ASSIGNMENT

FOR COMPUTER ARCHITECTURE

1. Requirement for assignment:

   Design and write MIPS assembly language for implementing a text-based Tic-Tac-Toe game for two players.

2. Introduction about **Tic-tac-toe** game.

Tic-tac-toe game is a **paper-and-pencil game** for two players, *X* and *O*, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a diagonal, horizontal, or vertical row is the winner.(the definition in Wikipedia)

3. Approach to solution.

1	2	3
4	5	6
7	8	9

Firstly, I will create an array, namely **myArray** and store 9 charaters (‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’, ‘7’, ‘8’, ‘9’) and at address (0, 4, 8, 12, 16, 20, 24, 28, 32) respectively.

Secondly, I use $t1 to get address of **myArray** and use $t0 to store character ‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’, ‘7’, ‘8’, ‘9’ to **myArray** at corresponding address.

```
la $t0, '1'
sb $t0 , myArray($t1)
addi $t1, $t1, 4
```

Next, I create 3 sub-functions, namely **printCharacter**, **printMarker** and **checkXorOWin.**

- **printCharacter** function is used to print the table, for example

```
 --- --- ---
| 1 | 2 | 3 |
 --- --- ---
| 4 | 5 | 6 |
 --- --- ---
| 7 | 8 | 9 |
 --- --- ---
```

- **printMarker** function is used to replace ‘X’ or ‘O’ charater to **myArray** whenever user enter a number from 1 to 9. For example, when user enter number 1 then **myArray** will store ‘O’ character like the picture below.

```
The first player selects a number(from 1 - 9): 1
 --- --- ---
| O | 2 | 3 |
 --- --- ---
| 4 | 5 | 6 |
 --- --- ---
| 7 | 8 | 9 |
 --- --- ---
```

- Beside, **printMarker** function also checks whether or not a cell is filled by 'X' or 'O'. if there exists charater 'X' or 'O' then assign value 1 to $v1 to tell the user enter a different number in main function.

```
# check if there exists charater 'X' or 'O'
# if there exists charater 'X' or 'O',then tell user enter a different number
lb $s3, myArray($s0)
la $t3, 'O'
beq $t3, $s3, ext
la $t3, 'X'
beq $t3, $s3, ext



ext:
        # return $v1 = 1 to know the charater 'X' or 'O' has already existed
        addi $v1, $zero, 1
```

- **checkXorOWin** function will check 3 charaters 'X' or 'O' in a diagonal, horizontal, or vertical row.
  - I use $t5 for 'X' count variable, $t8 for 'O' count variable, and $t6 for count variable when checking 3 times in a diagonal, horizontal, or vertical row.

```
sw $t5, 16($sp) # 'X' count
sw $t6, 20($sp) # use for count 3 times
sw $t8, 28($sp) # 'O' count
```

  - At the beginning, when checking a horizontal row, I will firstly check whether or not 'X' character has been in **myArray** by using following code.

```
# check myArray contains 'X' or not, if not move to chek lable
la $t3, 'X'
lb $t2, myArray($t1)
bne $t2, $t3, chek
```

  - If **myArray** contains 'X' then increase $t5 to 1.

```
# if a row contains 'X' then increase $t5 to 1
addi $t5, $t5, 1
```

  - If this slot is not character 'X', then move to the **chek** label to check character 'O'. If **myArray** contains 'O' then increase $t8 to 1. If this slot is not character 'O' then move to the next slot(move to **for3** label) to check.

```
chek:
# to check myArray contains 'O' or not
la $t3, 'O'
lb $t2, myArray($t1)

# if myArray does not contain 'O' then move to for3 label
bne $t2, $t3, for3
# if a row contains 'X' then increase $t8 to 1
addi $t8, $t8, 1


j for3
```

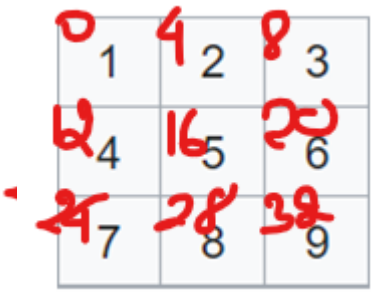
- Whenever $t6 is equal to 3 then move to **check** label to check if there are enough 3 charaters ‘X’ or ‘O’ in that horizontal row($t5 == 3 or $t8 == 3) then assign ‘X’ to $v1($t5 == 3) or ‘O’ to $v1($t8 == 3), respectively. Else if there are not enough then reset values of $t5, $t6 and $t8 to 0.

```
check:
        # check if contains enough 3 'X' or 3 '0' characters then go to out and out10 lable store 'X' and '0' character to $v1 respectively
        beq $t5, 3, out
        beq $t8, 3, out10
        # reset $t5, $t8, $t6 to 0 if myArray does not contain enough 3 'X' or 3 '0' characters
        addi $t5, $zero, 0
        addi $t8, $zero, 0
        addi $t6, $zero, 0

        j for3
out:
        la $v1, 'X'

out10:
       la $v1, '0'
```

- Checking vertically for **myArray** is similar when checking horizontally, except for computing to get correct address index of **myArray** .
- Lastly, I will check diagonal row for ‘X’ and ‘O’ whenever checking vertically and checking horizontally are wrong.



At the beginning, I check a diagonal row 1-5-9 which has address index 0, 16, 32 respectively. And I set all value to 0. Then I get value from the memory at index 0 and check if there is ‘X’ charater, then I will add 1 to $t5 to count ‘X’ character and do some computations to assign address index 16 to $t1. And do similarly with rest of the code in the right side. Whenever there are enough 3 ‘X’characters then assign ‘X’ to $v1 at **out2** label. In the contrast, if one of slots at 1, 5, 9 is not enough ‘X’ then move to **exit5** label to check remaining diagonal row(3-5-7) with ‘X’ character.

Check again diagonally with ‘O’ character if there are not enough 3 characters ‘X’ in a vertical row.

```
        addi $t1, $zero, 0
        addi $t5, $zero, 0
        addi $t4, $zero, 0
        addi $t7, $zero, 0

        la $t3, 'X'
        lb $t2, myArray($t1)
        bne $t2, $t3, exit5
        addi $t5, $t5, 1
        addi $t7, $t7, 4
        mul $t7, $t7, 4
        move $t1, $t7
        div $t7, $t7, 4

        lb $t2, myArray($t1)
        bne $t2, $t3, exit5
        addi $t5, $t5, 1
        addi $t7, $t7, 4
        mul $t7, $t7, 4
        move $t1, $t7
        div $t7, $t7, 4

        lb $t2, myArray($t1)
        bne $t2, $t3, exit5
        beq $t5, 3, out2

out2:
        la $v1, 'X'
```

- After creating 3 sub-functions, I will implement main function.

Firstly, program will ask player to chose number 1 or 2 in order to become first player or second player. However, if player a random number which is not 1 or 2 then the program throw an error "Please select the correct option!!!!!", like figure in the right side.

```
Again:
li $v0, 4
la $a0, input9
syscall
li $v0, 4
la $a0, input12
syscall
li $v0, 5
syscall
move $s0, $v0

slti $s5, $s0, 3
beq $s5, $zero, again1
slt $s5, $zero, $s0
beq $s5, $zero, again1
bne $s5, $zero, while

again1:
li $v0, 4
la $a0, input10
syscall
j Again
```

```
Who do you want to become?
1. The first player: O
2. The second player: X
Enter your selection:
```

```
Who do you want to become?
1. The first player: O
2. The second player: X
Enter your selection: 4
Please select the correct option!!!!!
Who do you want to become?
1. The first player: O
2. The second player: X
Enter your selection:
```

- Secondly, whenever a player (1 or 2) select a number from 1 to 9 then **myArray** will store that selected number by using **printMarker**. If player enter a number which is different from 1 to 9 then the program move to **hack1** label to force that player enter number again.

```
div $s1, $s4 # $s1 / 2
mfhi $s3
# whenever $s1 is odd then ask player 2 for entering a number, else if $s1 is even then ask player 1 for entering a number
bne $s3, 0, moveto
li $v0, 4
la $a0, input1
syscall
j here
moveto:
li $v0, 4
la $a0, input2
syscall
# player enters a number
here:
li $v0, 5
syscall
move $s0, $v0
# check whether or not that selected number is less than 10
# if not move to hack1 lable to force player enter a different number again
slti $s5, $s0, 10
beq $s5, $zero, hack1
slt $s5, $zero, $s0
beq $s5, $zero, hack1
jal printMarker
bne $v1, $zero, hack1


hack1:

        li $v0, 4
        la $a0, input8
        syscall
        addi $v1, $zero, 0
        j while
```

- Finally, check player 1 or player 2 who is win by the sub-function namely **checkXorOXin.** If function **checkXorOXin** return 'X' to $v1 then move to **exitX** label to print the table and print the string "The second player is the winner!!!!" and exit the program. Else return 'O' to $v1 then move to **exitO** label to print the table and print the string "The first player is the winner!!!!".

```
jal checkXor0Win
la $s2, 'X'
beq $v1, $s2, exitX
la $s2, '0'
beq $v1, $s2, exit0
```

```
exitX:
addi $t1, $zero, 0
jal printCharacter
li $v0, 4
la $a0, input7
syscall
j down
```

```
exit0:
addi $t1, $zero, 0
jal printCharacter
li $v0, 4
la $a0, input5
syscall
j down
```

- If after 9 turns then the program move to **exit** label to print the string "The result is draw!!!" which is stands for no one is the winner and finish program.

```
# count 9 times to exit
beq $s1, 9, exit
```

```
exit:
li $v0, 4
la $a0, input11
syscall
```

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH TRƯỜNG

ĐẠI HỌC BÁCH KHOA

o0o

ASSIGNMENT COMPUTER ARCHITECTURE

GVHD: Phạm Quốc Cường

SVTH: **Cao Tuấn Kiệt – 2053166**