REPORT

***Title**: Tic Tac Toe Game Using Python with AI and User Interaction*

By :- SHUBHAM KUMAR

University roll :- 202401100300244

Branch :- CSEAI-D

Introduction

The Tic Tac Toe game is a popular two-player puzzle game that serves as an ideal project for implementing logic and interactivity in Python. In this project, a User competes against an AI opponent, demonstrating the use of decision-making and random choice functions. This program is an excellent way to understand programming concepts like loops, functions, and conditional statements. The AI makes its moves strategically using randomization, providing a basic but enjoyable challenge for the user.

Methodology

The game is implemented in Python and follows these steps:

1. **Game Initialization**:
   - A 3x3 grid (board) is created as a list of lists filled with empty spaces (" ").
   - Two players are defined: User and AI, with their respective symbols as X and O.
2. **User Interaction**:
   - The user is prompted to input a row and column to make their move.
   - Invalid inputs or attempts to play on an occupied cell are handled gracefully by asking the user to try again.
3. **AI Logic**:

- The AI identifies all empty cells on the board and chooses one randomly using the random module to make its move.

4. **Game Rules**:
   - After each turn, the program checks for a winner or a tie using the check_winner and is_full functions.
   - The game alternates turns between the User and the AI until a result (win or tie) is determined.
5. **Output**:
   - The game board is printed after every move for visual clarity.
   - A message declares the result, whether the User won, AI won, or it's a tie.

Code Typed

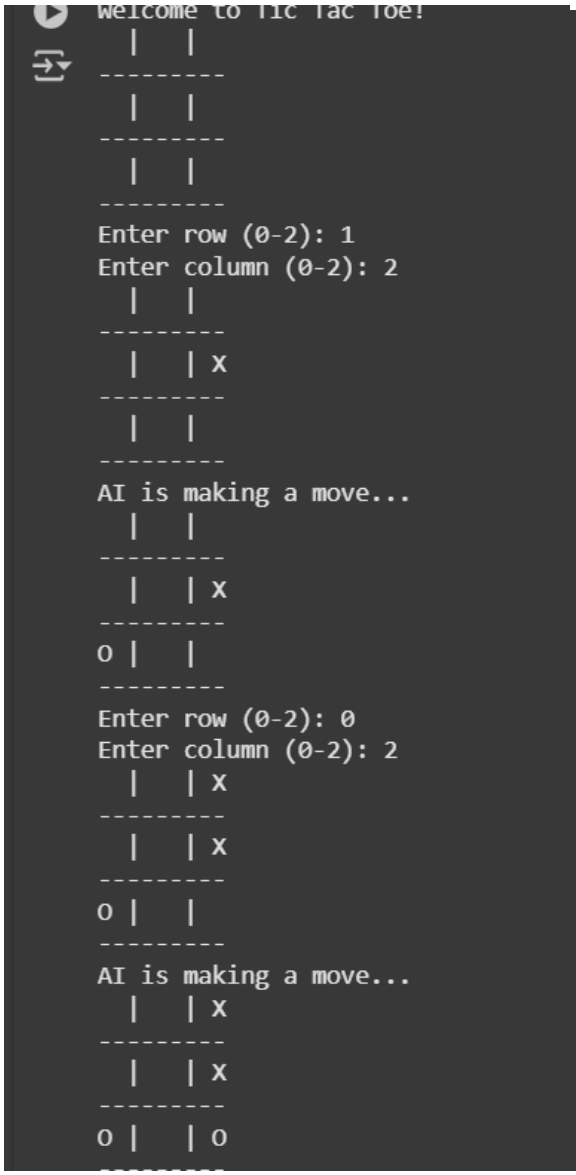
```
import random


# Function to print the game board
def print_board(board):
    for row in board:
        print(" | ".join(row))
        print("-" * 9)


# Function to check if a player has won
def check_winner(board, player):
    for row in board:
        if all(s == player for s in row):
            return True

    for col in range(3):
        if all(row[col] == player for row in board):
            return True
```

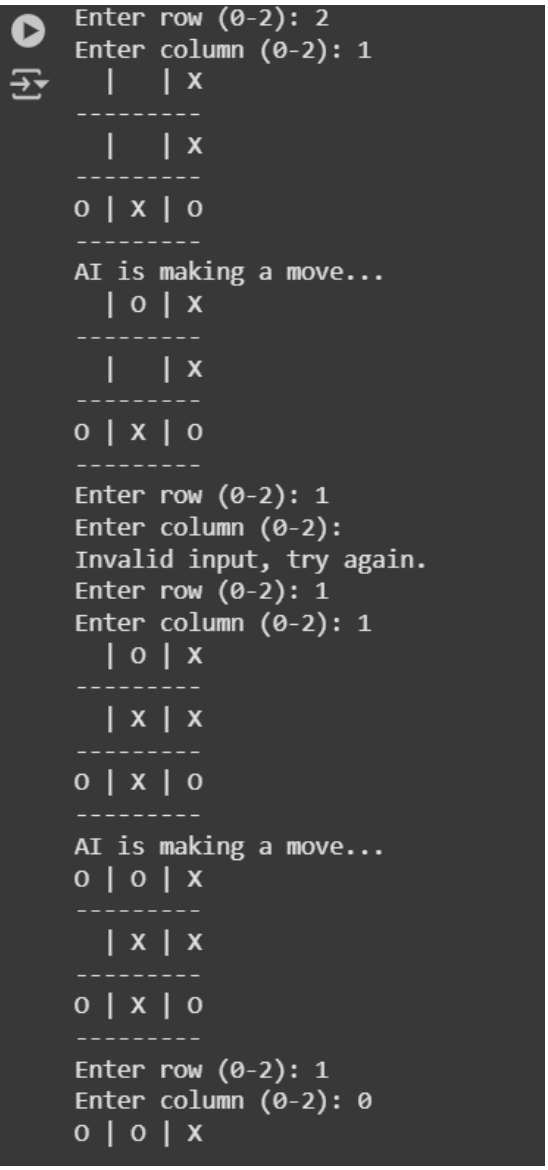
```
    if all(board[i][i] == player for i in range(3)) or all(board[i][2 - i] == player for i in range(3)):
        return True

    return False

# Function to check if the board is full
def is_full(board):
    return all(cell != " " for row in board for cell in row)

# Function for the AI to make a move
def ai_move(board):
    empty_cells = [(i, j) for i in range(3) for j in range(3) if board[i][j] == " "]
    return random.choice(empty_cells)

# Main game loop
def tic_tac_toe():
    board = [[" " for _ in range(3)] for _ in range(3)]
    players = ["User", "AI"]
    symbols = {"User": "X", "AI": "O"}
```

```
current_player = "User"

print("Welcome to Tic Tac Toe!")
print_board(board)

while True:
    if current_player == "User":
        # User move
        try:
            row = int(input("Enter row (0-2): "))
            col = int(input("Enter column (0-2): "))
            if board[row][col] != " ":
                print("Cell already taken, try again.")
                continue
        except (ValueError, IndexError):
            print("Invalid input, try again.")
            continue

    else:  # AI move
        print("AI is making a move...")
```

```
            row, col = ai_move(board)

        board[row][col] = symbols[current_player]
        print_board(board)

        if check_winner(board, symbols[current_player]):
            print(f"{current_player} wins!")
            break

        if is_full(board):
            print("It's a tie!")
            break

        # Switch player
        current_player = "AI" if current_player == "User" else "User"

# Start the game
tic_tac_toe()
```

Screenshots/outputs






Enter row (0-2): 1
Enter column (0-2):
Invalid input, try again.
Enter row (0-2): 1
Enter column (0-2): 1
| O | X

| X | X

O | X | O

AI is making a move...
O | O | X

| X | X

O | X | O

Enter row (0-2): 1
Enter column (0-2): 0
O | O | X

X | X | X

O | X | O

User wins!

References

1. Python Documentation: https://docs.python.org/3/ *The official Python documentation was referred to for understanding functions like* random.choice *and other Python basics.*

2. Online Tutorials:

- "How to Code Tic Tac Toe in Python" - TutorialsPoint
- "Building a Simple Python Game" - Real Python

3. Personal Learning: The project was built using concepts from Python programming courses and hands-on practice with decision-making and function implementation.

4. AI Interaction: Guidance provided by Microsoft Copilot for structured code development and explanation.