24/9/24

Bafna Gold
Date: Page:

Tic Tac Toe Game

Algorithm :

1. Create a 3x3 array board[3][3].
2. Player is given choice for the type of move chosen (X) (or) (O).
3. Initialize the board [3][3] = '-'
4. function : check_for_winner ( )

```
for i in range(3):
    if board[i][0] == board[i][1] == board[i][2]:
        return board[i][0]
    if board[0][i] == board[1][i] == board[2][i]:
        return board[0][i]
```

// Check for diagonals

```
if board[0][0] == board[1][1] == board[2][2]
    return board[0][0]
if board[0][2] == board[1][1] == board[2][0]
    return board[0][2]
```

// If any above conditions is satisfied that symbol is returned as winner.

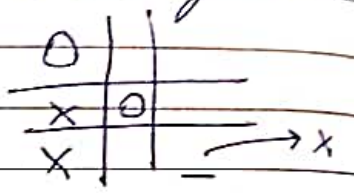
5. function : draw_condition ( )
   if all the cells are full and there is no winner condition return as draw game.
6. For AI_move we have 2 functions almost_win ( ) and to_win ( ).

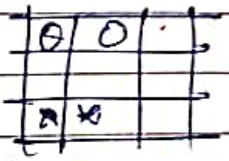
7. almost_win( ) function

here program must check the current state of board.

if any row or column or diagonal has two similar elements that would make the player win, such condition must be blocked by AI move.

if the almost_win conditions are true symbol 'x' is placed to avoid winning situation of user.





8. to_win( ) function

this function brings up the winning condition for AI.

if almost_win( ) conditions are false

AI can make move to match winning condition

else

make random choice.

9. print_board( ) function

board is printed on every move and winner is printed when board is full or check_winner is true.

10. if almost_win( ) & to_win( ) conditions are true AI will prioritize to_win( ) function.

Code:

```
import random
def print_board (board):
    for row in board:
        print("|".join(row))
        print ("-" * 9)

def check_winner (board):
    for i in range (3):
        if board [i][0] == board[i][1] == board [i][2] !=
            return board[i][0]                    " ":
        if board [0][i] == board[1][i] == board [2][i] != " ":
            return board [0][i]
    if board [0][0] == board[1][1] == board [2][2] != " ":
        return board [0][0]
    if board [0][2] == board [1][1] == board [2][0] != " ":
        return board [0][2]
    return None

def is_board_full (board):
    return all (cell != " " for row in board for cell
                in row)

def get_available_moves (board):
    return [(i, j) for i in range(3) for j in
            range(3) if board[i][j] == " "]

def human_move (board):
    while True:
        try:
            move = int (input (" Enter your move (1-9):"))-1
```

```
        if move < 0 or move > 8:
            raise ValueError
        row, col = divmod(move, 3)
        if board[row][col] == " ":
            return row, col
        else:
            print("Cell already taken, try again")
    except ValueError:
        print("Invalid input")

def computer_move(board):
    for move in get_available_moves(board):
        board[move[0]][move[1]] = "O"
        if check_winner(board) == "O":
            return move
        board[move[0]][move[1]] = " "

    for move in get_available_moves(board):
        board[move[0]][move[1]] = "X"
        if check_winner(board) == "X":
            board[move[0]][move[1]] = "O"
            return move
        board[move[0]][move[1]] = " "
    return random.choice(get_available_moves(board))

def main():
    board = [[" " for _ in range(3)] for _ in range(3)]
    print("Welcome to Tic Tac Toe! You are 'X'
          and the computer is 'O'.")
    while True:
        print_board(board)
        row, col = human_move(board)
```

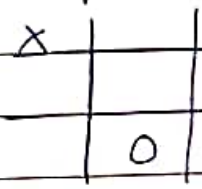
```
            board[row][col] = "X"
            if check_winner(board) == "X":
                print_board(board)
                print("Congratulations! You win!")
                break
            if is_board_full(board):
                print_board(board)
                print("It is a tie!")
                break
            print("Computer's turn")
            row, col = computer_move(board)
            board[row][col] = "O"

            if check_winner(board) == "O":
                print_board(board)
                print("Computer wins!")
                break
            if is_board_full(board):
                print_board(board)
                print("It's a tie")
                break

if __name__ == "__main__":
    main()
```

Output:

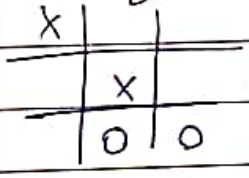
Welcome to Tic Tac Toe! You are 'X' and the computer is 'O'

Enter your move (1-9): 1

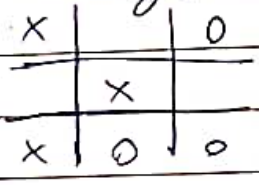
Computer's turn:



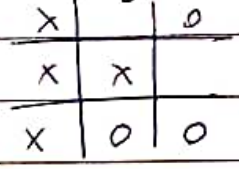
Enter your move (1-9): 5



Enter your move (1-9): 7



Enter your move (1-9): 4



Congratulations! You Win!