TIC-TAC-TOE

21CSS101J – PROGRAMMING FOR PROBLEM SOLVING

Mini Project Report

Submitted by

Aryan Dangwal [RA2211004010433]
B.Tech. CSE -<<core>>

Tanay Gupta [RA2211003010406]
B.Tech. CSE - <<core>>

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

SCHOOL OF COMPUTING
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
(Under Section 3 of UGC Act, 1956)
S.R.M. NAGAR, KATTANKULATHUR – 603 203
KANCHEEPURAM DISTRICT
December 2022

TABLE OF CONTENTS

ABOUT TIC-TAC-TOE

Tic-Tac-Toe is a very simple two-player game. So only two players can play at a time. This game is also known as Noughts and Crosses or **X's and O's** game. One player plays with X and the other player plays with O. In this game we have a board consisting of a 3X3 grid. The number of grids may be increased.

Game Rules

Traditionally the first player plays with "X". So you can decide who wants to go with "X" and who wants to go with "O".

Only one player can play at a time.

If any of the players have filled a square then the other player and the same player cannot override that square.

There are only two conditions that may match will be draw or may win.

The player that succeeds in placing three respective marks (X or O) in a horizontal, vertical, or diagonal row wins the game

ALGORITHM FOR CODE OF TIC TAC TOE IN PHTHON LANGUAGE

We will now discuss the algorithm to write the code. This algorithm will help you to write code in any programming language of your choice. Let’s see how it’s done.

- Create a board using a 2-dimensional array and initialize each element as empty.
    - You can represent empty using any symbol you like. Here, we are going to use a hyphen. '-'.
- Write a function to check whether the board is filled or not.
    - Iterate over the board and return false if the board contains an empty sign or else return true.
- Write a function to check whether a player has won or not.
    - We have to check all the possibilities that we discussed in the previous section.
    - Check for all the rows, columns, and two diagonals.
- Write a function to show the board as we will show the board multiple times to the users while they are playing.
- Write a function to start the game.

Select the first turn of the player randomly.

Write an infinite loop that breaks when the game is over (either win or draw).

- Show the board to the user to select the spot for the next move.
- Ask the user to enter the row and column number.
- Update the spot with the respective player sign.
- Check whether the current player won the game or not.

        - If the current player won the game, then print a winning message and break the infinite loop.
        - Next, check whether the board is filled or not.
        - If the board is filled, then print the draw message and break the infinite loop.
    - Finally, show the user the final view of the board.

You may be able to visualize what’s happening. Don’t worry, even if you didn’t understand it completely. You will get more clarity once you see the code.

So, let’s jump into the code section. I assume you have Python installed on your PC to try the code.

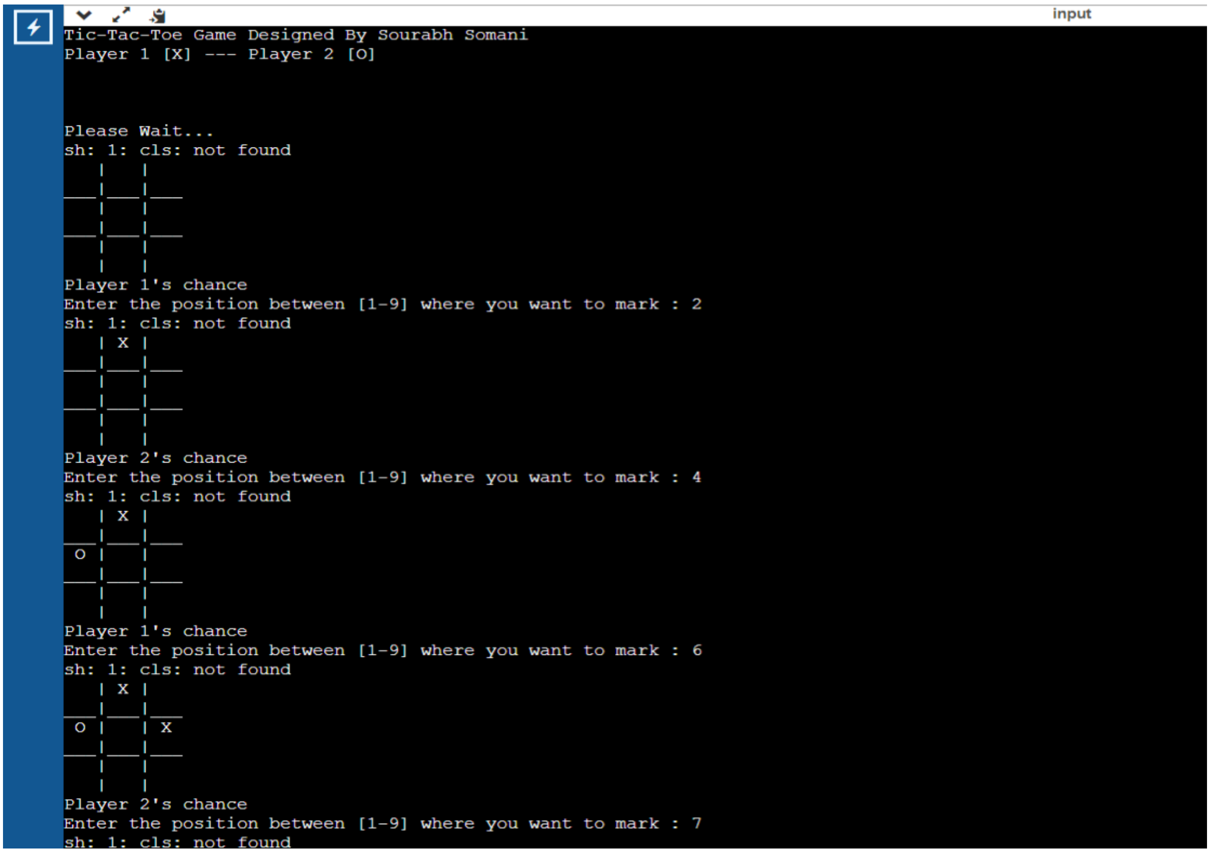
code

```
IMPORT OS
IMPORT TIME
BOARD = [' ',' ',' ',' ',' ',' ',' ',' ',' ',' ']
PLAYER = 1
WIN FLAGS##########
WIN = 1
DRAW = -1
RUNNING = 0
STOP = 1
###########################
GAME = RUNNING
MARK = 'X'
#THIS FUNCTION DRAWS GAME BOARD
DEF DRAWBOARD():
    PRINT(" %C | %C | %C " % (BOARD[1],BOARD[2],BOARD[3]))
    PRINT("___|___|___")
    PRINT(" %C | %C | %C " % (BOARD[4],BOARD[5],BOARD[6]))
    PRINT("___|___|___")
    PRINT(" %C | %C | %C " % (BOARD[7],BOARD[8],BOARD[9]))
    PRINT("   |   |   ")
#THIS FUNCTION CHECKS POSITION IS EMPTY OR NOT
DEF CHECK POSITION(X):
    IF(BOARD[X] == ' '):
        RETURN TRUE
    ELSE:
        RETURN FALSE
#THIS FUNCTION CHECKS PLAYER HAS WON OR NOT
```

```
DEF CHECKWIN():
    GLOBAL GAME
    #HORIZONTAL WINNING CONDITION
    IF(BOARD[1] == BOARD[2] AND BOARD[2] == BOARD[3] AND BOARD[1] != ' '):
        GAME = WIN
    ELIF(BOARD[4] == BOARD[5] AND BOARD[5] == BOARD[6] AND BOARD[4] != ' '):
        GAME = WIN
    ELIF(BOARD[7] == BOARD[8] AND BOARD[8] == BOARD[9] AND BOARD[7] != ' '):
        GAME = WIN
    #VERTICAL WINNING CONDITION
    ELIF(BOARD[1] == BOARD[4] AND BOARD[4] == BOARD[7] AND BOARD[1] != ' '):
        GAME = WIN
    ELIF(BOARD[2] == BOARD[5] AND BOARD[5] == BOARD[8] AND BOARD[2] != ' '):
        GAME = WIN
    ELIF(BOARD[3] == BOARD[6] AND BOARD[6] == BOARD[9] AND BOARD[3] != ' '):
        GAME=WIN
    #DIAGONAL WINNING CONDITION
    ELIF(BOARD[1] == BOARD[5] AND BOARD[5] == BOARD[9] AND BOARD[5] != ' '):
        GAME = WIN
    ELIF(BOARD[3] == BOARD[5] AND BOARD[5] == BOARD[7] AND BOARD[5] != ' '):
        GAME=WIN
    #MATCH TIE OR DRAW CONDITION
    ELIF(BOARD[1]!=' ' AND BOARD[2]!=' ' AND BOARD[3]!=' ' AND BOARD[4]!=' ' AND BOARD[5]!=' ' AND BOARD[6]!=' ' AND BOARD[7]!=' ' AND BOARD[8]!=' ' AND BOARD[9]!=' '):
        GAME=DRAW
    ELSE:
        GAME=RUNNING
PRINT("TIC-TAC-TOE GAME DESIGNED BY SOURABH SOMANI")
PRINT("PLAYER 1 [X] --- PLAYER 2 [O]\N")
PRINT()
PRINT()
PRINT("PLEASE WAIT...")
TIME.SLEEP(3)
WHILE(GAME == RUNNING):
    OS.SYSTEM('CLS')
    DRAWBOARD()
    IF(PLAYER % 2 != 0):
        PRINT("PLAYER 1'S CHANCE")
        MARK = 'X'
    ELSE:
        PRINT("PLAYER 2'S CHANCE")
        MARK = 'O'
    CHOICE = INT(INPUT("ENTER THE POSITION BETWEEN [1-9] WHERE YOU WANT TO MARK : "))
    IF(CHECKPOSITION(CHOICE)):
        BOARD[CHOICE] = MARK
        PLAYER+=1
        CHECKWIN()
OS.SYSTEM('CLS')
DRAWBOARD()
```

```
IF(GAME==DRAW):
    PRINT("GAME DRAW")
ELIF(GAME==WIN):
    PLAYER-=1
    IF(PLAYER%2!=0):
        PRINT("PLAYER 1 WON")
    ELSE:
        PRINT("PLAYER 2 WON")
```

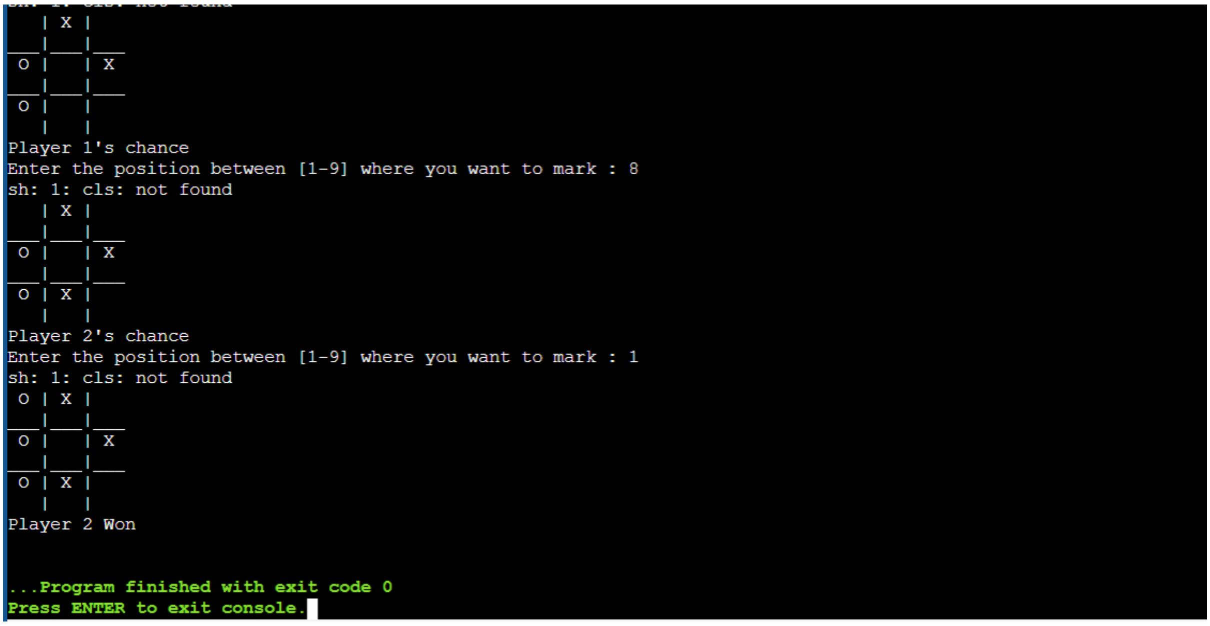
Results



```
Tic-Tac-Toe Game Designed By Sourabh Somani
Player 1 [X] --- Player 2 [O]



Please Wait...
sh: 1: cls: not found
   |   |
___|___|___
   |   |
___|___|___
   |   |
   |   |
Player 1's chance
Enter the position between [1-9] where you want to mark : 2
sh: 1: cls: not found
   | X |
___|___|___
   |   |
___|___|___
   |   |
   |   |
Player 2's chance
Enter the position between [1-9] where you want to mark : 4
sh: 1: cls: not found
   | X |
___|___|___
 O |   |
___|___|___
   |   |
   |   |
Player 1's chance
Enter the position between [1-9] where you want to mark : 6
sh: 1: cls: not found
   | X |
___|___|___
 O |   | X
___|___|___
   |   |
   |   |
Player 2's chance
Enter the position between [1-9] where you want to mark : 7
sh: 1: cls: not found
```



```
sh: 1: cls: not found
   | X |
___|___|___
 O |   | X
___|___|___
 O |   |
   |   |
Player 1's chance
Enter the position between [1-9] where you want to mark : 8
sh: 1: cls: not found
   | X |
___|___|___
 O |   | X
___|___|___
 O | X |
   |   |
Player 2's chance
Enter the position between [1-9] where you want to mark : 1
sh: 1: cls: not found
 O | X |
___|___|___
 O |   | X
___|___|___
 O | X |
   |   |
Player 2 Won


...Program finished with exit code 0
Press ENTER to exit console.
```

CONCLUSION

We have created a game completely from scratch. It is not one of the visual games that we play daily. But it helps you to write logic and maintain clean structure in code.

This GUI framework is available in python standard library. It runs on windows ,macOS ,linux , so your game will work great on all three of these platforms.

THANK YOU