Studio Shodwe

CLASSWORK PROJECT

Presented by: Maisa_Shad

TicTacToe

```python
import random

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

def check_winner(board, player):
    for row in board:
        if all(cell == player for cell in row):
            return True

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

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

def get_empty_cells(board):
    return [(row, col) for row in range(3) for col in range(3) if board[row][col] == " "]
```

```python
def minimax(board, depth, maximizing):
    if check_winner(board, "X"):
        return -1
    if check_winner(board, "O"):
        return 1
    if is_board_full(board):
        return 0

    if maximizing:
        max_eval = -float("inf")
        for row, col in get_empty_cells(board):
            board[row][col] = "O"
            eval = minimax(board, depth + 1, False)
            board[row][col] = " "
            max_eval = max(max_eval, eval)
        return max_eval
    else:
        min_eval = float("inf")
        for row, col in get_empty_cells(board):
            board[row][col] = "X"
            eval = minimax(board, depth + 1, True)
            board[row][col] = " "
            min_eval = min(min_eval, eval)
        return min_eval
```

```
def get_best_move(board):
    best_move = None
    best_eval = -float("inf")
    for row, col in get_empty_cells(board):
        board[row][col] = "O"
        eval = minimax(board, 0, False)
        board[row][col] = " "
        if eval > best_eval:
            best_eval = eval
            best_move = (row, col)
    return best_move

def tic_tac_toe():
    board = [[" " for _ in range(3)] for _ in range(3)]
    current_player = "X"

    print("Welcome to Tic-Tac-Toe!")

    while True:
        print_board(board)

        if current_player == "X":
            row, col = -1, -1
            while row not in range(3) or col not in range(3) or board[row][col] != " ":
                try:
                    row, col = map(int, input(f"Player {current_player}, enter your row and column (e.g., 1 2): ").split())
                    row -= 1
                    col -= 1
                except ValueError:
                    pass
```

```
            board[row][col] = current_player
        else:
            print(f"Player {current_player} (Bot) is thinking...")
            row, col = get_best_move(board)
            board[row][col] = current_player

        if check_winner(board, current_player):
            print_board(board)
            print(f"Player {current_player} wins! Congratulations!")
            break
        elif is_board_full(board):
            print_board(board)
            print("It's a draw! The game is over.")
            break

        current_player = "X" if current_player == "O" else "O"


tic_tac_toe()
```