Project 2

Tic-Tac-Toe

CSC 5

Section 46023

Summer 2014

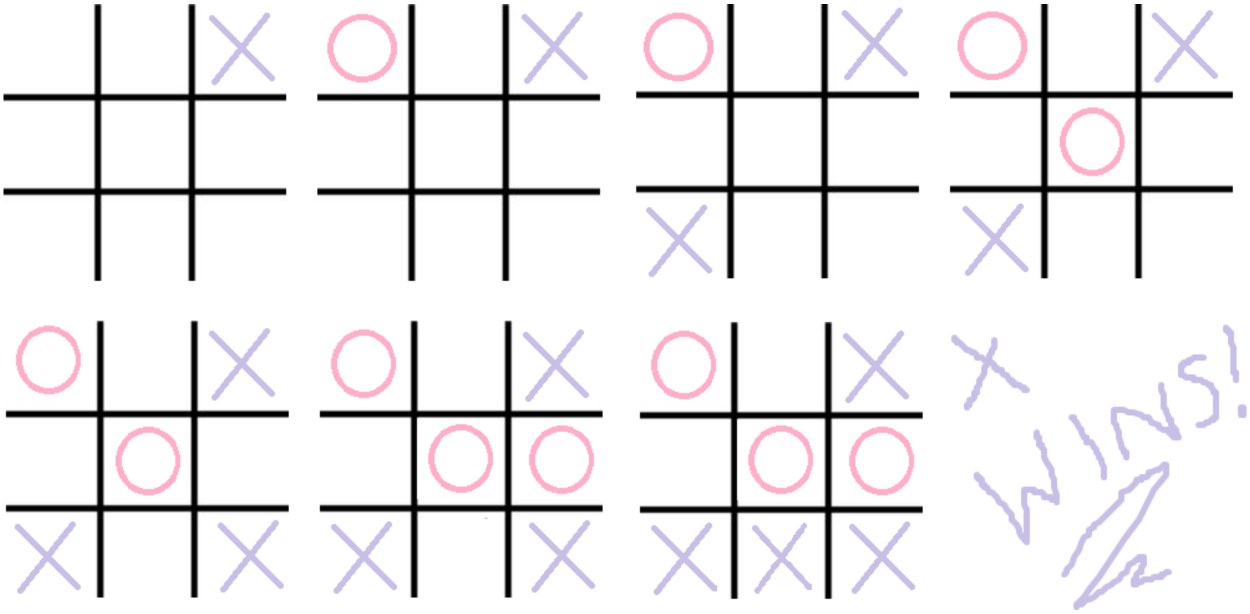
Brynn Caddel

Introduction: Tic-Tac-Toe

The program presented here was created to demonstrate a game of Tic-Tac-Toe. I pity your childhood if you don't know how to play the game by now and suggest you go watch WarGames instead of reading this write up. But for the sake of formalities, the premise of tic-tac-toe is as follows:

> Two players, *X* and *O*, take turns marking the spaces in a 3×3 grid.
> The player who succeeds in placing three respective marks in a horizontal, vertical, or diagonal row wins the game.

For example:



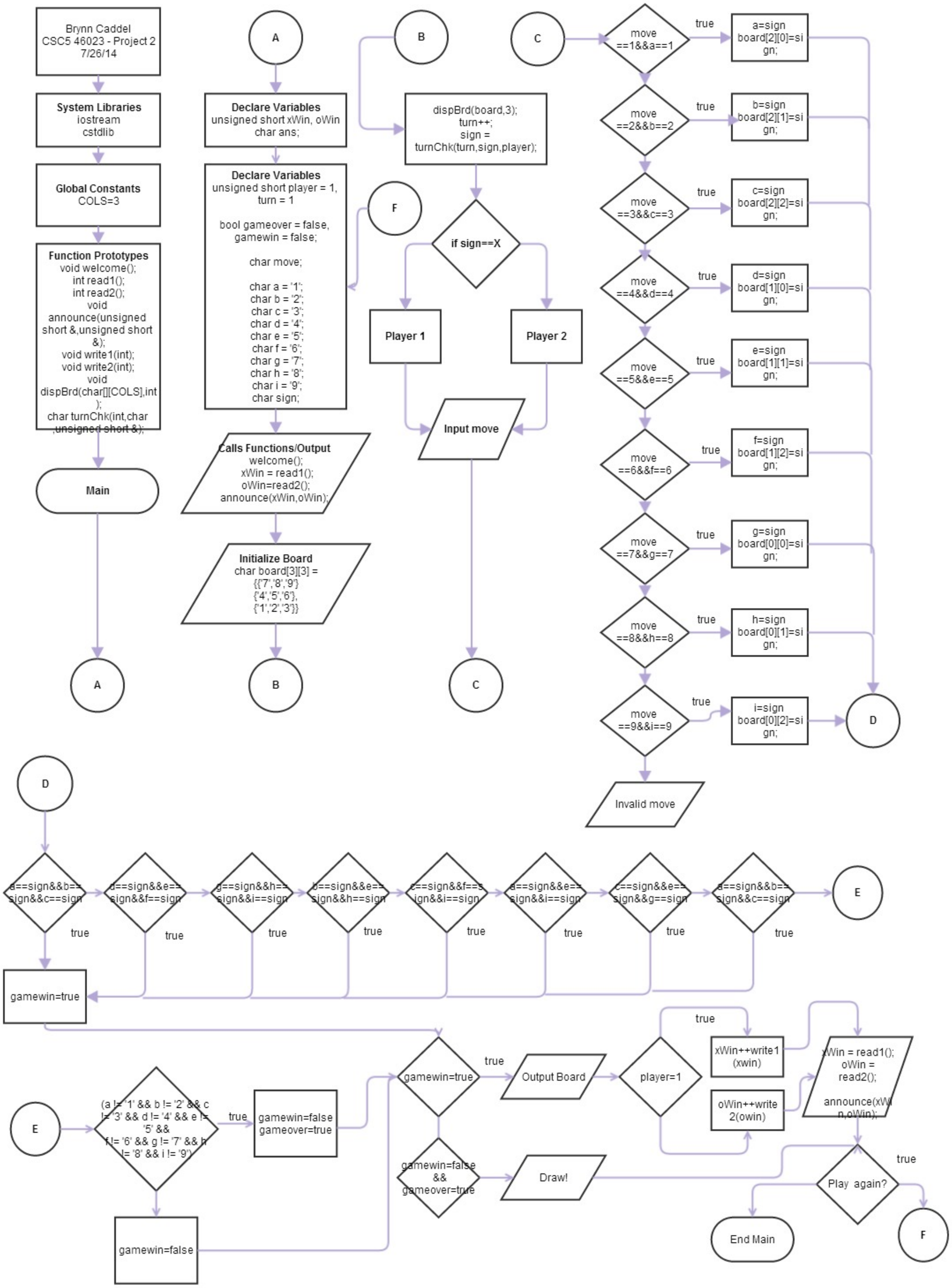
I included a win counter in the game to keep track of how many times player 1 (X) and player 2 (O) wins. This count is displayed at the beginning of each game and is read out to two separate files called "xWin.dat" and "oWin.dat."

Development Summary

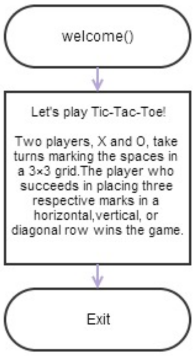
For my first project, I wrote a Blackjack game. I wanted to modify it for the second project, but found it difficult to implement a two-dimensional array. I choose to write a Tic-Tac-Toe program because it seemed like a good chance to practice writing two-dimensional arrays. This program also utilizes functions prototypes to display the board, check the player's turn, read in and write to files, announce the winner of each game, and welcome the user to the game. Writing these functions probably would have been more difficult if it were not for the research I did for the first project where I used functions to deal random cards and randomly assign suits to cards. The most tedious part of writing this came from i/if else statements that checked to see if either X or

O had made one complete line. Reading to and from files also took a little bit of time to get right, but I finally managed to both read in and out successfully.

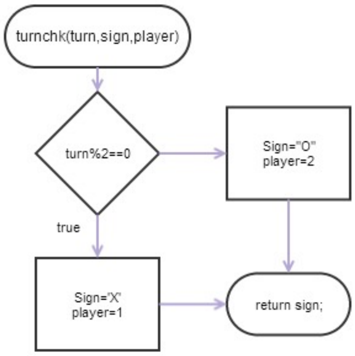
Brynn Caddel - CSC5 46023 - Final Project - Tic Tac Toe



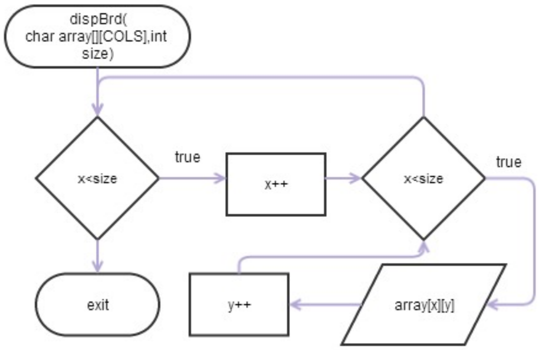
Welcome Function


welcome()
Let's play Tic-Tac-Toe!
Two players, X and O, take turns marking the spaces in a 3×3 grid.The player who succeeds in placing three respective marks in a horizontal,vertical, or diagonal row wins the game.
Exit

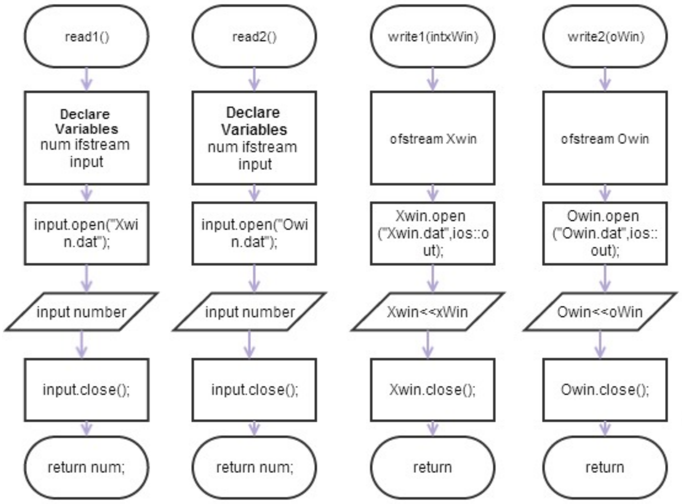
Turn Check Function



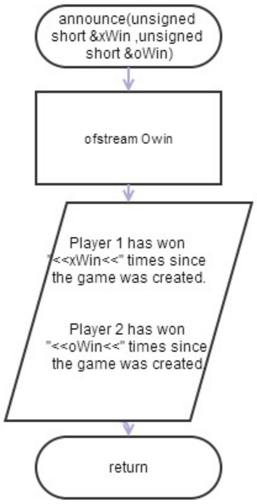
Display Board Function



Read and Write Functions



Announce Function


announce(unsigned short &xWin ,unsigned short &oWin)
ofstream Owin
Player 1 has won "<<xWin<<" times since the game was created.
Player 2 has won "<<oWin<<" times since the game was created.
return

Pseudo Code

```
Declare xWin and oWin as unsigned short
Declare ans as char data type
Do
        Set Player to 1
        Set Turn to 1
        Set gameover to false
        Set gamewin to false
        Run the welcome function to show welcome message
        Read from xwin.dat
        Read from owin.dat
        Announce the win count so far
        Initialize the game
        Set up the 2d array
        Do
        Draw the 3 x 3 grid using a 2d array function.
        + 1 to Turn
        Run the turncheck function
        Set sign = turncheck function output
        If sign is equivalent to X
                Set player to 1
        Else set player to 2Get input
        check move's validity
        If move is valid and a is available
                 Mark 'a' as sign
                 Set [2][0] position on gaming board to sign
        else If move is valid and b is available
                 Mark 'b' as sign
                Set [2][1] position on gaming board to sign
         Else If move is valid and c is available
                 Mark 'c' as sign
                 Set [2][2] position on gaming board to sign
         Else If move is valid and a is available
                 Mark 'd' as sign
                 Set [1][0] position on gaming board to sign
         Else If move is valid and e is available
                 Mark 'e' as sign
                 Set [1][1] position on gaming board to sign
         Else If move is valid and f is available
                 Mark 'f' as sign
                 Set [1][2] position on gaming board to sign
         Else If move is valid and g is available
        Mark ‘g’ as sign
        Set [0][0] position on gaming board to sign
Else If move is valid and h is available
        Mark ‘h’ as sign
        Set [0][1] position on gaming board to sign
 Else If move is valid and i is available
        Mark ‘i’ as sign
        Set [0][2] position on gaming board to sign
 Else
        Prompt for valid move
        -1 to turn in order to compensate for move taken

Check win condition
If a is equivalent to sign and b is equivalent to sign and c is equivalent to sign
        Set gamewin to true
Else If d is equivalent to sign and e is equivalent to sign and f is equivalent to sign
        Set gamewin to true
Else If g is equivalent to sign and h is equivalent to sign and i is equivalent to sign
        Set gamewin to true
Else If a is equivalent to sign and d is equivalent to sign and g is equivalent to sign
        Set gamewin to true
Else If b is equivalent to sign and e is equivalent to sign and h is equivalent to sign
        Set gamewin to true
Else If c is equivalent to sign and f is equivalent to sign and i is equivalent to sign
        Set gamewin to true
Else If a is equivalent to sign and e is equivalent to sign and i is equivalent to sign
        Set gamewin to true
Else If c is equivalent to sign and e is equivalent to sign and g is equivalent to sign
        Set gamewin to true
Else If a is not equal to '1' and b is not equal to '2' and c is not equal to '3' and d is not equal to '4' and e is not equal to '5' and f is not equal to '6' and g is not equal to '7' and h is not equal to '8' and i is not equal to '9'
        Set gamewin to false
        Set gameover to true
```

```
    Else
        Set gamewin to false
    Display result
    If gamewin is true
        Print the placement on the board
        Print the player who won
    If player 1 won
        Add 1 to X’s win counter
        Write the new win count to xwin.dat
    If player 2 won
        Add 1 to O’s win counter
        Write the new win count to owin.dat
    Else if gameover is true and gamewin is false
        Print the placement on the board
        Print “Nobody wins! It's a draw!"
    Run the read1() function
    Run the read2() function
    Announce the new data
    While gameover is false
    Ask for repeat
    Get answer
While answer is yes
Exit Program
```

Variables Used:

Type	Variable Name	Description
unsigned short	xWin	win counter for x
	oWin	win counter for o
	Player	player 1 or 2
	Turn	used to alternate players
char	ans	answer for repeat
	sign	a placeholder for x or o
	a	lower left portion of board
	b	lower mid portion of board
	c	lower right portion of board
	d	left part fo board
	e	middle part of board
	f	right part of board
	g	upper left portion of board
	h	upper mid portion of board
	i	upper right portion board
	Board [3][3]	3x3 array used as board

char	Move	player’s input
Boolean	Gameover	indicates game is over
	Gamewin	indicates game is won
int	size	size to loop the display board
	num	the number in the data file

Concepts Used:

Textbook Used: *Problem Solving with C++ 8th Edition by Walter Savitch*

Chapter 2
2.1 Variables and Assignments
2.2 Input and Output
2.3 Data Types and Expressions
2.4 Simple FLow Control
2.5 Program Style

Chapter 3
3.1 Using Boolean Expressions
3.2 Multiway Branches
3.3 More About C++ Loop Statements
3.4 Designing Loops

Chapter 4
4.1 Top-Down Design
4.2 Predefined Functions
4.3 Programmer-Defined Functions
4.4 Procedural Abstractions
4.5 Scope and Local Variables

Chapter 5
5.1 Void Functions
5.2 Call-By_Reference Parameters
5.4 Testing and Debugging Functions
5.5 General Debugging Techniques

Chapter 6

6.1 Streams and Basic File I/O
6.2 Tools for Stream !/O
6.3 Character I/O

Chapter 7
7.1 Introduction to Arrays
7.2 Arrays in Functions
7.3 Programming with Arrays
7.4 Multidimensional Arrays

From Class Lectures and Lab:

1. Input and Output
2. Loops
3. Branching Constructs
4. Boolean Expressions
5. User Interactivity
6. Two Dimensional Arrays
7. Void Functions

Copy of Code:

```
//System Libraries
#include <cstdlib>
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;

//Global Constants
const int COLS=3;           //3 columns since tic-tac-toe is 3x3

//Function Prototype
void welcome();
int read1();
int read2();
void announce(unsigned short &,unsigned short &);
void write1(int);
void write2(int);
void dispBrd(char[][COLS],int);
char turnChk(int,char ,unsigned short &);

//Execution Begins Here
int main(int argc, char** argv) {
   //Declare Variables
   unsigned short xWin, oWin;          //Counts the wins for X and O
   char ans;                        //Allows the user to proceed with the program
   do{
      //Declare Variables
      unsigned short player = 1, turn = 1;//Initialize the game
      bool gameover = false, gamewin = false;
      char move;
      //Char type for the hidden game board
      char a = '1';
      char b = '2';
      char c = '3';
      char d = '4';
      char e = '5';
      char f = '6';
      char g = '7';
      char h = '8';
      char i = '9';
      char sign;
```

```
//Welcome Message
welcome();                      //Calls Welcome Function
//Read from data file
xWin = read1();
oWin = read2();
//Announces Number of Times X and O Have Won Since Game Began
announce(xWin,oWin);
//Initialize the game
char board[3][3] = {{'7','8','9'},//3x3 array is going to make the board look pretty
                   {'4','5','6'},
                   {'1','2','3'}};

do{

    dispBrd(board,3);            //Display the board using a two-dimensional array

    turn++;                      //Increment the turn to start the game

    sign = turnChk(turn,sign,player);

    if (sign == 'X'){
        player=1;
    }else player=2;

    //Player inputs moves
    cout<<endl;
    cin>>move;

//Determines if move is valid
if(move == '1' && a == '1') {
        a = sign;
        board[2][0] = sign;
}else if(move == '2' && b == '2') {
        b = sign;
        board[2][1] = sign;
}else if(move == '3' && c == '3') {
        c = sign;
        board[2][2] = sign;
}else if(move == '4' && d == '4') {
        d = sign;
        board[1][0] = sign;
}else if(move == '5' && e == '5') {
        e = sign;
        board[1][1] = sign;
}else if(move == '6' && f == '6') {
        f = sign;
        board[1][2] = sign;
}else if(move == '7' && g == '7') {
        g = sign;
        board[0][0] = sign;
}else if(move == '8' && h == '8') {
        h = sign;
        board[0][1] = sign;
}else if(move == '9' && i == '9') {
        i = sign;
        board[0][2] = sign;
}else{  //Use if input is not valid
cout << "Sorry, that was not a valid move. Please enter a valid move." << endl;
turn--; //Decrement 1 turn to compensate for invalid moves
}

//How to determine a win
if(a == sign && b == sign && c == sign){      //check if a,b,and c are in 1 line
    gamewin = true;
}else if(d == sign && e == sign && f == sign){//check if c,e,and f are in 1 line
    gamewin = true;
}else if(g == sign && h == sign && i == sign){//check if g,h,and i are in 1 line
    gamewin = true;
}else if(a == sign && d == sign && g == sign){//check if a,d,and g are in 1 line
    gamewin = true;
}else if(b == sign && e == sign && h == sign){//check if b,e,and h are in 1 line
    gamewin = true;
}else if(c == sign && f == sign && i == sign){//check if c,f,and i are in 1 line
    gamewin = true;
}else if(a == sign && e == sign && i == sign){//check if a,e,and i are in 1 line
    gamewin = true;
}else if(c == sign && e == sign && g == sign){//check if c,e,and g are in 1 line
    gamewin = true;
}else if(a != '1' && b != '2' && c != '3' && d != '4' && e != '5' &&
         f != '6' && g != '7' && h != '8' && i != '9'){
    gamewin = false;    //Draw means no one wins the game
    gameover = true;    //Game over
}else gamewin = false;  //Loop the check winning condition until draw or one wins

//End the game if a player won and display result
if(gamewin == true){
      cout<<endl;
        //Display the ending placement of the game
        cout<<g<<" | "<<h<<" | "<<i<<endl;   //Display the winning board
        cout<<"-----------"<<endl;
        cout<<d<<" | "<<e<<" | "<<f<<endl;
        cout<<"-----------"<<endl;
        cout<<a<<" | "<<b<<" | "<<c<<endl;
        cout<<endl;
        cout<<endl;
        cout<<"Player "<<player<<" won!"<<endl;//Display which player won
        cout<<endl;

        if(player == 1){
          xWin++;       //Increment the win count
          write1(xWin);
        }
        if(player == 2){
          oWin++;       //Increment the win count
          write2(oWin);
        }
        gameover = true;  //Game is over

    }else if(gameover == true && gamewin == false){//Game over and no one wins
        cout<<endl;
        cout<<g<<" | "<<h<<" | "<<i<<endl;    //Display the  board
        cout<<"-----------"<<endl;
        cout<<d<<" | "<<e<<" | "<<f<<endl;
        cout<<"-----------"<<endl;
        cout<<a<<" | "<<b<<" | "<<c<<endl;
        cout<<endl;
        cout<<endl;
        cout<<"Draw! No one wins."<<endl;     //Displayed if game is a draw
    }

      }while(gameover == false);                //Loops the game until game is over
      //Files that keep track of winnings
      xWin = read1();//read from xwin.dat
      oWin = read2();//read from owin.dat
      announce(xWin,oWin);

      cout<<"Do you want to repeat?"<<endl;      //Asks if the user would like to repeat the
game
      cin>>ans;//get answer
    }while(ans == 'Y'||ans == 'y');
    //QED
    return 0;
}

//Function Definitions
void dispBrd(char array[][COLS],int size){

    for (int x=0; x<size; x++)
    {
        cout<<endl;
        for (int y=0;y<size;y++)
        {
            cout<<setw(2)<<array[x][y]<<setw(2);  //This draws the grid
            cout<<setw(2);
        }
        cout<<endl;
    }
    cout<<endl;
}

char turnChk(int turn,char sign,unsigned short &player ){

        if(turn%2 == 0) {//if the turn is multiples of 2,then it'll be player 1's turn.
        sign = 'X';
        player = 1;

        //skip 2 lines for better visibility
        cout<<endl;
        cout<<endl;
        cout<<"Player "<<player<<"'s turn to play."<<endl;
        cout<<"Please enter the number to place your sign."<<endl;
        cout<<"You are "<<sign<<endl;//reminding the player about their sign

    }else{
        sign = 'O';
        player = 2;//player 2's turn

        //skip 2 lines for better visibility
        cout<<endl;
        cout<<endl;
        cout<<"Player "<<player<<"'s turn to play"<<endl;
        cout<<"Please enter the number to place your sign."<<endl;
        cout<<"You are "<<sign<<endl;//reminding the player about their sign

    }
        return sign;  //return sign to be used to determine move's validity
}

int read1(){
    //Declare variables
    ifstream input;
    int num;
    //open the file
    input.open("Xwin.dat");
    //Read the data
    input>>num;
    //close the file
    input.close();
    //exit
    return num;
}

int read2(){
    //Declare variables
    ifstream input;
    int num;
    //open the file
    input.open("Owin.dat");
    //Read the data
    input>>num;
    //close the file
    input.close();
    //exit
    return num;
}

void write1(int xWin){
    ofstream Xwin;
    //open the file
    Xwin.open ("Xwin.dat",ios::out);
    //record the data
    Xwin<<xWin;
    //close the file
    Xwin.close();
}
```

```
void write2(int oWin){
    ofstream Owin;
    //open the file
    Owin.open ("Owin.dat",ios::out);
    //record the data
    Owin<<oWin;
    //close the file
    Owin.close();
}

void announce(unsigned short &xWin ,unsigned short &oWin){

    cout<<"Player 1 has won "<<xWin<<" times since the game was created."<<endl;
    cout<<"Player 2 has won "<<oWin<<" times since the game was created."<<endl;

}

void welcome(){
    cout<<"Let's play Tic-Tac-Toe!"<<endl;
    cout<<endl;
    cout<<"Rules: Two players, X and O, take turns marking the spaces in a 3×3 grid."<<endl;
    cout<<"The player who succeeds in placing three respective marks in a horizontal,"<<endl;
    cout<<"vertical, or diagonal row wins the game. "<<endl;
    cout<<endl;
    cout<<"Player 1 will start with X."<<endl;
    cout<<endl;

}
//End Function Definitions
```