#Riya Jaison Kunnumkada
#TE Comps A
#9553

ARTIFICIAL INTELLIGENCE EXP 1.1

```
class TicTacToeBruteForce:
    WINNING_POSITIONS = [
        # Rows
        [0, 1, 2], [3, 4, 5], [6, 7, 8],
        # Columns
        [0, 3, 6], [1, 4, 7], [2, 5, 8],
        # Diagonals
        [0, 4, 8], [2, 4, 6]
    ]

    @staticmethod
    def get_winner(board):
        for positions in TicTacToeBruteForce.WINNING_POSITIONS:
            if board[positions[0]] == board[positions[1]] == board[positions[2]] != 0:
                return board[positions[0]]
        return 0

    @staticmethod
    def print_board(board):
        for i in range(3):
            for j in range(3):
                print(board[i * 3 + j] if board[i * 3 + j] != 0 else " ", end=" | ")
            print("\n---------")

    @staticmethod
    def is_board_full(board):
        return all(cell != 0 for cell in board)

    @staticmethod
    def play_game():
        board = [0] * 9
        player_turn = True

        while True:
            TicTacToeBruteForce.print_board(board)
            winner = TicTacToeBruteForce.get_winner(board)
            if winner:
                print("Player wins" if winner == 1 else "Computer wins")
                break
            if TicTacToeBruteForce.is_board_full(board):
                print("Tie game")
                break
```

```
            if player_turn:
                position = int(input("Player turn\nEnter a position (1-9): ")) - 1
                if board[position] == 0:
                    board[position] = 1
                    player_turn = False
            else:
                # Computer's turn
                position = TicTacToeBruteForce.get_computer_move(board)
                board[position] = 2
                player_turn = True

        TicTacToeBruteForce.print_board(board)
        print("Game ended")

    @staticmethod
    def get_computer_move(board):
        # Simple strategy: choose the first available empty position
        for i in range(9):
            if board[i] == 0:
                return i


TicTacToeBruteForce.play_game()
```

Output:

```
PS C:\Users\riyaj\AI> python bruteforce.py
  |   |   |
---------
  |   |   |
---------
  |   |   |
---------
Player turn
Enter a position (1-9): 1
1 |   |   |
---------
  |   |   |
---------
  |   |   |
---------
1 | 2 |   |
---------
  |   |   |
---------
  |   |   |
---------
Player turn
Enter a position (1-9): 5
1 | 2 |   |
---------
  | 1 |   |
---------
  |   |   |
---------
1 | 2 | 2 |
---------
  | 1 |   |
---------
  |   |   |
---------
Player turn
```

```
Enter a position (1-9): 9
1 | 2 | 2 |
---------
  | 1 |   |
---------
  |   | 1 |
---------
Player wins
1 | 2 | 2 |
---------
  | 1 |   |
---------
  |   | 1 |
---------
Game ended
```

```
PS C:\Users\riyaj\AI> python bruteforce.py
  |   |   |
---------
  |   |   |
---------
  |   |   |
---------
Player turn
Enter a position (1-9): 1
1 |   |   |
---------
  |   |   |
---------
  |   |   |
---------
1 | 2 |   |
---------
  |   |   |
---------
  |   |   |
---------
Player turn
Enter a position (1-9): 4
1 | 2 |   |
---------
1 |   |   |
---------
  |   |   |
---------
1 | 2 | 2 |
---------
1 |   |   |
---------
  |   |   |
---------
Player turn
Enter a position (1-9): 8
```

```
Enter a position (1-9): 8
1 | 2 | 2 |
---------
1 |   |   |
---------
  | 1 |   |
---------
1 | 2 | 2 |
---------
1 | 2 |   |
---------
  | 1 |   |
---------
Player turn
Enter a position (1-9): 6
1 | 2 | 2 |
---------
1 | 2 | 1 |
---------
  | 1 |   |
---------
1 | 2 | 2 |
---------
1 | 2 | 1 |
---------
2 | 1 |   |
---------
Computer wins
1 | 2 | 2 |
---------
1 | 2 | 1 |
---------
2 | 1 |   |
---------
Game ended
PS C:\Users\riyaj\AI>
```