COMPUTER ARCHITECTURE ASSIGNMENT REPORT

FOUR IN A ROW.

Nguyễn Anh Khang - 2053097

1. **Data structure.**

Using a one-dimensional array to store 42 data of the gameboard. Each square equals one element in the array and every set of 7 consecutive elements in the array are considered as a row in the gameboard, totally there are 6 sets ,6 rows, and 7 elements are considered as 7 columns.

| 0\_1st element | 1\_2nd element | 2… | 3… | 4… | 5… | 6… |
| --- | --- | --- | --- | --- | --- | --- |
| 7… | 8… | 9… | 10… | 11… | 12… | 13… |
| 14… | 15… | 16… | 17… | 18… | 19… | 20… |
| 21… | 22… | 23… | 24… | 25… | 26… | 27… |
| 28… | 29… | 30… | 31… | 32… | 33… | 34… |
| 35… | 36… | 37… | 38… | 39… | 40… | 41… |

1. **Algorithm.**
   1. **Switching player every turn.**

There are only two symbols ‘x’ (120 decimal in ascii code) and ‘o’ (111 decimal in ascii code). So when I want to switch ‘x’ to ‘y’ I sub 231 to 120 to get ‘y’ and vice versa.

* 1. **Printing game board to screen.**

As mentioned above, every set of 7 consecutive elements is a row. After printing those 7 elements, I will jump to the next line and print the next 7 elements. Repetitively until the last element is printed.

* 1. **Checking winning conditions.**

1. *Vertical checking process:*

I check column by column and stop when the last column is already checked. Firstly, I start at the intersection of the bottom row and the leftmost column. If the symbol stored in that position matches the symbol of the player who just finished their turn, a counting variable will increase to 1 (initialized 0) and then by subtracting the current position to 7 to move the upper slot and check. If the counting variable equals 4 then that player wins, otherwise, if the symbol does not match that of the player, the counting variable will be reset to 0 and the process continues until reaching the top of the game board. In addition, the process will move to the next column if any of the symbols which belong to row 3,2,1,0 does not match the player’s symbol because there will be less than 4 positions which means the winning condition will not happen.

1. *Horizontal checking process:*

I check row by row and stop when the top row is already checked. Firstly, I start at the intersection of the bottom row and the leftmost column. If the symbol stored in that position matches the symbol of the player who just finished their turn, a counting variable will increase to 1 (initialized 0) and then by adding the current position with 1 to move to the next slot and check the symbols of the next positions. If the counting variable equals 4 then that player wins, otherwise, if the symbol does not match that of the player, the counting variable will be reset to 0 and the process continues until reaching the last column of the game board.In addition, the process will move to next column if any of symbol which belong to column 3,4,5,6 does not match the player’s symbol because there will be less than 4 position which means the winning condition will not happen.

1. *Diagonal checking process:*

It is similar to others checking processes, but one of them moves vertically slot by slot to the top and the other one moves horizontally slot by slot to the right. This stage has two moving patterns which are called forward slash and backslash. The former start at the same position with the other two processes while the latter start from the top and the leftmost column of the game board. Backslash pattern moves to next slot to check by adding the current position with 8 (a cycle to go to the same column but in a different row is 7, so add 1 more to move diagonally).However forward slash pattern moves to next slot to check by subtracting the current position to 6.

1. *Tie condition:*

A match is only a tie when all slots are chosen but neither player wins.

III. Summary process.



Some more processes such as: Checking number of times that player is allowed to choose again, checking whether players choose illegal positions are implemented by using counting variables and apply the if-else method during the game.