Location: 6,6
95
96     Turn: 23
97     Player: O
98     Location: 3,5
99
100    Turn: 24
101    Player: X
102    Location: 7,7
~
```