Artificial Intelligence

Lab 6

Submitted by

Muhammad Bilal

FA21-BCS-110

Comsats University Islamabad,

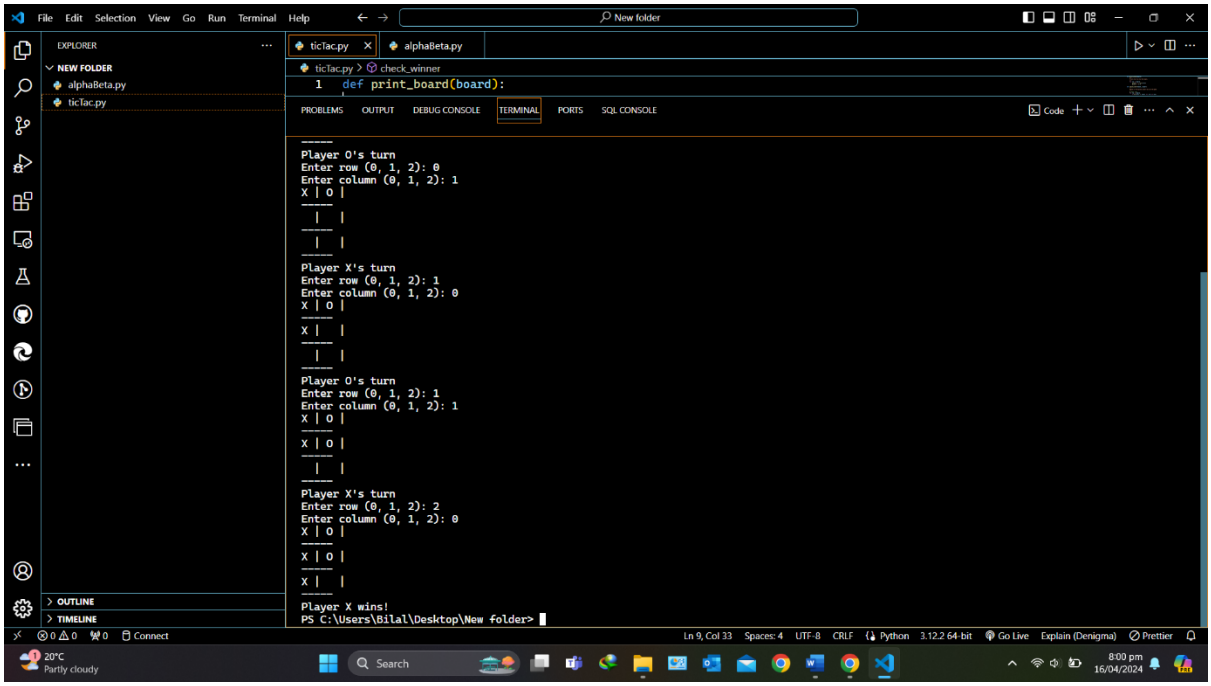
Abbottabad Campus.

Task 1 [10 points]

Write a Program to Implement Tic-Tac-Toe game using Python.

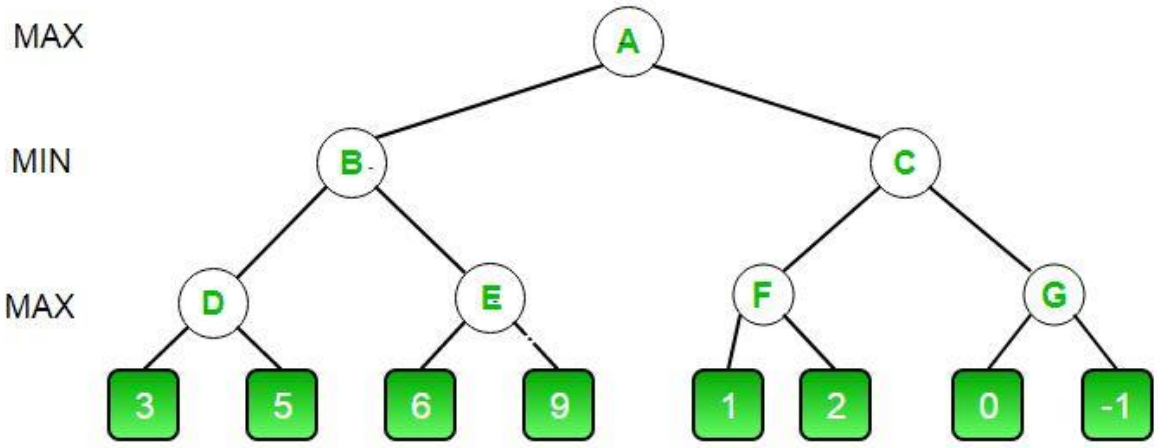
```
def print_board(board):
    """
    Prints the Tic-Tac-Toe board.
    """
    for row in board:
        print(" | ".join(row))
        print("-" * 5)

def check_winner(board, player):
    """
    Checks if the given player has won the game.
    """
    # Check rows
    for row in board:
        if all(cell == player for cell in row):
            return True

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

    # Check diagonals
    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):
    """
    Checks if the board is full (no empty cells).
    """
    for row in board:
        for cell in row:
            if cell == " ":
                return False
    return True

def main():
    """
```

```
    Main function to run the Tic-Tac-Toe game.
    """
    board = [[" " for _ in range(3)] for _ in range(3)]
    players = ["X", "O"]
    turn = 0

    while True:
        print_board(board)
        player = players[turn % 2]
        print(f"Player {player}'s turn")
        row = int(input("Enter row (0, 1, 2): "))
        col = int(input("Enter column (0, 1, 2): "))

        if board[row][col] == " ":
            board[row][col] = player
            if check_winner(board, player):
                print_board(board)
                print(f"Player {player} wins!")
                break
            if is_board_full(board):
                print_board(board)
                print("It's a tie!")
                break
            turn += 1
        else:
            print("That spot is already taken, try again.")

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



Task 2 [10 points]

Write a Program to Implement Alpha-Beta Pruning using Python.




```
def minimax(depth, nodeIndex, maximizingPlayer, values, alpha,
beta):
    if depth == 3:
        return values[nodeIndex]
```

```
    if maximizingPlayer:
        best = float("-inf")
        for i in range(2):
            val = minimax(depth + 1, nodeIndex * 2 + i, False,
values, alpha, beta)
            best = max(best, val)
            alpha = max(alpha, best)
            if beta <= alpha:
                break
        return best
    else:
        best = float("inf")
        for i in range(2):
            val = minimax(depth + 1, nodeIndex * 2 + i, True,
values, alpha, beta)
            best = min(best, val)
            beta = min(beta, best)
            if beta <= alpha:
                break
        return best

if __name__ == "__main__":
    values = [3, 5, 6, 9, 1, 2, 0, -1]
    print("The optimal value is:", minimax(0, 0, True, values,
float("-inf"), float("inf")))
```