```
// Samson Fung

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

int w, x, y, z;
int gamescount;
int lengthparameter, n;
int BOARD[9];
int WinXTotal, WinOTotal, TieGameTotal, UnfinishedTotal;

void ResetBoardandVariables();
void loadBOARDFromStream();
void displayGame(int arg[], int lengthparameter);
int WinX();
int WinY();
int getGameStatus();
int displayTotals();

void loadBOARDsFromStream()
    {
        std::ifstream file;
        file.open("games.txt");
        while (!file.eof())
            {
            cout << endl;
            for (z=0; z<9; ++z)
            {
                file >> x >> y;
                w = (x + 3*(y-1));
                if (x != 0 or y != 0)
                    {if(z%2 == 0)
                        BOARD[w-1] = 1;
                    else
                        BOARD[w-1] = 2;}
                else
                {
                    z = 10, gamescount++;
                    getGameStatus();
                    displayGame(BOARD, 9);
                    int BOARD[9] = {0,0,0,0,0,0,0,0,0};
                }
            }
            }
        file.close();
    }

void displayGame(int arg[], int lengthparameter)
    {
        for (n=0; n<lengthparameter; ++n)
            {
            cout << "  ";
            if (arg[n] == 0)
                cout << " ";
            if (arg[n] == 1)
                cout << 'X';
            if (arg[n] == 2)
                cout << 'O';
            cout << "  ";
            if ((n+1)%3 == 0 && n>1)
                cout << endl << "- - - - - - - - -" << endl;
            else cout << '|';
            }
    }

int WinX()
{
    if (BOARD[0] == BOARD[1] && BOARD[1] == BOARD[2] && BOARD[2] == 1)
        return 1;
    else if (BOARD[3] == BOARD[4] && (BOARD[4] == BOARD[5] == 1))
        return 1;
    else if (BOARD[6] == BOARD[7] && BOARD[7] == BOARD[8] && BOARD[8] == 1)
        return 1;
    else if (BOARD[0] == BOARD[3] && BOARD[3] == BOARD[6] && BOARD[6] == 1)
        return 1;
    else if (BOARD[1] == BOARD[4] && BOARD[4] == BOARD[7] && BOARD[7] == 1)
```

```
        return 1;
    else if (BOARD[2] == BOARD[5] && BOARD[5] == BOARD[8] && BOARD[8] == 1)
        return 1;
    else if (BOARD[0] == BOARD[4] && BOARD[4] == BOARD[8] && BOARD[8] == 1)
        return 1;
    else if (BOARD[2] == BOARD[4] && BOARD[4] == BOARD[6] && BOARD[6] == 1)
        return 1;
    else return 0;
}

int WinO()
{
    if (BOARD[0] == BOARD[1] && BOARD[1] == BOARD[2] && BOARD[2] == 2)
        return 2;
    else if (BOARD[3] == BOARD[4] && BOARD[4] == BOARD[5] && BOARD[5] == 2)
        return 2;
    else if (BOARD[6] == BOARD[7] && BOARD[7] == BOARD[8] && BOARD[8] == 2)
        return 2;
    else if (BOARD[0] == BOARD[3] && BOARD[3] == BOARD[6] && BOARD[6] == 2)
        return 2;
    else if (BOARD[1] == BOARD[4] && BOARD[4] == BOARD[7] && BOARD[7] == 2)
        return 2;
    else if (BOARD[2] == BOARD[5] && BOARD[5] == BOARD[8] && BOARD[8] == 2)
        return 2;
    else if (BOARD[0] == BOARD[4] && BOARD[4] == BOARD[8] && BOARD[8] == 2)
        return 2;
    else if (BOARD[2] == BOARD[4] && BOARD[4] == BOARD[6] && BOARD[6] == 2)
        return 2;
    else return 0;
}

int Tie()
    {
        if (BOARD[1] != 0 && BOARD[2] != 0 && BOARD[3] != 0
                    && BOARD[4] != 0 && BOARD[5] != 0 && BOARD[6] != 0
                  && BOARD[7] != 0 && BOARD[8] != 0 && BOARD[9] != 0)
        return 3;
        else return 0;
    }

int getGameStatus()
    {
        if (WinX() == 1 or WinO() == 2 or Tie() == 3)
            {
            if (WinX() == 1)
                {
                    ++WinXTotal;
                    cout << "Game " << gamescount << " is a win for X." << endl;
                }
            else
            if (WinO() == 2)
                {
                    ++WinOTotal;
                    cout << "Game " << gamescount << " is a win for O." << endl;
                }
            else
            if (Tie() == 3)
                {
                    ++TieGameTotal;
                    cout << "Game " << gamescount << " is a tie." << endl;
                }
            }
        else
            {
                ++UnfinishedTotal;
                cout << "Game " << gamescount << " is unfinished." << endl;
            }
    }

int displayTotals()
    {
        cout << endl;
        cout << "X has won " << WinXTotal << " times." << endl;
        cout << "O has won " << WinOTotal << " times." << endl;
        cout << "Total tie games: " << TieGameTotal << "." << endl;
        cout << "Total unfinished games: " << UnfinishedTotal << "." << endl;
        cout << "Total games: " << WinXTotal+WinOTotal+TieGameTotal+UnfinishedTotal <<
```

```
"." << endl;
    }

int main()
{
    loadBOARDsFromStream();
    displayTotals();
    return 0;
}

```