OOPS PROJECT REPORT

TIC-TAC-TOE GAME

SUBMITTED TO

DR. NONITA SHARMA

MADE BY

BHART BANSAL (20103039)

AYUSH KUMAR JHA (20103035)

ASHANPREET SINGH SRA (20103033)

SECTION-A GROUP-2

TABLE OF CONTENTS

INTRODUCTION

Tic-tac-toe game also known as noughts and crosses, or **X's and O's** is a paper and pencil game for two players who take turns marking the spaces in a three-by-three grid with *X* or *O*. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner. It is a solved game, with a forced draw assuming best play from both players.

OBJECTIVE

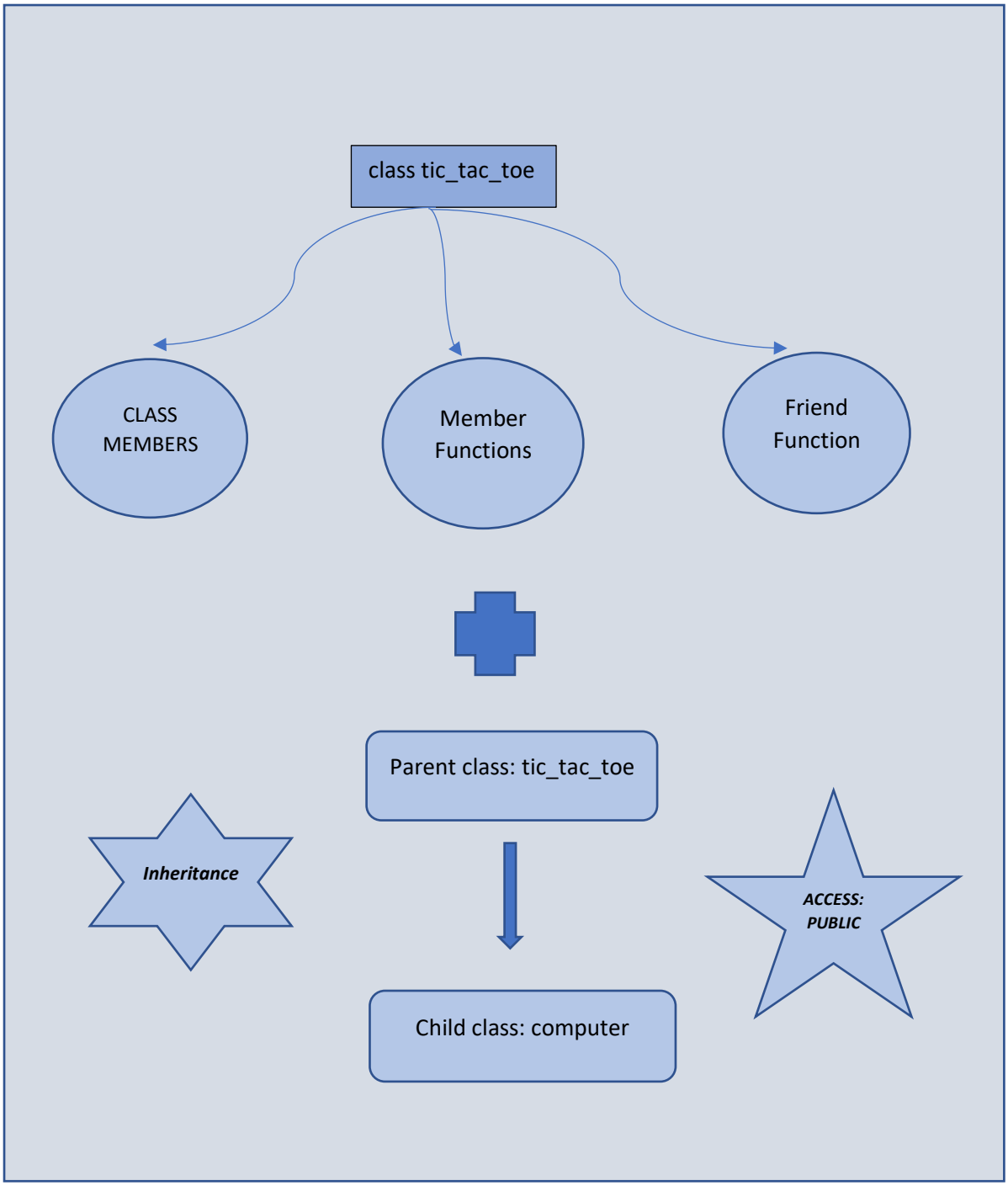
Create a project tic tac toe that will enable you to write a complete program to play the game of tic tac toe. The class contains as

- private data a 3 by 3 2 -array of integers.
- The constructor should initialize the empty board to all zeros.

Allow two human players. whenever the first player moves, place a 1 in the specified square. Please 2 whenever the second player moves. Each move must be to an empty square. After each move determine whether the game has been won or is a draw.

1. Modify the program so that the computer makes the move for one of the players. (rand ())
2. Also allow the player to specify whether he or she wants to go first or second.
3. Modify the three-dimensional tic tac toe on a 4 by 4 board

CLASS DIAGRAM



CLASS MEMBERS

PRIVATE MEMBERS:

String name1,name2(To get the name of players)

PROTECTED MEMBERS:

Char *arr0, int *arr1,int *arr2

PUBLIC MEMBERS:

Int pos(To get position where symbol has to be placed)

Int count(counter counts number of turns took place)

Char sym1, char sym2(To give symbol to player)

MEMBER FUNCTIONS

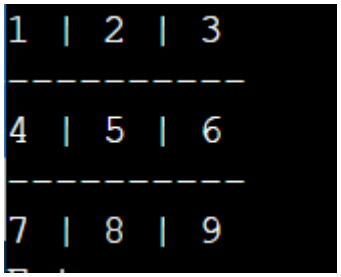
1) bool call ()

2) friend bool winner (int n, int *arr)

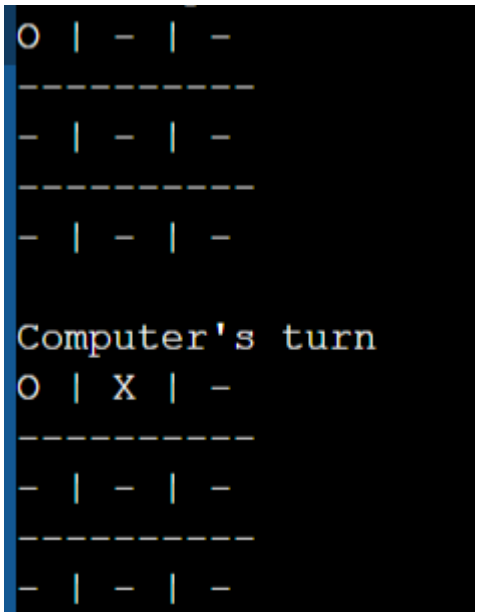
3) void display ()

4) void Initial_display ()

->**void Initial_display ()**------It is function to display initial setup/view of tic tac toe game showing position 1 to 9 where symbol can be placed.



->**void display()**--------It is function to display setup/view of tic tac toe game showing symbol (X or O) placed after each turn of player.



->**bool call()**-----First call function check whether previously symbol is placed or not. If it is placed earlier it prints “Invalid position” else it places symbol at required position and make change in respective array of player(arr1 or arr2).

*Then it calls display function to display current scenario of game after player move

*After that it also call winner function to check whether any winner condition is achieved or not.

*It also maintain counter count to check number of turns took place.

->**bool winner**(int n, int *arr)------It takes int n (pos-1) and int * arr as argument and check winner condition is achieved or not after each move of every player

1)Int p=n%3 . It gives (column-1) in which current symbol is placed and check with other 2 element in same row whether they are been placed with same symbol or not. If yes it return true .

*if (p=0) column=1 we need to check elements in column 2 and column 3 have been place with same symbol as that in column 1 of particular same row. So condition arr[n+2] and arr[n+1] are equal or not are checked.

Eg. Player 1 chooses Pos=1

n=0;

p=0;

so to whether winner condition is achieved or not in row 1 it need to check position 1,2,3 are placed with same symbol or not so it check by if condition given in code

******In figure if marked places would have been placed with O winner condition would have been achieved**********

*if (p=1) column=2 we need to check elements in column 1 and column 3 have been place with same symbol as that in column 2 of particular same row. So condition arr[n-1] and arr[n+1] are equal or not are checked.

*if (p=2) column=3 we need to check elements in column 2 and column 1 have been place with same symbol as that in column 3 of particular same row. So condition arr[n-2] and arr[n-1] are equal or not are checked.

2)Int p1=n/3 It gives (row-1) in which current symbol is placed and check with other 2 element in same column whether they are been placed with same symbol or not. If yes it return true .

*if (p1=0) row=1 we need to check elements in row 2 and row 3 have been place with same symbol as that in row 1 of particular same column. So condition arr[n+3] and arr[n+6] are equal or not are checked.

Eg. If player give pos=1;

N=0;p1=0;

Winner possibility for player is when it has all elements same in column 1 so for that at position 1 , 4,7 same symbol has to be there so for that arr[n+3] ,arr[n] ,arr[n+6] need to be equal.

*if (p1=1) row=2 we need to check elements in row 1 and row 3 have been place with same symbol as that in row 1 of particular same column. So condition arr[n+3] and arr[n-3] are equal or not are checked.

*if (p1=0) row=3 we need to check elements in row 2 and row 1 have been place with same symbol as that in row 3 of particular same column. So condition arr[n-3] and arr[n-6] are equal or not are checked.

For diagonal we use both p and p1 .

First we check p1----If p1=0 it means row 1 if p=0 it means column 1 if p=2 it means column 3. If (p1=0 &p=0) it means position 1, for diagonal condition to satisfy to win we need to check whether at position 5 and 9 also same symbol is there or not.So accordingly if condition has been set up

*If (p1=0 &p=2) it means position 3,for diagonal condition to satisfy to win we need to check whether at position 5 and 7 also same symbol is there or not. So accordingly if condition has been set up

All other condition has been set accordingly such that to win either one complete row or column or diagonal has been placed with same symbol.

Other OOPS CONCEPT USED

1)Constructor

A constructor is a special member function of a class which is automatically invoked at the time of creation of an object to initialize or construct the values of data members of the object.
The name of the constructor is the same as that of the class to which it belongs.

In this program, constructor is used to initialize all 3 arrays and print message of Game begin

Code:

```
tic_tac_toe()
   {

       arr0 = new char[9];
       arr1 = new int[9];
       arr2 = new int[9];
       for (int i = 0; i < 9; i++)
       {
           arr0[i] = ' ';
           arr1[i] = 0;
           arr2[i] = 0;
       }
       cout << "Game Begin" << endl;
   }
```

2)Destructor

- Like a constructor, a destructor is also a member function that is automatically invoked.
- However, unlike the constructor which constructs the object, the job of destructor is to destroy the object.
- For this, it de-allocates the memory dynamically allocated to the variable(s) or perform other clean up operations.

Code

```
~tic_tac_toe()
   {
       cout << "Game ended" << endl;
   }
```

3)**Friend Function**

A friend function of a class is a non-member function of the class that can access its private and protected members. Friend function is a normal external function that is given special access privileges.

Code

```
friend bool winner(int n, int *arr);
```

```
bool winner(int n, int *arr)
{
    int p = n % 3;
    switch (p)
    {
    case 0:
        if (arr[n + 2] && arr[n + 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 1:
        if (arr[n - 1] && arr[n + 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 2:
        if (arr[n - 2] && arr[n - 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    }
    int p1 = n / 3;
    switch (p1)
    {
    case 0:
```

```
        if (arr[n + 3] && arr[n + 6])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 1:
        if (arr[n - 3] && arr[n + 3])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 2:
        if (arr[n - 3] && arr[n - 6])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    }
    switch (p1)
    {
    case 0:
        if (p == 0)
        {
            if (arr[n + 4] && arr[n + 8])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        if (p == 2)
        {
            if (arr[n + 2] && arr[n + 4])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    case 1:
        if (p == 1)
        {
            if (arr[n - 4] && arr[n + 4])
            {
                cout << "Winnner" << endl;
                return true;
```

```
            }
            if (arr[n - 2] && arr[n + 2])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    case 2:
        if (p == 0)
        {
            if (arr[n - 4] && arr[n - 2])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        if (p == 2)
        {
            if (arr[n - 4] && arr[n - 8])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    }
    return false;
}
```

4)**Operator Overloading (>>)**

In this program overloaded >> to improve representation of code.

The insertion and extraction of operatorsare overloaded by using a friend function because we need to call these functions without any object.The insertion and extraction operators must return the value of the left operand.

Code:

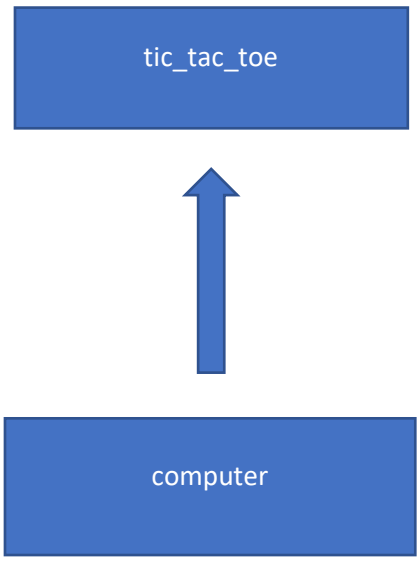
```
friend void operator>>(istream &input, tic_tac_toe &s)
    {
        cout << "Enter name player 1: ";
        input >> s.name1;
        cout << "Enter name player 2: ";
        input >> s.name2;
        cout << "Choose your(" << s.name1 << ") symbol(O, X): ";
        cin >> s.sym1;
```

```
        s.sym2 = (s.sym1 == 'X') ? 'O' : 'X';
    }
```

5)**Inheritance**

The capability of a class to derive properties and characteristics from another class is called **Inheritance**. Inheritance is one of the most important features of Object-Oriented Programming.
Sub Class: The class that inherits properties from another class is called Sub class or Derived Class.
Super Class: The class whose properties are inherited by sub class is called Base Class or Super class.

In this program class computer has been inherited from class **tic_tac_toe** as in computer class we need to add some additional features. In class **computer** turn() function is made to ask whether player or computer makes first turn.



```
class computer : public tic_tac_toe
{
   bool flag;
   string name;
public:
   friend void operator>>(istream &input, computer &s)
   {
      cout << "Enter your name: ";
      input >> s.name;
      cout << "Choose your(" << s.name << ") symbol(O, X): ";
      cin >> s.sym1;
      s.sym2 = (s.sym1 == 'X') ? 'O' : 'X';
   }
   void turn()
   {
      int t;
      cout << "Enter 1 to specify whether you want to go first else 2: ";
      cin >> t;
      if (t == 1)
         flag = true;
      else
         flag = false;
   }
   bool call();
};
```

**********Use of rand function**********

rand () function is an inbuilt function in C++ STL, which is defined in <cstdlib> header file. rand () is used to generate a series of random numbers. We use this function when we want to generate a random number in our code.

Like we are making a game of tic tac toe in C++ and we needed to generate any random number between 1 and 9 to fill position of computer turn.

Source Code

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

class tic_tac_toe
{
private:
    string name1, name2;

protected:
    char *arr0;
    int *arr1;
    int *arr2;

public:
    int pos, count = 0;
    char sym1, sym2;
    tic_tac_toe()
    {

        arr0 = new char[9];
        arr1 = new int[9];
        arr2 = new int[9];
        for (int i = 0; i < 9; i++)
        {
            arr0[i] = ' ';
            arr1[i] = 0;
            arr2[i] = 0;
        }
        cout << "Game Begin" << endl;
    }
    friend void operator>>(istream &input, tic_tac_toe &s)
    {
        cout << "Enter name player 1: ";
        input >> s.name1;
        cout << "Enter name player 2: ";
        input >> s.name2;
        cout << "Choose your(" << s.name1 << ") symbol(O, X): ";
        cin >> s.sym1;
        s.sym2 = (s.sym1 == 'X') ? 'O' : 'X';
    }
```

```
    bool call();
    friend bool winner(int n, int *arr);
    void display();
    void Initial_display();
    ~tic_tac_toe()
    {
        cout << "Game ended" << endl;
    }
};

class computer : public tic_tac_toe
{
    bool flag;
    string name;

public:
    friend void operator>>(istream &input, computer &s)
    {
        cout << "Enter your name: ";
        input >> s.name;
        cout << "Choose your(" << s.name << ") symbol(O, X): ";
        cin >> s.sym1;
        s.sym2 = (s.sym1 == 'X') ? 'O' : 'X';
    }
    void turn()
    {
        int t;
        cout << "Enter 1 to specify whether you want to go first else 2: ";
        cin >> t;
        if (t == 1)
            flag = true;
        else
            flag = false;
    }
    bool call();
};

bool winner(int n, int *arr);
bool computer ::call()
{
    if (flag)
    {
    to1:
        cout << "Choose pos " << name << " : ";
        cin >> pos;
        if (arr0[pos - 1] == ' ')
        {
            if (pos >= 1 && pos <= 9)
```

```
        {
            arr0[pos - 1] = sym1;
            arr1[pos - 1]++;
            display();
            if (winner(pos - 1, arr1))
            {
                cout << name << endl;
                return true;
            }
            count++;
            if (count < 9)
            {
            to:
                srand(time(NULL));
                pos = rand() % 9 + 1;
                while (arr0[pos - 1] != ' ')
                {
                    srand(time(NULL));
                    pos = rand() % 9 + 1;
                }
                if (arr0[pos - 1] == ' ')
                {
                    arr0[pos - 1] = sym2;
                    arr2[pos - 1]++;
                    cout << endl
                         << "Computer's turn" << endl;
                    display();
                    if (winner(pos - 1, arr2))
                    {
                        cout << "Computer" << endl;
                        return true;
                    }
                    count++;
                }
                else
                {
                    cout << "Invalid position" << endl;
                    goto to;
                }
            }
        }
    }
    else
    {
        cout << "Invalid position" << endl;
        goto to1;
    }
    return false;
```

```
    }
    else
    {
        if (count < 9)
        {
            srand(time(NULL));
            pos = rand() % 9 + 1;
            while (arr0[pos - 1] != ' ')
            {
                srand(time(NULL));
                pos = rand() % 9 + 1;
            }
            arr0[pos - 1] = sym2;
            arr2[pos - 1]++;
            cout << endl
                 << "Computer's turn" << endl;
            display();
            if (winner(pos - 1, arr2))
            {
                cout << "Computer" << endl;
                return true;
            }
            count++;
            if (count < 9)
            {
            to3:
            to4:
                cout << "Choose pos " << name << " : ";
                cin >> pos;
                if (arr0[pos - 1] == ' ')
                {
                    if (pos >= 1 && pos <= 9)
                    {
                        arr0[pos - 1] = sym1;
                        arr1[pos - 1]++;
                        display();
                        if (winner(pos - 1, arr1))
                        {
                            cout << name << endl;
                            return true;
                        }
                        count++;
                    }
                    else
                    {
                        cout << "Invalid position" << endl;
                        goto to3;
                    }
```

```
                }
                else
                {
                    cout << "Invalid position" << endl;
                    goto to4;
                }
            }
        }
        return false;
    }
    return false;
}

bool tic_tac_toe::call()
{
to1:
    cout << "Choose pos " << name1 << " : ";
    cin >> pos;
    if (arr0[pos - 1] == ' ')
    {
        if (pos >= 1 && pos <= 9)
        {
            arr0[pos - 1] = sym1;
            arr1[pos - 1]++;
            display();
            if (winner(pos - 1, arr1))
            {
                cout << name1 << endl;
                return true;
            }
            count++;
            if (count < 9)
            {
            to:
                cout << "Choose pos " << name2 << " : ";
                cin >> pos;
                if (arr0[pos - 1] == ' ')
                {
                    arr0[pos - 1] = sym2;
                    arr2[pos - 1]++;
                    display();
                    if (winner(pos - 1, arr2))
                    {
                        cout << name2 << endl;
                        return true;
                    }
                    count++;
                }
```

```
                else
                {
                    cout << "Invalid position" << endl;
                    goto to;
                }
            }
        }
    }
    else
    {
        cout << "Invalid position" << endl;
        goto to1;
    }
    return false;
}

bool winner(int n, int *arr)
{
    int p = n % 3;
    switch (p)
    {
    case 0:
        if (arr[n + 2] && arr[n + 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 1:
        if (arr[n - 1] && arr[n + 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 2:
        if (arr[n - 2] && arr[n - 1])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    }
    int p1 = n / 3;
    switch (p1)
    {
    case 0:
        if (arr[n + 3] && arr[n + 6])
```

```
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 1:
        if (arr[n - 3] && arr[n + 3])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    case 2:
        if (arr[n - 3] && arr[n - 6])
        {
            cout << "Winner" << endl;
            return true;
        }
        break;
    }
    switch (p1)
    {
    case 0:
        if (p == 0)
        {
            if (arr[n + 4] && arr[n + 8])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        if (p == 2)
        {
            if (arr[n + 2] && arr[n + 4])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    case 1:
        if (p == 1)
        {
            if (arr[n - 4] && arr[n + 4])
            {
                cout << "Winnner" << endl;
                return true;
            }
```

```
            if (arr[n - 2] && arr[n + 2])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    case 2:
        if (p == 0)
        {
            if (arr[n - 4] && arr[n - 2])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        if (p == 2)
        {
            if (arr[n - 4] && arr[n - 8])
            {
                cout << "Winnner" << endl;
                return true;
            }
        }
        break;
    }
    return false;
}

void tic_tac_toe::Initial_display()
{
    int n1 = 0;
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (i == 1 || i == 3)
                cout << "--";
            else if (j == 1 || j == 3)
                cout << "| ";
            else
            {
                cout << n1 + 1 << " ";
                n1++;
            }
        }
        cout << endl;
    }
```

```
}
void tic_tac_toe::display()
{
    int n1 = 0;
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (i == 1 || i == 3)
                cout << "--";
            else if (j == 1 || j == 3)
                cout << "| ";
            else
            {
                if (arr0[n1] == ' ')
                    cout << "-"
                         << " ";
                else
                    cout << arr0[n1] << " ";
                n1++;
            }
        }
        cout << endl;
    }
}

int main()
{
    int choice;
    cout << "Enter 1 for player vs computer\n 2 for player vs player: ";
    cin >> choice;
    if (choice == 1)
    {
        computer t;
        t.Initial_display();
        cin >> t;
        t.turn();
        do
        {
            if (t.call())
                break;

        } while (t.count < 9);

        if (t.count == 9)
            cout << "Draw" << endl;
    }
    else if (choice == 2)
```

```
    {
        tic_tac_toe t;
        t.Initial_display();
        cin >> t;
        do
        {
            if (t.call())
                break;

        } while (t.count < 9);

        if (t.count == 9)
            cout << "Draw" << endl;
    }
    else
        cout << "Invalid choice\n";
    return 0;
}
```