Group Members

1. Abdur Rehman Kazmi (460951)
2. Muhammad Talha Kashif (454135)
3. Muhammad Owais (461359)

Introduction

In this report, we have made a tic tac toe game in which a human fights an AI and explained the code and its output.

Code

```
#include <iostream>

using namespace std;


char b[3][3]; char AI = 'X', HU = 'O';


bool check_win(char p) {

   for (int i = 0; i < 3; i++) if ((b[i][0] == p && b[i][1] == p && b[i][2] == p) || (b[0][i] == p && b[1][i] == p && b[2][i] == p)) return true;

   return (b[0][0] == p && b[1][1] == p && b[2][2] == p) || (b[0][2] == p && b[1][1] == p && b[2][0] == p);

}


bool check_draw() { for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) if (b[i][j] == ' ') return false; return true; }
```

```
void best_move(int &x, int &y) {
    for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) if (b[i][j] == ' ')
{ b[i][j] = AI; if (check_win(AI)) { x = i; y = j; b[i][j] = ' '; return; }
b[i][j] = ' '; }
    for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) if (b[i][j] == ' ')
{ b[i][j] = HU; if (check_win(HU)) { x = i; y = j; b[i][j] = ' ';
return; } b[i][j] = ' '; }
    if (b[1][1] == ' ') { x = 1; y = 1; return; }
    int corners[4][2] = {{0, 0}, {0, 2}, {2, 0}, {2, 2}};
    for (int i = 0; i < 4; i++) {
        int cx = corners[i][0], cy = corners[i][1];
        if (b[cx][cy] == ' ') { x = cx; y = cy; return; }
    }
    for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) if (b[i][j] == ' ')
{ x = i; y = j; return; }
}

int main() {
    for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) b[i][j] = ' ';
    while (true) {
        for (int i = 0; i < 3; i++, cout << endl) for (int j = 0; j < 3;
j++) cout << b[i][j] << ' ';
```

```
        cout << "Enter your move (row 1-3, column 1-3): "; int x,
y;
        if (!(cin >> x >> y)) {
            cout << "Invalid input! Please enter numbers only.\n";
            cin.clear(); cin.ignore(10000, '\n'); continue;
        }
        x--; y--;
        if (x >= 0 && x < 3 && y >= 0 && y < 3 && b[x][y] == ' ')
b[x][y] = HU;
        else { cout << "Invalid move! Enter row and column
numbers between 1 and 3.\n"; continue; }
        if (check_win(HU)) { cout << "You win!\n"; break; } if
(check_draw()) { cout << "Draw!\n"; break; }
        best_move(x, y); b[x][y] = AI; cout << "AI placed at row "
<< x+1 << " column " << y+1 << endl;
        if (check_win(AI)) { cout << "AI wins!\n"; break; } if
(check_draw()) { cout << "Draw!\n"; break; }
    }
    return 0;
}
```

Explanation of Code

The code has been broken down into functions, these are as follows:

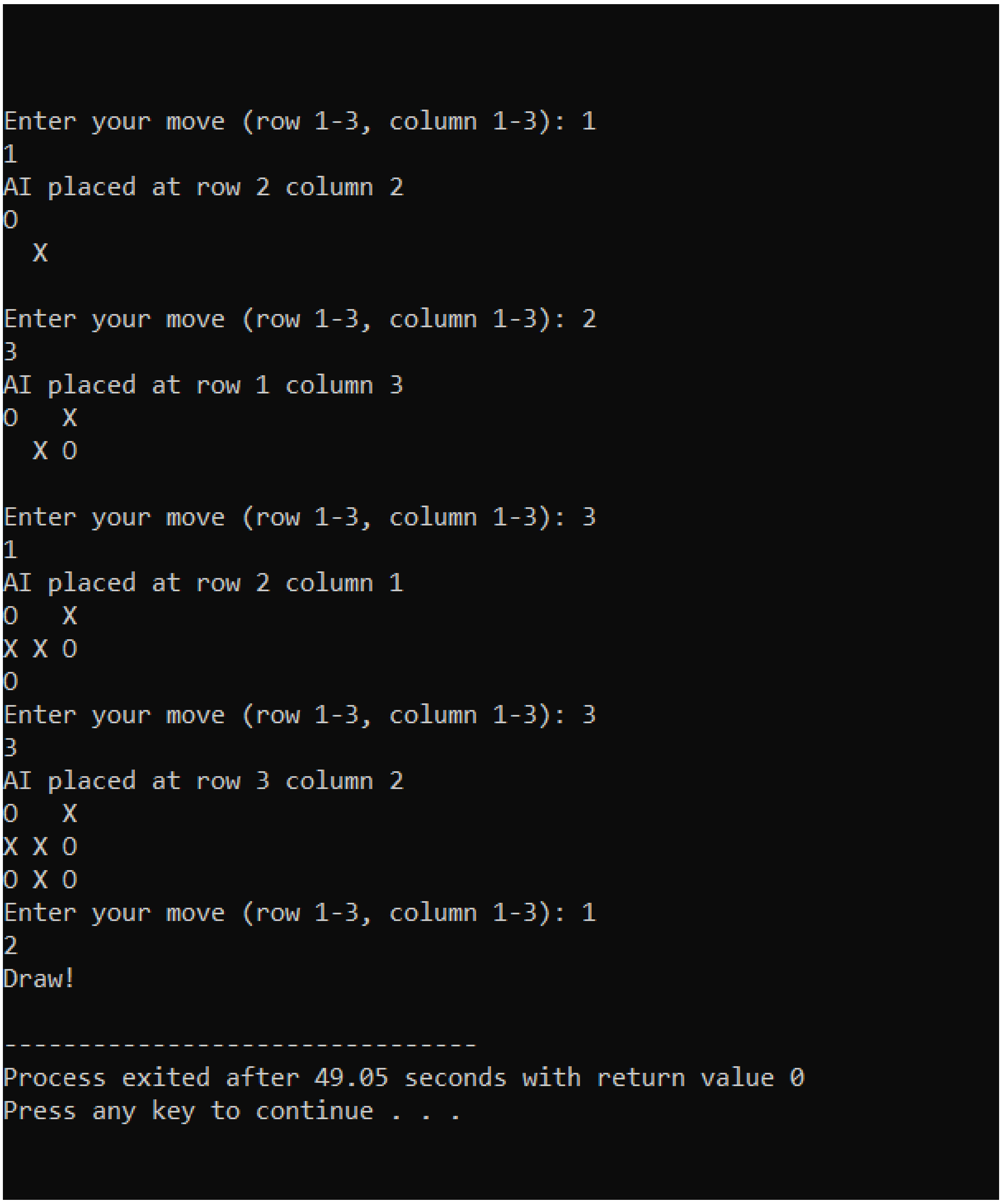
1. **check_win:** check_win function checks who won by going through the board and checking if there is a line (vertical, horizontal or diagonal) of the same character (X or O) and returns true if the line has the same character.

2. **check_draw:** check_draw function checks the draw by going through the board and checking if there are empty spaces in the board. If there is no win and no empty positions in the board, then it returns true.

3. **best_move:** This function finds the best move for the AI through the following way:
   - Checks if there is any move that wins for the AI
   - If there is no winning move for the AI, then it blocks any winning move for the human.
   - If there is no winning move for the human, then it prioritizes the center.
   - If move at center is not possible, then it tfirst available corner.
   - If move at corners is not possible then it makes a move at first available space.

4. **Main function:** In the main function, all the functions are called, all the processes occurring in the main function are as follows:
    - Creates a board to play on and prompts the user to enter a move (if user enters an incorrect value then it asks the user to re-enter a move).
    - After the move of the player is made, it makes the move of the AI.
    - Shows the current position of the game and asks the user to enter a move again.
    - After every move, it checks for a draw or a win.
    - Repeats step 2,3 and 4 till the game ends (someone won or a draw has been reached) and exits the loop.

Output



```
Enter your move (row 1-3, column 1-3): 1
1
AI placed at row 2 column 2
O
  X

Enter your move (row 1-3, column 1-3): 2
3
AI placed at row 1 column 3
O   X
  X O

Enter your move (row 1-3, column 1-3): 3
1
AI placed at row 2 column 1
O   X
X X O
O
Enter your move (row 1-3, column 1-3): 3
3
AI placed at row 3 column 2
O   X
X X O
O X O
Enter your move (row 1-3, column 1-3): 1
2
Draw!

--------------------------------
Process exited after 49.05 seconds with return value 0
Press any key to continue . . .
```

Explanation of Output

The output can be explained in the following bullets.

- Draws the board.
- Asks the user to enter a move.
- If the user enters a wrong move, then it shows an error and asks the user to re-enter a move.
- Displays the board after every move.
- If a draw has been reached, then it displays “Draw!”.
- If a win has been reached, then it shows who won.

Conclusion

This code provides a game where a player battles an AI in a game of tic tac toe.

Future Works and Applications

This group project has allowed us to improve our coding skills along with team cooperation.