Name Muhammad Hammad.

Class Bs-IT 1(A).

Enrollment No: 01-135211-102.

Code:

```
#include<iostream>    //header file.

#include<iomanip>    //header file for manipulator.


using namespace std;

char board[3][3] = { { '1', '2', '3' }, { '4', '5', '6' }, { '7', '8', '9' } }; // storing digits as character. //using double array for storing data in multiple rows or columns.

char turn = 'X'; //global declaration of variable.

int row, column; //global declaration of row and columns.

int choice;            // global declaration of type integer.

bool draw = false; // using type bool with variable name and its value.

void display_board()      //using function.

{

        system("cls");    //used for screen clearing.

        cout << "\tTic Tac toe game " << endl;                                                    // Game name;

        cout << "\t    Player 1[X] \n \t    Player 2[0]" << endl << endl;                    // players name player [x] & player [y].

        cout << "\t\t          |          |          " << endl;                                    // Basic display of game.

        cout << "\t\t    " << board[0][0] << "     |      " << board[0][1] << " | " << board[0][2] << " \n ";

        cout << "\t\t_____|_____|_____" << endl;

        cout << "\t\t          |          |          " << endl;

        cout << "\t\t    " << board[1][0] << "     |      " << board[1][1] << " | " << board[1][2] << " \n ";

        cout << "\t\t_____|_____|_____" << endl;

        cout << "\t\t          |          |          " << endl;

        cout << "\t\t    " << board[2][0] << "     |      " << board[2][1] << " | " << board[2][2] << " \n ";
```

```
	cout << "\t\t	|	|	" << endl << endl;


}
void player_turn()//function declaration.
{
	int choice;


	if (turn == 'X')
	{
		cout << "\n\tPlayer1 [X] turn: ";	// player 1[x] turn.
      }
	if (turn == 'O')
	{
		cout << "\n\tplayer2 [O] turn: ";
	}
	cin >> choice;
	switch (choice)
	{


	case 1:
		row = 0, column = 0;		// value of players will be update at at that position of row and column.
		break;
	case 2:
		row = 0, column = 1;		// value of players will be update at at that position of row and column.
		break;
	case 3:
		row = 0, column = 2;		// value of players will be update at at that position of row and column.
```

```
                break;
        case 4:
                row = 1, column = 0;      // value of players will be update at at that position of row and
column.
                break;
        case 5:
                row = 1, column = 1;    // value of players will be update at at that position of row and
column.
                break;
        case 6:
                row = 1, column = 2;    // value of players will be update at at that position of row and
column.
                break;
        case 7:
                row = 2, column = 0;    // value of players will be update at at that position of row and
column.
                break;
        case 8:
                row = 2, column = 1;    // value of players will be update at at that position of row and
column.
                break;
        case 9:
                row = 2, column = 2;    // value of players will be update at at that position of row and
column.
                break;
        default:
                cout << "\t\t Invalid output " << endl;    //using default for invalid input.
        }
        if (turn == 'X' && board[row][column] != 'X' && board[row][column] != 'O')    //using a condition
to not to overwrite on dashboard of game.
        {
```

```
                board[row][column] = 'X';          // players turn than other player turn.
                turn = 'O';


        }
        else if (turn == 'O' && board[row][column] != 'X' && board[row][column] != 'O')
        {
                board[row][column] = 'O';    // players turn than other player turn.
                turn = 'X';


        }
        else
        {
                cout << "\t\t Box is already filled!!\n \t\t Please try again\n\n";      // used for
overwriting on a place of box.
                player_turn();                                                                  //
Again calling a function.


        }
        display_board();



}
bool gameover()
{
        //check win.
        for (int i = 0; i < 3; i++)
                if (board[i][0] == board[i][1] && board[i][0] == board[i][2] || board[0][i] == board[1][i]
&& board[0][i] == board[2][i])        //comparing variables x or o with row and column position.
                        return false;                                   //for winning a game we return false.
```

```
	if (board[0][0] == board[1][1] && board[0][0] == board[2][2] || board[0][2] == board[1][1] &&
board[0][2] == board[2][0]) // Or checking column wise for a player to win while using a condtion.
		return false;					// for winning a game we return false.



	for (int i = 0; i < 3; i++)				//if there is any box is not filled.
	{
		for (int j = 0; j < 3; j++)
		{
			if (board[i][j] != 'X' && board[i][j] != 'O')
			{
				return true;


			}


		}


	}
	draw = true;
	return false;


}


int main()
{
	while (gameover())	//using while loop.
	{


		display_board();	//calling function from main body.
```

```
            player_turn();        //calling function from main body.


            gameover();           //calling again gameover function.


        }
        if (turn == 'X' && draw == false)                    //if turn==x means player [O] wins the
game.
        {
            cout << "\tPlayer2 [O] wins!! Congratulations\n";
        }
        else if (turn == 'O' && draw == false)               //if turn==O means player [X] wins the game.
        {
            cout << "\tPlayer1 [X] wins!! Congratulations\n";
        }
        else
        {
            cout << "\tGame Draw!!" << endl;
        }
        system("pause");
}
```


Tic Tac toe game
Player 1[X]
Player 2[0]
X | X | X
O | 5 | O
7 | 8 | 9
Player1 [X] wins!! Congratulations