

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



Lab REPORT
Computer Architecture CO2008

Assignment

TIC TAC TOE

Instructor: Băng Ngọc Bảo Tâm

Student: Nguyễn Hoàng Thuận – 2052729 – CC03

HO CHI MINH CITY, APRIL 2022

Contents

1. Introduction	2
2. Main idea and explanation	2
2.1. Describe the main idea by flowchart	2
2.2. Assign 'X' or 'Y' on the board	4
2.3. Undo the previous step	5
2.4. Check central point in each player's 1st turn	6
2.5. Check winning conditions	7
2.6. Check full board	9
2.7. The final result	10
3. Conclusion	11

1 Introduction

Tic-Tac-Toe is a game in which two players take turns in drawing either an 'O' or an 'X' in one square of a board consisting of twenty-five squares. The winner is the first player to get three of the same symbols in a row. In addition, the game has a few requirements:

1. During the first turn of both players, they are not allowed to choose the central point (row 3 & column 3).
2. Players can undo 1 move before the opponent plays.

The assignment requires us to design and write MIPS assembly language for implementing a text-based **Tic-Tac-Toe** game for two players.

2 Main idea and explanation

2.1 Describe the main idea by flowchart

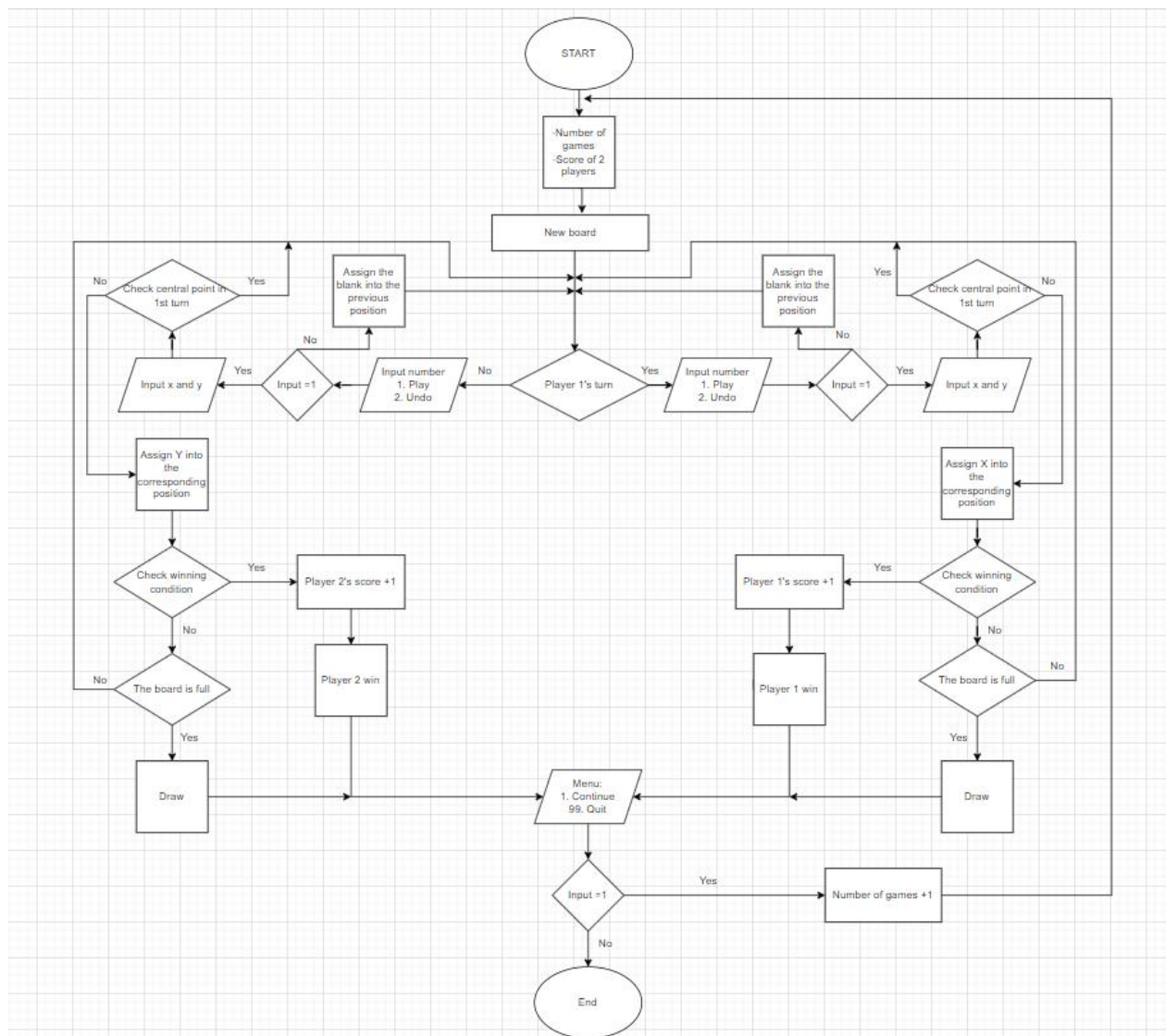
Explain the idea :

- First of all, the program will START. Then, it will display the Number of games that players have played and the Score of each player.

After that, it will Create new board for a new game. The program will check turn of player if it is Player 1 or not and Assign 'X' or 'Y' suitable for each player. Then the program will ask players if they want to Play or Undo, they can undo the previous step or continue to place marker by inputting x and y value corresponding with position in the board. (they can not place the central point in their first turn)

- The next step, the game will Check winning condition of the recent board if it true the player who has just made a move will win and the score of that player plus one. The program will print the result to the screen and go to the step Display the Menu. In the case, two player haven't won but board is full (All the board are 'X' and 'Y'), it display "Game Draw". The board is not full, the program continues to require input from players.

- In the step Display the Menu. The program will waiting for the Number from user. If it is 99, the program will END. In another case (the input number different from 99), It will auto understand that the user want to play another game and Number of games increase 1. And the program will show the recent result and start a new game. The process is described in the Flowchart below.



As we can see, some steps are easy to implement but some of them are not.

- The first one is how to assign 'X' or 'Y' from x and y value.
- The second one is how to undo the previous step.
- The third one is how to check central point in each player's 1st turn.
- The fourth one is how to check winning condition in the board.
- The final is check full board (the board is full of 'X' and 'Y').

We will go to the detail of each procedure below.

2.2 Assign 'X' or 'Y' on the board

First of all, I declare an array with 25 characters with the value ' '. The address distance of them are 4 bit, so I have an array with address respectively 0, 4, 8, 12,..., 96.

Therefore, the address of each character on the board should be:

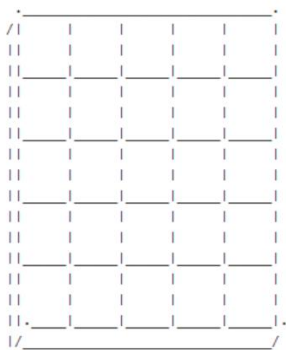
0	4	8	12	16
20	24	28	32	36
40	44	48	52	56
60	64	68	72	76
80	84	88	92	96

Now, when we input the x and y position value, we have to transform them into the proper address. To do it, we have to use the formula:

$$\text{Address} = (x - 1) * 4 * \text{board size} + (y - 1) * 4$$

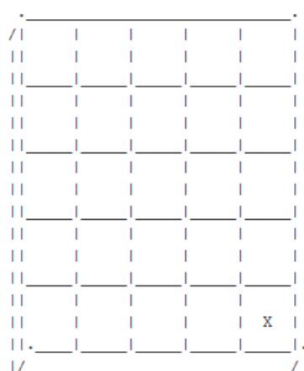
Then 'X' or 'Y' will replace the value ' ' at that position's address.

For example, we want to mark at position [5,5]



```
Player 1 in turn: thuan
[1] Play
[2] Undo
Option: 1
Choose x position (1 to 5): 5
Choose y position (1 to 5): 5
```

Clear

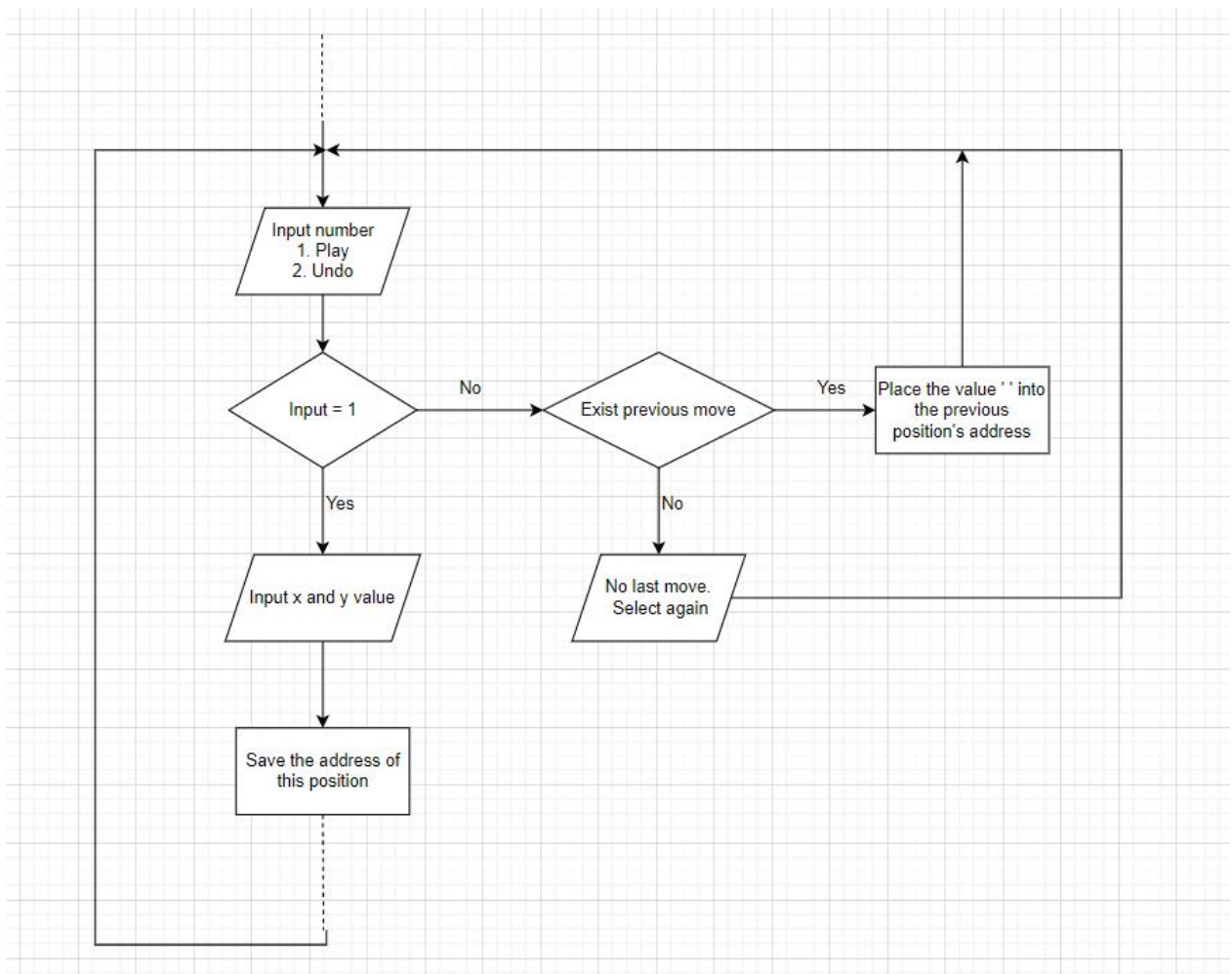


```
Player 2 in turn: thu
[1] Play
[2] Undo
Option: 
```

2.3 Undo the previous step

At the beginning of each player's turn, the program will ask the player to choose Play or Undo

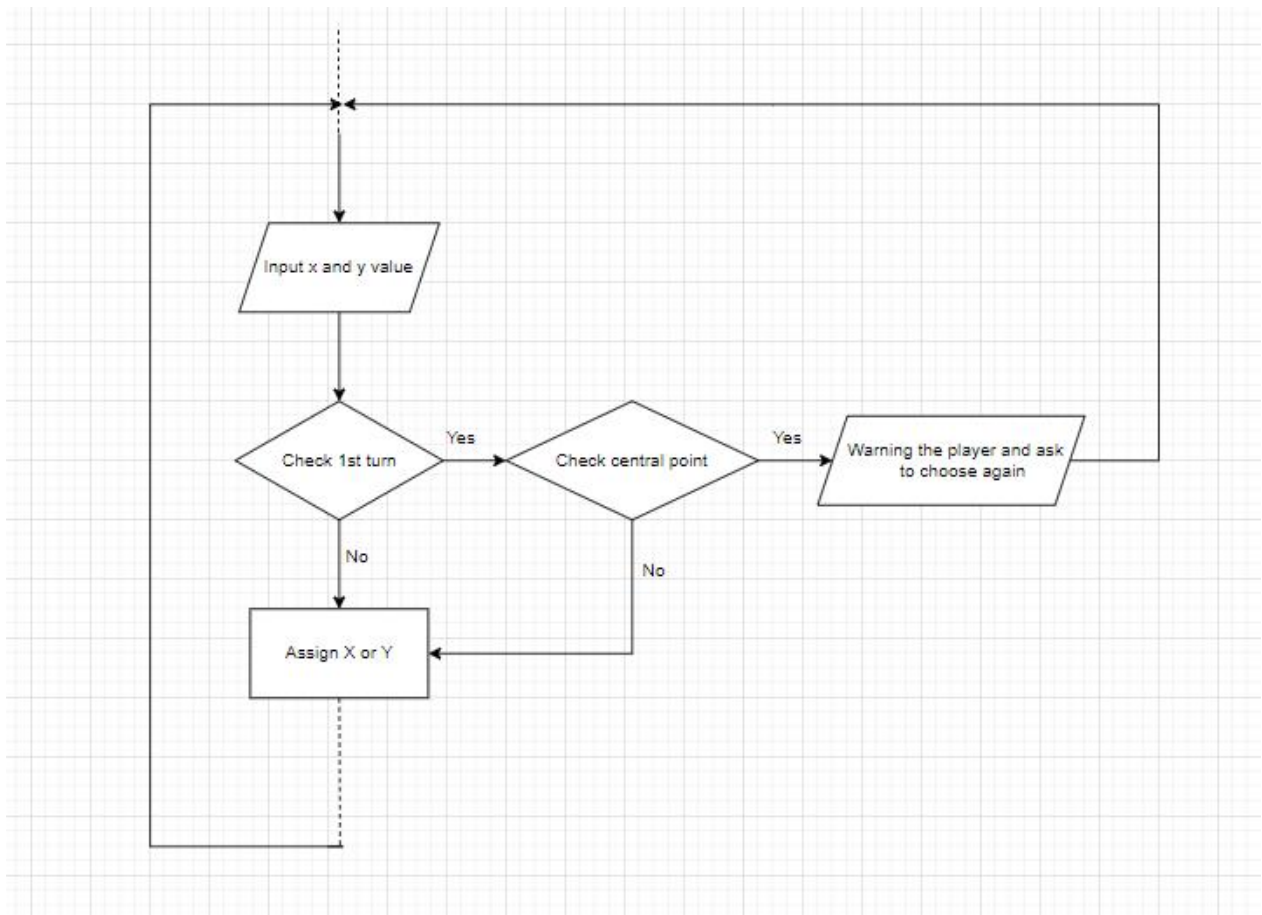
- When players choose Play, then they have to input x and y value and if they are not invalid values, the program will save this mark's address.
- After that, if the players choose Undo, the program will place the value ' ' into the previous saved address. In case the players did not place any marker in their previous turn, the game will print out: No last move and ask them to select again



2.4 Check central point in each player's 1st turn

In this part, when the players finish inputting x and y value, before placing the mark, the program will check if it is their 1st turn.

- If it is their 1st turn, it continue to check the mark is the central point or not. If it is a central point, the program will warn the players and require them to select again.
- If it is not their 1st turn, it continue to assign the mark.



2.5 Check winning conditions:

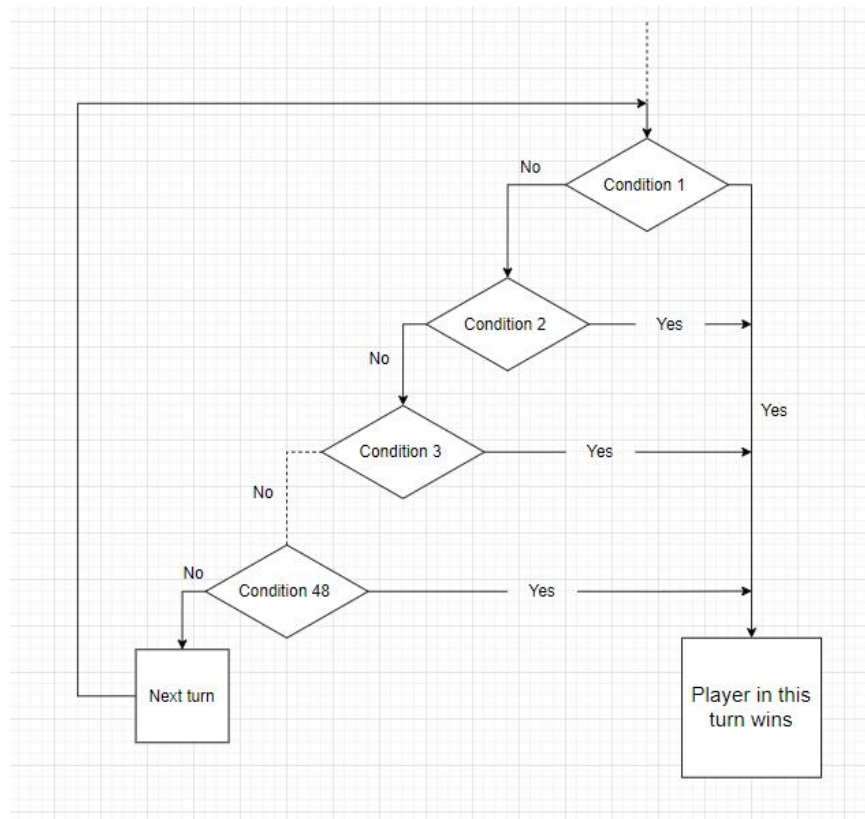
In order to check the winning conditions. We must know how to win in this game. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner.

So I decided to list all the possible winning cases.

0	4	8	12	16
20	24	28	32	36
40	44	48	52	56
60	64	68	72	76
80	84	88	92	96

Some examples of winning cases

There are total 48 winning cases. I will consider it step by step from the first condition to the forty-eighth condition.



For example :

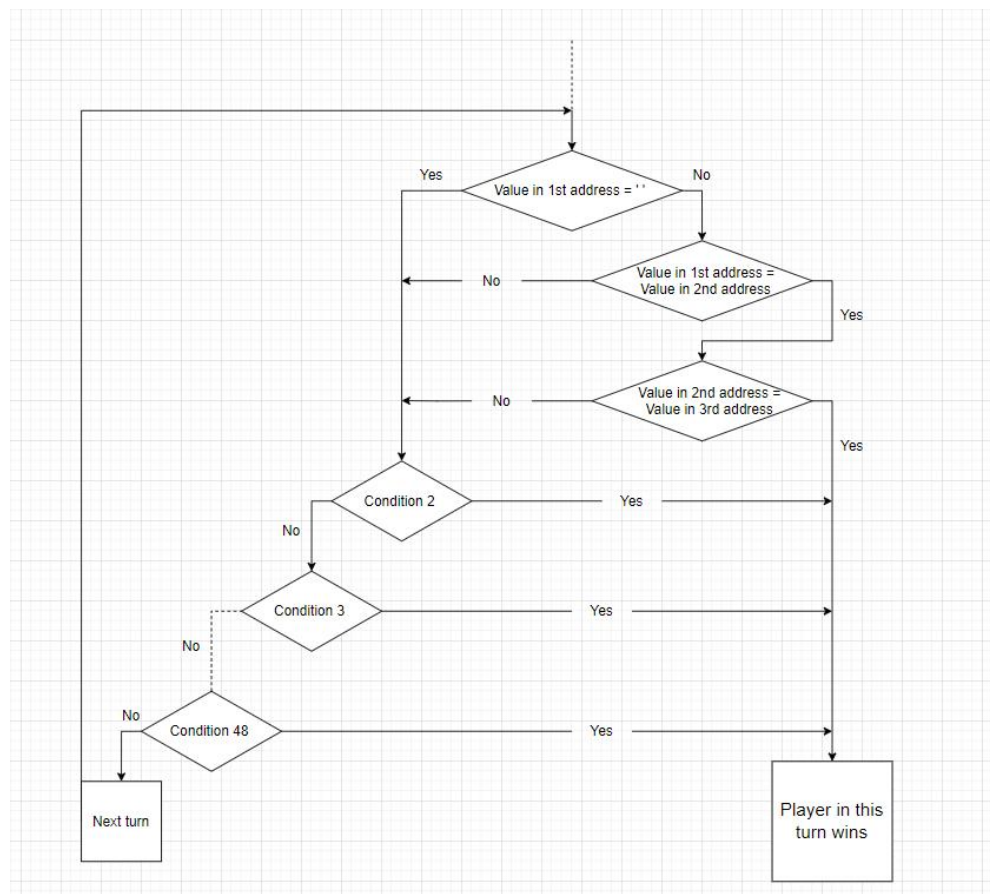
I assume that the condition 1 is 3 values' addresses {0,4,8}.

First of all, I will consider if the value in first address is as same as value ' '. If the answer is :

+ No. I will continue compare with the value in second and third address. If it is yes. I conclude the player in this turn wins.

If no, the program will consider condition 2. And do the same steps for the other conditions.

+ Yes. I will consider the condition 2.



Condition1 in detail

2.6 Check full board:

Last but not least, how to check the board is full or not. (All the values were assigned to 'X' or 'Y')

The main idea is to compare all the values of each address of the array with the value ' '. If the value is not equal to ' ', it means that the address's position has been marked.

For example :

- The array after some turns is 'X', 'Y', ' ', 'Y', 'X', ' ', ' ', ' ', ' '.

The procedure can be expressed as below :

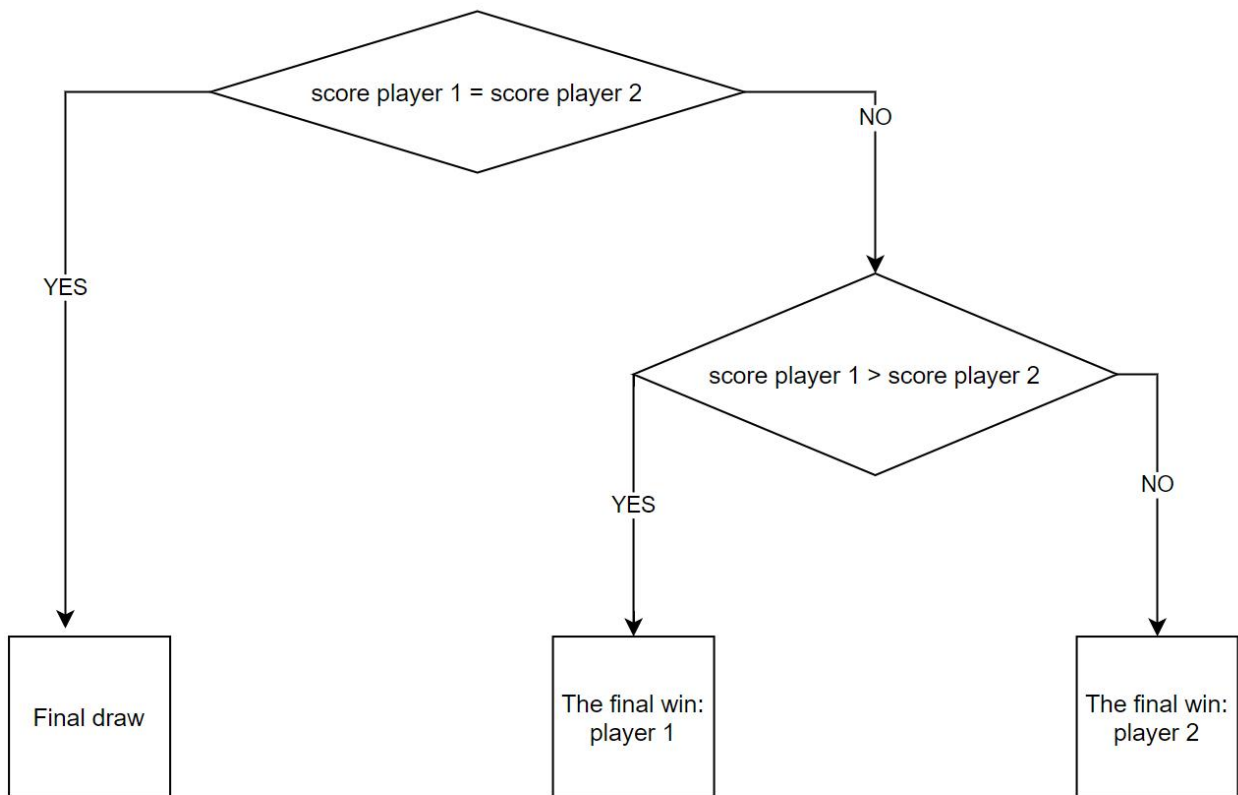
- 1) The program will compare the first character of array ('X') with ' '. It is different, it will consider to the second element.
- 2) 'Y' is different from ' '. Considering next element.
- 3) ' ' is same as ' '. The procedure is finish.

At this step, we can conclude that the board is not full.

2.7 The final result:

As I mentioned if we input number 99 in the Game Menu, the game will end. At that time, the program should display the final result for all score and decide the final result.

We will store a score of player 1 and player 2 in the array. After the player input number 99 in the Game Menu, we immediately compared the score of two player and display the output.



Final Result

3 Conclusion

Through this assignment, I have learned how to implement a simple Tic-Tac-Toe game in assembly language MIPS. I have learned more about using different MIPS instructions and logic of the game.

Thanks for reading.