VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY

UNIVERSITY OF TECHNOLOGY

FACULTY OF COMPUTER SCIENCE AND ENGINEERING

COMPUTER ARCHITECTURE LAB

ASSIGNMENT – TIC-TAC-TOE BY MIPS

STUDENT: NGUYỄN DUY BẢO

ID NUMBERS: 2052399

HO CHI MINH CITY, OCTOBER 2021

I. INTRODUCE:

1. Mips:

   MIPS (**Microprocessor without Interlocked Pipelined Stages**) is a reduced instruction set computer (RISC) instruction set architecture (ISA) developed by MIPS Computer Systems, now MIPS Technologies, based in the United States.

   Computer architecture courses in universities and technical schools often study the MIPS architecture. The architecture greatly influenced later RISC architectures such as Alpha. In March 2021, MIPS announced that the development of the MIPS architecture had ended as the company is making the transition to RISC-V.

2. Tic-tac-toe:

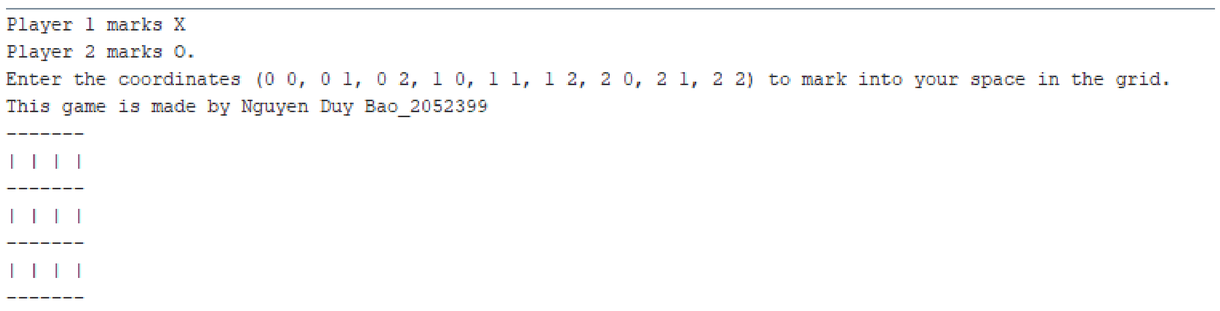
   Tic-tac-toe (American English), **noughts and crosses** (Commonwealth English), or **Xs and Os** (Irish English) is a paper-and-pencil game for two players who take turns marking the spaces in a three-by-three grid with *X* or *O*. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner. It is a solved game, with a forced draw assuming best play from both players.

   Tic-tac-toe is played on a three-by-three grid by two players, who alternately place the marks X and O in one of the nine spaces in the grid. Who gets 3 marks in a row or a column or a diagonal will win.

II. TIC-TAC-TOE BY MIPS:

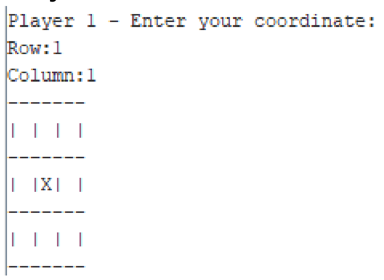
Firstly, player 1 will mark X, player 2 will mark O.
Enter the coordinates which is corresponding to the space you want to mark into.

(0;0)	(0;1)	(0;2)
(1;0)	(1;1)	(1;2)
(2;0)	(2;1)	(2;2)



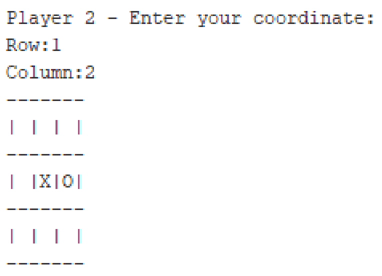
```
Player 1 marks X
Player 2 marks O.
Enter the coordinates (0 0, 0 1, 0 2, 1 0, 1 1, 1 2, 2 0, 2 1, 2 2) to mark into your space in the grid.
This game is made by Nguyen Duy Bao_2052399
-------
| | | |
-------
| | | |
-------
| | | |
-------
```

Player 1 marks X first



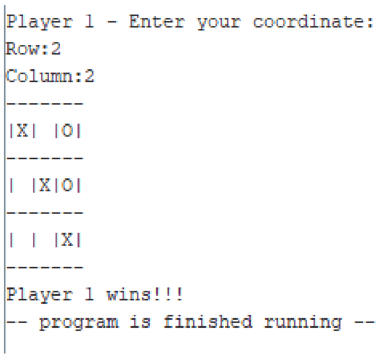
```
Player 1 - Enter your coordinate:
Row:1
Column:1
-------
| | | |
-------
| |X| |
-------
| | | |
-------
```

Then, player 2 marks O

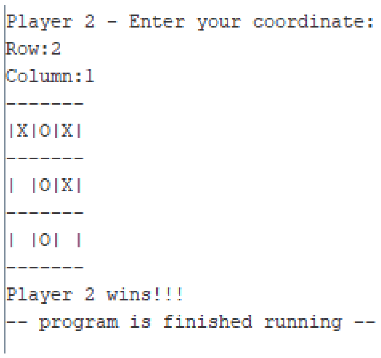


```
Player 2 - Enter your coordinate:
Row:1
Column:2
-------
| | | |
-------
| |X|O|
-------
| | | |
-------
```

This is when player 1 wins or player 2 wins. The game ends.

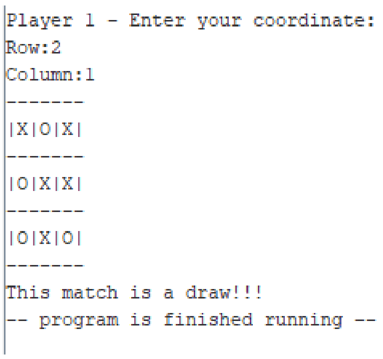


```
Player 1 - Enter your coordinate:
Row:2
Column:2
-------
|X| |O|
-------
| |X|O|
-------
| | |X|
-------
Player 1 wins!!!
-- program is finished running --
```



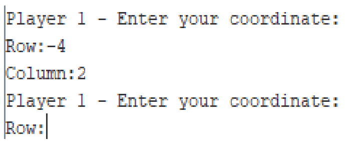
```
Player 2 - Enter your coordinate:
Row:2
Column:1
-------
|X|O|X|
-------
| |O|X|
-------
| |O| |
-------
Player 2 wins!!!
-- program is finished running --
```

Sometimes, the match is draw. The game ends.

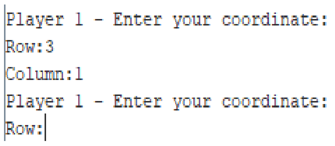


```
Player 1 - Enter your coordinate:
Row:2
Column:1
-------
|X|O|X|
-------
|O|X|X|
-------
|O|X|O|
-------
This match is a draw!!!
-- program is finished running --
```

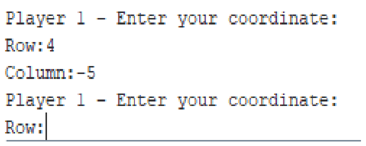
If players enter the wrong coordinate. The game will ask for the coordinate again.



```
Player 1 - Enter your coordinate:
Row:-4
Column:2
Player 1 - Enter your coordinate:
Row:
```



```
Player 1 - Enter your coordinate:
Row:3
Column:1
Player 1 - Enter your coordinate:
Row:
```



```
Player 1 - Enter your coordinate:
Row:4
Column:-5
Player 1 - Enter your coordinate:
Row:
```

III. CODE EXPLANATION:

1. Introduce:

Print “Player 1 marks X \nPlayer 2 marks O.\nEnter the coordinates (0 0, 0 1, 0 2, 1 0, 1 1, 1 2, 2 0, 2 1, 2 2) to mark into your space in the grid. \nThis game is made by Nguyen Duy Bao_2052399\n”
to introduce and instruct players to play.

2. Game play:

Ideas:

- Take a register $s0 to represent which player will win. If $s0 = 1, player 1 will win. If $s0 = 2, player 2 will win.
- Take a register $s7 to count the number of marks which is done. When $s7 = 9 and no one wins ($s0 = 0) => the match is draw.
- I use an array of byte to represent the grid 3x3, called “tick”. First the array will be 9,9,9,9,9,9,9,9,9. The coordinate will be calculated to find the position of the space in the grid:

  row*3 + column = the position

  example: the coordinate (2,1) => 2*3+1= position 7th: tick + 7

  the coordinate (0,0) => 0*3 + 1 = position 0th: tick+ 0
- Take a register $s3 to represent which player is playing and store into the array tick above
- If printboard function finds 9, the game print a blanket space “ ”. If printboard function finds 1, the game prints X. If printboard function finds 2, the game prints O.
- Check the player win or not by row, column and diagonal respectively. I will check after the input of player is valid. After checkwin, if player win, the $s0 will be replaced by 1 or 2. Otherwise, $s0 will not be changed and the game continue.

Some outstanding code and explanation:

main:

```
print the introduce
$s7 = 0     #The register to count the number of marks that is marked.
$s0 = 0     #Represent which player will win.
jal printboard #Print the blanket grid by function printboard
```

playloop:

loopplayer1: #if the checkinput1 function check if the input is not valid, return to this statement

Print and input the coordinate of player 1

$s3=1 #Set 1 means the next input and checkwin function is player 1's

Input row and column (coordinate of the place player1 want to mark).

Store coordinate into $s1 (row) and $s2 (column).

jal checkinput1 #function check if the input is valid. If not, the game will ask for the input again.

jal tickplayer #function to mark the place of the grid after the input is valid.

jal printboard #print the board after the place of grid is marked successfully.

jal checkwin #function to check if the player win or not. If not, continue the game ($s0 still equal to 0). Otherwise, $s0 will be 1 or 2, corresponding to player1 wins or player2 wins.

$s7 ++ (count++ the number of marks which is marked).

If $s0 = 1 (player1 wins), brand to statement player1win and end the game.

If $s7 = 9 (9 places is filled, the match is draw), brand to statement tie and end the game. Only check in the loop of player 1, because the last mark (9th) is belongs to player 1.

loopplayer 2: #the same to loopplayer 1

Print and input the coordinate of player 2

$s3=2 #Set 2 means the next input and checkwin function is player 2's.

Store coordinate into $s1 (row) and $s2 (column).

Jal checkinput2

Jal tickplayer

Jal printboard

```
        Jal checkwin
        $s7 ++
        If $s0 = 2, brand into statement player2 win and end the game.
    j playloop #jump back to statement playloop to input again. This loop
    will end when there is a result: player1 win ($s0 =1), player2 win ($s0
    =2), this match is draw ($s7=9).
```

Now, these are the statements and function used in the main.

```
player1win:
    Print the result: Player 1 wins
    j end
player2win:
    Print the result: Player 2 wins
    j end
tie:
    Print the result: This match is draw
end:
    End the program (End the game)
printboard:
    load address of array tick to $t0
    print the upper horizontal bar
    travel in the array and print the status of the places in the grid, which is
    corresponding to the position in the array. (I have explained in the ideas
    section):
        $t1 = 0 (use for the next loop)
        loop:
        If $t1 = 9, end this loop
        if array + $t0 : = 9 => print " "; =1=> print X; =2=> print O
            #I use some statement inside this loop: Xprint, Oprint,
    blankprint
        $t1++ #count the number of printed places.
```

```
                print the vertical bar in middle of them.
        print the power horizontal bar
checkinput1:
        If row or column<0, return loopplayer1
        If row or column>2, return loopplayer1
        Check if the place is marked before yet?:
        $t3 = row*3 + column = $s1*3 + $s2 (I have explained in the idea section)
        Load byte from array tick + $t3 to $t6
        If $t6 = 9 => ok, store 1 (means X) to the array tick
        Otherwise, return loopplayer1.
        If no return loopplayer1, jr $ra to get back to the checkinput1 sign.
checkinput 2:
        The same to checkinput1 but return to loopplayer 2 if the input is not valid. If it is valid, jr $ra.


tickplayer:
        $t0 = row*3 + column = $s1*3 + $s2 #Find the position in array
        Store $s3 (which means the mark is X or O, 1 means X, 2 means O) to array tick + $t0
        ja $ra
checkwin: (I check by 3 ways to win: win by row, win by column, win by diagonal)
        If $s1 = 0, brand into statement firstrow
        If $s1 = 1, brand into statement secondrow
        If $s1 = 2, brand into statement thirdrow


        firstrow:
                Check array tick in position: +0, +1, +2. If they are all equal to $s3 ($s3 represent which player is checking, $s3 =1 means player 1, $s3 = 2 means player 2)
```

If win, $s0 = $s3 and jr $ra to get back to the loopplayer and it will print the result.

If not, brand into checkcolumn.

secondrow: the same to firstrow, check +3, +4, +5

thirdrow: the same to firstrow, check +6, +7, +8

checkcolumn:

The same idea with check the rows,

If $s2 = 0 => firstcolumn: check +0, +3, +6

If $s2 = 1 => secondcolumn: check +1, +4, +7

If $s2 = 2 => thirdcolumn: check +2, +5, +8

If win, $s0 = $s3. Otherwise, brand to statement checkdiagonal

checkdiagonal: (there are only 2 cases: +0, +4,+8 and +2,+4,+6)

check the position tick+4 first. Then check +0, +8.

If win, $s0 = $s3. Otherwise, check +2, +6.

If win, $s0 = $s3. Otherwise, jr $ra and the game will continue.