School of Mechanical & Manufacturing Engineering (SMME),

National University of Science and Technology (NUST),

Sector H-12, Islamabad

Program: BE-Aerospace Section: AE-01

Session: Fall 2023 Semester: 1st

Course Title: Fundamentals of Programming (CS-109)

PROJECT

"Fundamentals of Programming"

NAME	CMS
IKHLAS JAMSHAID	***454122***
MUHAMMAD KASHF RAZA	***466317***

Tasks: Make a Tic Tac Toe game

Abstract

This C++ program implements a simple console-based Tic-Tac-Toe game for two players. The game utilizes a 3x3 matrix to represent the board, and players take turns placing their respective markers ('X' or 'O') in an attempt to win by forming a horizontal, vertical, or diagonal line. The program includes functions for displaying the board, handling player moves, determining a winner, and managing the overall game flow.

Approach:

1. Initialization

1.1 Matrix Initialization

- **matrix** array is initialized as a 3x3 grid to represent the Tic-Tac-Toe board.
- The initial values '1' through '9' serve as placeholders for player moves.

1.2 Variable Declaration

- **row** and **coloumn** variables are declared to track the current player's move.
- **token** is initialized to 'X' as the starting marker.
- **tie** is set to false, indicating the game hasn't ended in a tie.
- **n1** and **n2** store the names of the two players.

2. Player Input

2.1 Name Entry

- Players are prompted to enter their names using **getline**.
- Player 1's name is stored in **n1**, and Player 2's name is stored in **n2**.

2.2 Player Order Display

- A message is displayed informing players of the order in which they will take turns.

3. Game Loop

3.1 Continuous Execution

- The program enters a perpetual loop using **while (true)**.
- Inside the loop, the **board** function is called to display the current state of the board.
- Subsequently, the **placement** function is invoked to handle the current player's move.

3.2 Exit Condition

- The loop continues until the **win** function detects a winner or a tie, triggering a **break** statement to exit the loop.

4. Display Board

4.1 Visualization

- The **board** function visually presents the current state of the Tic-Tac-Toe board.
- **cout** statements are utilized to create a grid layout with matrix values.

5. Player Move

5.1 User Prompt

- The **placement** function prompts the current player (determined by **token**) to enter a position on the board.

5.2 Input Validation

- Input validation ensures the entered position is valid (1 to 9) and not already occupied.

5.3 Update Variables

- The **row** and **coloumn** variables are updated based on the player's input.

6. Win Condition

6.1 Check Rows and Columns

- The **win** function examines rows and columns for matching markers ('X' or 'O').

6.2 Check Diagonals

- Diagonals are checked to determine if a player has won.

6.3 Tie Check

- If the board is full and no winner is found, the game is declared a tie.

7. Game Outcome

7.1 Winner Determination

- After the game loop exits, the program determines the outcome based on the value of **token**.

7.2 Display Result

- Victory messages are displayed for the winning player, or a draw message is shown if the game ended in a tie.

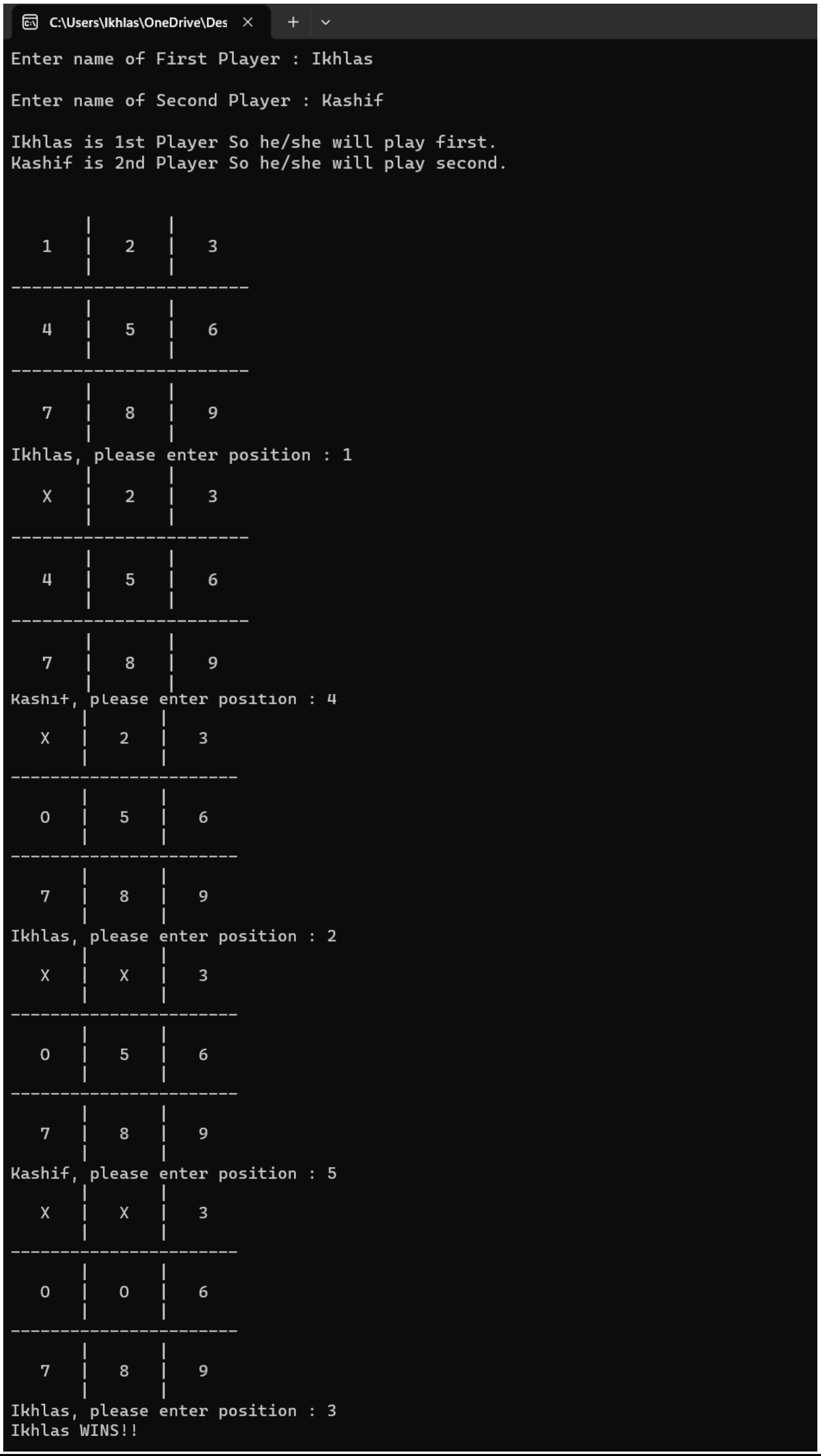
7.3 Program Termination

- The program returns 0, indicating successful execution.

THE CODE:

```
#include<iostream>
using namespace std;

char matrix[3][3] = {{'1','2','3'},{'4','5','6'},{'7','8','9'}};
int row;
int coloumn;
char token = 'X';
bool tie = false;
string n1;
string n2;

void board(void) {
    cout << "     |     |     " << endl;
    cout << "  " << matrix[0][0] << "  |" << "  " << matrix[0][1] << "  |" << "  " << matrix[0][2]
<< endl;
    cout << "     |     |     " << endl;
    cout << "-----------------------" << endl;
    cout << "     |     |     " << endl;
    cout << "  " << matrix[1][0] << "  |" << "  " << matrix[1][1] << "  |" << "  " << matrix[1][2]
<< endl;
    cout << "     |     |     " << endl;
    cout << "-----------------------" << endl;
    cout << "     |     |     " << endl;
    cout << "  " << matrix[2][0] << "  |" << "  " << matrix[2][1] << "  |" << "  " << matrix[2][2]
<< endl;
    cout << "     |     |     " << endl;
}

void placement(void) {
    int digit;
    cout << (token == 'X' ? n1 : n2) << ", please enter position : ";
    cin >> digit;
    cin.ignore();  // Consume the newline character

    if (digit == 1) {
        row = 0;
        coloumn = 0;
    } else if (digit == 2) {
        row = 0;
        coloumn = 1;
    } else if (digit == 3) {
        row = 0;
        coloumn = 2;
    } else if (digit == 4) {
        row = 1;
        coloumn = 0;
    } else if (digit == 5) {
        row = 1;
        coloumn = 1;
    } else if (digit == 6) {
        row = 1;
        coloumn = 2;
    } else if (digit == 7) {
        row = 2;
        coloumn = 0;
    } else if (digit == 8) {
        row = 2;
        coloumn = 1;
    } else if (digit == 9) {
        row = 2;
        coloumn = 2;
    } else {
        cout << "INVALID POSITION " << endl;
        return;
    }

    if (matrix[row][coloumn] != 'X' && matrix[row][coloumn] != 'O') {
        matrix[row][coloumn] = token;
        token = (token == 'X' ? 'O' : 'X');
    } else {
        cout << "Position already occupied. Choose another position." << endl;
        placement();
    }
}

bool win(void) {
    for (int i = 0; i < 3; i++) {
        if ((matrix[i][0] == matrix[i][1] && matrix[i][1] == matrix[i][2]) || (matrix[0][i] ==
matrix[1][i] && matrix[1][i] == matrix[2][i])) {
            return true;
        }
    }

    if ((matrix[0][0] == matrix[1][1] && matrix[1][1] == matrix[2][2]) || (matrix[0][2] ==
matrix[1][1] && matrix[2][0] == matrix[1][1])) {
        return true;
    }

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            if (matrix[i][j] != 'X' && matrix[i][j] != 'O') {
                return false;
            }
        }
    }

    tie = true;
    return true;
}

int main() {
    cout << "Enter name of First Player : ";
    getline(cin, n1);
    cout << endl;

    cout << "Enter name of Second Player : ";
    getline(cin, n2);
    cout << endl;

    cout << n1 << " is 1st Player So he/she will play first. " << endl;
    cout << n2 << " is 2nd Player So he/she will play second. " << endl << endl << endl;

    while (true) {
        board();
        placement();
        if (win()) {
            break;
        }
    }

    if (token == 'X' && !tie) {
        cout << n2 << " WINS!!" << endl;
    } else if (token == 'O' && !tie) {
        cout << n1 << " WINS!!" << endl;
    } else {
        cout << "IT IS A DRAW " << endl;
    }

    return 0;
}
```

THE OUTPUT



C:\Users\Ikhlas\OneDrive\Des

```
Enter name of First Player : Ikhlas

Enter name of Second Player : Kashif

Ikhlas is 1st Player So he/she will play first.
Kashif is 2nd Player So he/she will play second.


       |       |
   1   |   2   |   3
       |       |
-----------------------
       |       |
   4   |   5   |   6
       |       |
-----------------------
       |       |
   7   |   8   |   9
       |       |
Ikhlas, please enter position : 1
       |       |
   X   |   2   |   3
       |       |
-----------------------
       |       |
   4   |   5   |   6
       |       |
-----------------------
       |       |
   7   |   8   |   9
       |       |
Kashif, please enter position : 4
       |       |
   X   |   2   |   3
       |       |
-----------------------
       |       |
   O   |   5   |   6
       |       |
-----------------------
       |       |
   7   |   8   |   9
       |       |
Ikhlas, please enter position : 2
       |       |
   X   |   X   |   3
       |       |
-----------------------
       |       |
   O   |   5   |   6
       |       |
-----------------------
       |       |
   7   |   8   |   9
       |       |
Kashif, please enter position : 5
       |       |
   X   |   X   |   3
       |       |
-----------------------
       |       |
   O   |   O   |   6
       |       |
-----------------------
       |       |
   7   |   8   |   9
       |       |
Ikhlas, please enter position : 3
Ikhlas WINS!!
```

CONCLUSION:

In summary, the C++ Tic-Tac-Toe program employs a systematic and modular approach. It initializes the game, captures user input, and executes a well-organized game loop. Logical win conditions and clear outcome messages contribute to its educational value for learning C++ and game development concepts. The program stands as a concise and effective tool for both understanding programming principles and creating an interactive gaming experience.

References:

https://www.youtube.com/watch?v=dv_75WfQ1rA&t=129s&ab_channel=Simplilearn