Joseph Reimbold
CSC-17A 42448
6/9/2014

Project

Tic Tac Toe!

Introduction
Game: Tic Tac Toe
The objective of this game is to select three spaces of a grid that make a line (3 in a row) in order to win.
Ex) If you can line up all 3 spaces in a horizontal or vertical row then you win the game.
It is possible for the game to end in a draw if both players are able to counter the others moves.

Summary
Project Size: Approx 800 lines
# of Variables Used: 25+
# of Functions Used: 10
# of Classes Used: 2
This game is a very basic game which everybody knows and loves. I used a number of functions in order to handle each main task of the game. The game features a 2 player mode and 2 different AI opponents which are Easy and Advanced.

Description
I chose this program because it was a game that I used to play as a kid quite a bit and thought it would be fun to recreate it from scratch. After creating a base for my first project I wanted to return and improve it quite a bit. Every step of the game was added into functions. I also have the game reading in player names from a file and outputting the results to a file specified by the user at the end. A 1d array holds the game spaces and an array of structures was used to keep track of wins and losses for each player (or AI) as the user(s) play multiple games consecutively.

Things To Improve
- Create a GUI and play with the mouse (will investigate this winter break!)
- Create a database stored with player win and loss ratios against each other and each of the different AI opponents

Psuedo Code
```
//Libraries
//Globals
//Function Prototypes

//Begin Execution Here
    //Declare Variables
    //Title Screen
    //inform user about settings file so they may update their name
    //prompt for number of players or AI opponent selection
    //read in player names from settings.txt
    //play while user selects yes

    //alternate first player's turn each game
    //reset game board values
        //continue running until there is a winner
```

```
                //change AI name if playing against AI
                //Call gameBoard function to draw board
                //Call plyrTrn function to determine which player's turn
                //Call gtSpc function to get user's selection for space
                //Call mrkSpc function to get mark the board with selection
                //Call gmOvr function to check to see if game is over
        //Call gameBoard last time to output final result to screen
        //Call rslt function to output overall stats to screen
        //Ask if user would like to play the game again
        //Call rcrdScr function to record the score to file
//End of Main

//rcrdScr function
        //get file name for output file from user
        //declare the outfile and open it
        //output final results to screen
        //output results to file
        //close file

//rslt function
        //Show end results of game

//gmOvr function
        //Determine if game ends in a win
        //Determine if game ends in draw
        //Determine if game continues

//mrkSpc function
        //Check users selection and compare it with empty space and mark it

//advAI function
        //Check for 2 horizontal spots held by player and chooses remaining
        //Check for 2 vertical spots held by player and chooses remaining
        //Check for 2 diagonal spots held by player and chooses remaining
        //If none of above, select random available

//gtSpc function
        //select space if player 1's turn
        //select space for player 2's turn
                //if player 2 is Easy AI
                //if player 2 is Advanced AI
                //if player 2 is a second user
        //verify space input is valid

//plyrTrn function
        //Set player1 to X
        //Set player2 to O

//gameBoard function
```

```
        //Output board to screen

//sttngs function
        //declare inputfile and open it
        //read in player names from file
```

Major Variables

Type	Variable Name	Description	Location
Player	P1, P2	Class which holds name and wins/loss data	Main(), gameboard(),gtSpc(),rslt(),rcrdScr(), “player.h”, “player.cpp”, “computer.h”
Computer	P2	Inherited Class from Player Class, additionally holds the difficulty setting	Main(), gameboard(),gtSpc(),rslt(),rcrdScr(),”computer.h”
int	state	Stores the status of the game (0=not over, 1=win, 2=draw)	Main(), rslt()
	SIZE	Set to 10 for game board	main()
	player	Determines which players turn it is	Main(), plyrTrn(), gtSpc(), mrkSpc(), gmOvr(), rslt()
	numPlyr	Sets if you are playing against easy or advanced AI or a second player	Main(), Sttngs(), gtSpc()
	gmNum	Holds game number for current session	Main(), GameBoard(), gmOvr()
	draws	Counts number of draws this session	Main(), rslt()
char	board[SIZE]	Holds game board values	Main(), gameBoard(), gtSpc, advAI(), mrkSpc(), gmOvr(),
	space	Current board selection	Main(), gtSpc(), mrkSpc()
	choice	Holds play again value	main()
	plyrMrk	Holds 'X' or 'O' for proper player	Main(), mrkSpc()
ifstream	inputFile	Used to input info from file	sttngs()

ofstream	outputFile	Used to store info in file	rcrdScr()

Reference

1. *Starting out with C++* - Tony Gaddis

Program

```
/*
  Joseph Reimbold
  5/30/2014
  Project 2: Tic Tac Toe
*/

/* Things to Add/Modify!
 * Class - done (player and computer classes)
 * Copy Constructor - done (copies one player into a new one)
 * Operator Overloading - done (= operator for copying player information)
 * Inheritance - done (Computer class inherits Player class)
 * Polymorphism - done (many functions accept 2 players
 *                      even if p2 is a derived computer class)
 * Virtual Function
 * Abstract Class
 * Templates - done (sttngs function works with both player or computer classes)
 * Attempt Exception?
 */

//Libraries
#include <cstdlib>
#include <iostream>
#include <string>
#include <fstream>
#include <ctime>
using namespace std;

//Our Libraries
#include "player.h"
#include "computer.h"

//No Globals

//Function Prototypes
void gameBoard(Player &,Player &,char [],int);
char plyrTrn(int &);
int advAI(char []);
int exprtAI(char []);
char gtSpc(Player &,Player &,char [],int,int);
void mrkSpc(char [],char,char,int &);
int gmOvr(char [],int &);
```

```
void rslt(Player &,Player &,int,int,int &);
void rcrdScr(Player &,Player &,int,int);

//Template Functions created to handle if player 2 is player or computer class
template <class T>
void sttngs(int numP, Player &p1, T &p2){
   //sets default name for p1 if none in file
   string p1Name="Derp";
   p1.setName(p1Name);

   //declare inputfile and open it
   ifstream inputFile;
   inputFile.open("settings.txt");

   //read in player names
   inputFile>>p1Name;
   p1.setName(p1Name);

   //sets default name for p2 if none in file
   if(numP==4){
      string p2Name="Bob";
      p2.setName(p2Name);
      inputFile>>p2Name;
      p2.setName(p2Name);
   }

   //close inputfile
   inputFile.close();
}

//Begin Execution Here
int main(int argc, char *argv[])
{
   //Declare Variables
   const int SIZE=10,PLYRS=2;
   char board[SIZE]={'0','1','2','3','4','5','6','7','8','9'}; //Board spaces
   char space; //Board Selection space
   char choice='y'; //Play again?
   int state=0; //determines if game is over
   int player=1; //Determines which player's turn is happening
   int numPlyr; //holds number of players
   int gmNum=1; //holds number of games played
   int draws=0; //counts number of draws
   char plyrMrk='X'; //Bases the X or O off player #
   srand(static_cast<unsigned int>(time(0)));//seed random number generator

   do{
      //system("CLS");
      //Introduce game
```

```
cout<<"**********************************************************************"<<en
dl;
      cout<<"* ______ ______ ______   ______    _    ______   ______ ______ ______ *"<<endl;
      cout<<"*  __     __  __         __     _ _  __         __   __  __ __     *"<<endl;
      cout<<"*  __     __  __         __    _____ __         __   __  __ ____     *"<<endl;
      cout<<"*  __     __  __         __   __   __ __        __   __  __ __     *"<<endl;
      cout<<"*  __   ______ ______     __   __   __ ______    __   ______ ______ *"<<endl;
      cout<<"*                                                         *"<<endl;

cout<<"**********************************************************************"<<en
dl;
      cout<<endl<<endl;

      //inform user about settings file so they may update their name
      cout<<"***NOTE***"<<endl;
      cout<<"To edit player names, open up 'settings.txt' found within"
         <<"\nthis program's folder and replace the two that are there!"<<endl<<endl<<endl;

      //prompt for number of players or AI opponent selection
      cout<<"Enter 1 for Easy AI opponent"<<endl;
      cout<<"Enter 2 for Advanced AI opponent"<<endl;
      cout<<"Enter 3 for Expert AI opponent"<<endl;
      cout<<"Enter 4 for 2 players!"<<endl;
      cin>>numPlyr;
   }while(numPlyr<1||numPlyr>4);//check proper input for selection

   //play based on number of players selection
   if(numPlyr==1||2||3){
      Player p1;
      Computer p2(numPlyr);
      sttngs(numPlyr, p1, p2);

      //play while user selects yes
      do{
         //alternate first player's turn each game
         if(gmNum%2==1)
            player=1;//player 1 gets first turn if odd # game
         else
            player=2;//player 2 gets first turn if even # game
         state=0;//resets status of game to "not over"

         //reset game board values
         for (int i=0;i<SIZE;i++)
            board[i]='0'+i;

         //continue running until there is a winner
         do{
            //draw board
            gameBoard(p1,p2,board,gmNum);

            //determine which player's turn
            plyrMrk=plyrTrn(player);

            //get user's selection for space
            space=gtSpc(p1,p2,board,player,numPlyr);

            //mark the board with selection
            mrkSpc(board,space,plyrMrk,player);

            //check to see if game is over
            state=gmOvr(board,player);

        }while(state==0);

        //display result of game end
        gameBoard(p1,p2,board,gmNum);
        rslt(p1,p2,state,player,draws);

        //Ask if user would like to play the game again
        cout<<"Would you like to play again? (Y/N)"<<endl;
        cin>>choice;

    gmNum++; //adds to game# each time played
    //runs again if yes
    }while (choice=='y'||choice=='Y');

    //records the score by outputting to file
    rcrdScr(p1,p2,gmNum,draws);
}
else if(numPlyr==4){
    Player p1;
    Player p2;
    sttngs(numPlyr, p1, p2);

    //play while user selects yes
    do{
        //alternate first player's turn each game
        if(gmNum%2==1)
            player=1;//player 1 gets first turn if odd # game
        else
            player=2;//player 2 gets first turn if even # game
        state=0;//resets status of game to "not over"

        //reset game board values
        for (int i=0;i<SIZE;i++)
            board[i]='0'+i;
```

```
         //continue running until there is a winner
         do{
            //draw board
            gameBoard(p1,p2,board,gmNum);

            //determine which player's turn
            plyrMrk=plyrTrn(player);

            //get user's selection for space
            space=gtSpc(p1,p2,board,player,numPlyr);

            //mark the board with selection
            mrkSpc(board,space,plyrMrk,player);

            //check to see if game is over
            state=gmOvr(board,player);

         }while(state==0);

         //display result of game end
         gameBoard(p1,p2,board,gmNum);
         rslt(p1,p2,state,player,draws);

         //Ask if user would like to play the game again
         cout<<"Would you like to play again? (Y/N)"<<endl;
         cin>>choice;

      gmNum++; //adds to game# each time played
      //runs again if yes
      }while (choice=='y'||choice=='Y');

      //records the score by outputting to file
      rcrdScr(p1,p2,gmNum,draws);
   }

   //system("PAUSE");
   return EXIT_SUCCESS;
}//End of Main

void gameBoard(Player &p1,Player &p2,char a[],int g){
   //draws the game board
   //system("CLS");
   cout<<"Game "<<g<<endl;
   cout<<"\n"<<p1.getName()<<" is X's and "<<p2.getName()<<" is O's."<<endl<<endl;
   cout<<"     ___ ___ ___ "<<endl;
   cout<<"    |   |   |   |"<<endl;
   cout<<"    | "<<a[7]<<" | "<<a[8]<<" | "<<a[9]<<" |"<<endl;
   cout<<"    |___|___|___|"<<endl;
   cout<<"    |   |   |   |"<<endl;
```

```
    cout<<"    | "<<a[4]<<" | "<<a[5]<<" | "<<a[6]<<" |"<<endl;
    cout<<"    |___|___|___|"<<endl;
    cout<<"    |   |   |   |"<<endl;
    cout<<"    | "<<a[1]<<" | "<<a[2]<<" | "<<a[3]<<" |"<<endl;
    cout<<"    |___|___|___|"<<endl<<endl;
}

//determines if the  game is over
int gmOvr(char a[],int &p){
    int s=0;

    //game ends in a win
    if (a[1]==a[2]&&a[2]==a[3])
        s=1;
    else if (a[4]==a[5]&&a[5]==a[6])
        s=1;
    else if (a[7]==a[8]&&a[8]==a[9])
        s=1;
    else if (a[1]==a[4]&&a[4]==a[7])
        s=1;
    else if (a[2]==a[5]&&a[5]==a[8])
        s=1;
    else if (a[3]==a[6]&&a[6]==a[9])
        s=1;
    else if (a[1]==a[5]&&a[5]==a[9])
        s=1;
    else if (a[3]==a[5]&&a[5]==a[7])
        s=1;
    //game ends in draw
    else if (a[1]!='1'&&a[2]!='2'&&a[3]!='3'&&a[4]!='4'
        &&a[5]!='5'&&a[6]!='6'&&a[7]!='7'&&a[8]!='8'&&a[9]!='9')
        s=2;
    //game continues
    else{
        s=0;
        p++;
    }
    return s;
}

char gtSpc(Player &p1,Player &p2,char a[],int p,int numP){
    //get player's selection for space
    string space; //holds space typed by player
    char sp; //holds actual char value of first digit in string
    int aisp; //holds random# generated for easy ai selection
    do{
        //select space if player 1's turn
        if(p==1){
            cout<<p1.getName()<<", make your selection by typing the space number: ";
            cin>>space;
            //truncates the string and takes only the first character in the string
            sp=space[0];
        }
        //select space for player 2's turn
        else{
            //if player 2 is Easy AI
            if(numP==1){
                aisp=rand()%9+1;
                sp='0'+aisp;
            }
            //if player 2 is Advanced AI
            if(numP==2){
                sp='0'+advAI(a);
            }
            //if player 2 is Expert AI
            if(numP==3){
                sp='0'+exprtAI(a);
            }

            //if player 2 is a second user
            if(numP==4){
                cout<<p2.getName()<<", make your selection by typing the space number: ";
                cin>>space;
                //truncates the string and takes only the first character in the string
                sp=space[0];
            }
        }
        if(sp!='1'&&sp!='2'&&sp!='3'&&sp!='4'&&
            sp!='5'&&sp!='6'&&sp!='7'&&sp!='8'&&sp!='9')
            cout<<"Invalid selection!"<<endl;
    }while(sp!='1'&&sp!='2'&&sp!='3'&&sp!='4'&&
            sp!='5'&&sp!='6'&&sp!='7'&&sp!='8'&&sp!='9');
    return sp;
}

//marks the space the player selects
void mrkSpc(char a[],char sp,char mrk,int &p){
     if (sp=='1'&&a[1]=='1')
          a[1]=mrk;
     else if (sp=='2'&&a[2]=='2')
          a[2]=mrk;
     else if (sp=='3'&&a[3]=='3')
          a[3]=mrk;
     else if (sp=='4'&&a[4]=='4')
          a[4]=mrk;
     else if (sp=='5'&&a[5]=='5')
          a[5]=mrk;
     else if (sp=='6'&&a[6]=='6')
```

```
        a[6]=mrk;
    else if (sp=='7'&&a[7]=='7')
        a[7]=mrk;
    else if (sp=='8'&&a[8]=='8')
        a[8]=mrk;
    else if (sp=='9'&&a[9]=='9')
        a[9]=mrk;
    else{
        p--;//decrement player so that it runs again for the same player
    }
}

//Advanced AI board selection
int advAI(char a[]){
    //Checks for 2 horizontal and chooses remaining to win
    if(a[1]=='O'&&a[2]=='O'&&a[3]=='3')return 3;
    if(a[1]=='O'&&a[3]=='O'&&a[2]=='2')return 2;
    if(a[2]=='O'&&a[3]=='O'&&a[1]=='1')return 1;

    if(a[4]=='O'&&a[5]=='O'&&a[6]=='6')return 6;
    if(a[4]=='O'&&a[6]=='O'&&a[5]=='5')return 5;
    if(a[5]=='O'&&a[6]=='O'&&a[4]=='4')return 4;

    if(a[7]=='O'&&a[8]=='O'&&a[9]=='9')return 9;
    if(a[7]=='O'&&a[9]=='O'&&a[8]=='8')return 8;
    if(a[8]=='O'&&a[9]=='O'&&a[7]=='7')return 7;

    //Checks for 2 vertical and chooses remaining to win
    if(a[1]=='O'&&a[4]=='O'&&a[7]=='7')return 7;
    if(a[1]=='O'&&a[7]=='O'&&a[4]=='4')return 4;
    if(a[4]=='O'&&a[7]=='O'&&a[1]=='1')return 1;

    if(a[2]=='O'&&a[5]=='O'&&a[8]=='8')return 8;
    if(a[2]=='O'&&a[8]=='O'&&a[5]=='5')return 5;
    if(a[5]=='O'&&a[8]=='O'&&a[2]=='2')return 2;

    if(a[3]=='O'&&a[6]=='O'&&a[9]=='9')return 9;
    if(a[3]=='O'&&a[9]=='O'&&a[6]=='6')return 6;
    if(a[6]=='O'&&a[9]=='O'&&a[3]=='3')return 3;

    //Checks for 2 diagonal and chooses remaining to win
    if(a[1]=='O'&&a[5]=='O'&&a[9]=='9')return 9;
    if(a[1]=='O'&&a[9]=='O'&&a[5]=='5')return 5;
    if(a[5]=='O'&&a[9]=='O'&&a[1]=='1')return 1;

    if(a[3]=='O'&&a[5]=='O'&&a[7]=='7')return 7;
    if(a[3]=='O'&&a[7]=='O'&&a[5]=='5')return 5;
    if(a[5]=='O'&&a[7]=='O'&&a[3]=='3')return 3;
    //Checks for 2 horizontal spots held by player and blocks
    if(a[1]=='X'&&a[2]=='X'&&a[3]=='3')return 3;
    if(a[1]=='X'&&a[3]=='X'&&a[2]=='2')return 2;
    if(a[2]=='X'&&a[3]=='X'&&a[1]=='1')return 1;

    if(a[4]=='X'&&a[5]=='X'&&a[6]=='6')return 6;
    if(a[4]=='X'&&a[6]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[6]=='X'&&a[4]=='4')return 4;

    if(a[7]=='X'&&a[8]=='X'&&a[9]=='9')return 9;
    if(a[7]=='X'&&a[9]=='X'&&a[8]=='8')return 8;
    if(a[8]=='X'&&a[9]=='X'&&a[7]=='7')return 7;

    //Checks for 2 vertical spots held by player and blocks
    if(a[1]=='X'&&a[4]=='X'&&a[7]=='7')return 7;
    if(a[1]=='X'&&a[7]=='X'&&a[4]=='4')return 4;
    if(a[4]=='X'&&a[7]=='X'&&a[1]=='1')return 1;

    if(a[2]=='X'&&a[5]=='X'&&a[8]=='8')return 8;
    if(a[2]=='X'&&a[8]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[8]=='X'&&a[2]=='2')return 2;

    if(a[3]=='X'&&a[6]=='X'&&a[9]=='9')return 9;
    if(a[3]=='X'&&a[9]=='X'&&a[6]=='6')return 6;
    if(a[6]=='X'&&a[9]=='X'&&a[3]=='3')return 3;

    //Checks for 2 diagonal spots held by player and blocks
    if(a[1]=='X'&&a[5]=='X'&&a[9]=='9')return 9;
    if(a[1]=='X'&&a[9]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[9]=='X'&&a[1]=='1')return 1;

    if(a[3]=='X'&&a[5]=='X'&&a[7]=='7')return 7;
    if(a[3]=='X'&&a[7]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[7]=='X'&&a[3]=='3')return 3;

    //if none of the above, select random available
    int aisp=rand()%9+1;
    return aisp;
}

//Expert AI board selection
int exprtAI(char a[]){
    //if first turn is computer's turn, choose middle or corner
    if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
       a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
        int aisp=rand()%9+1;
        if(aisp==1||aisp==3||aisp==5||aisp==7||aisp==9)return aisp;
    }
```

```
//if player has first turn and chooses middle, computer chooses a corner
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='X'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    int aisp=rand()%9+1;
    if(aisp==1||aisp==3||aisp==7||aisp==9)return aisp;
}

//if player has first turn and chooses a corner, computer chooses mid
if(a[1]=='X'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    return 5;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='X'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    return 5;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='X'&&a[8]=='8'&&a[9]=='9'){
    return 5;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='X'){
    return 5;
}

//if player has first turn and chooses a side, computer chooses
if(a[1]=='1'&&a[2]=='X'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    int aisp=rand()%9+1;
    return aisp;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='X'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    int aisp=rand()%9+1;
    return aisp;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='X'&&a[7]=='7'&&a[8]=='8'&&a[9]=='9'){
    int aisp=rand()%9+1;
    return aisp;
}
if(a[1]=='1'&&a[2]=='2'&&a[3]=='3'&&a[4]=='4'&&a[5]=='5'&&
   a[6]=='6'&&a[7]=='7'&&a[8]=='X'&&a[9]=='9'){
    int aisp=rand()%9+1;
    return aisp;
}

//Checks for 2 horizontal and chooses remaining to win
if(a[1]=='O'&&a[2]=='O'&&a[3]=='3')return 3;
if(a[1]=='O'&&a[3]=='O'&&a[2]=='2')return 2;
if(a[2]=='O'&&a[3]=='O'&&a[1]=='1')return 1;

if(a[4]=='O'&&a[5]=='O'&&a[6]=='6')return 6;
if(a[4]=='O'&&a[6]=='O'&&a[5]=='5')return 5;
if(a[5]=='O'&&a[6]=='O'&&a[4]=='4')return 4;

if(a[7]=='O'&&a[8]=='O'&&a[9]=='9')return 9;
if(a[7]=='O'&&a[9]=='O'&&a[8]=='8')return 8;
if(a[8]=='O'&&a[9]=='O'&&a[7]=='7')return 7;

//Checks for 2 vertical and chooses remaining to win
if(a[1]=='O'&&a[4]=='O'&&a[7]=='7')return 7;
if(a[1]=='O'&&a[7]=='O'&&a[4]=='4')return 4;
if(a[4]=='O'&&a[7]=='O'&&a[1]=='1')return 1;

if(a[2]=='O'&&a[5]=='O'&&a[8]=='8')return 8;
if(a[2]=='O'&&a[8]=='O'&&a[5]=='5')return 5;
if(a[5]=='O'&&a[8]=='O'&&a[2]=='2')return 2;

if(a[3]=='O'&&a[6]=='O'&&a[9]=='9')return 9;
if(a[3]=='O'&&a[9]=='O'&&a[6]=='6')return 6;
if(a[6]=='O'&&a[9]=='O'&&a[3]=='3')return 3;

//Checks for 2 diagonal and chooses remaining to win
if(a[1]=='O'&&a[5]=='O'&&a[9]=='9')return 9;
if(a[1]=='O'&&a[9]=='O'&&a[5]=='5')return 5;
if(a[5]=='O'&&a[9]=='O'&&a[1]=='1')return 1;

if(a[3]=='O'&&a[5]=='O'&&a[7]=='7')return 7;
if(a[3]=='O'&&a[7]=='O'&&a[5]=='5')return 5;
if(a[5]=='O'&&a[7]=='O'&&a[3]=='3')return 3;

//Checks for 2 horizontal spots held by player and blocks
if(a[1]=='X'&&a[2]=='X'&&a[3]=='3')return 3;
if(a[1]=='X'&&a[3]=='X'&&a[2]=='2')return 2;
if(a[2]=='X'&&a[3]=='X'&&a[1]=='1')return 1;

if(a[4]=='X'&&a[5]=='X'&&a[6]=='6')return 6;
if(a[4]=='X'&&a[6]=='X'&&a[5]=='5')return 5;
if(a[5]=='X'&&a[6]=='X'&&a[4]=='4')return 4;

if(a[7]=='X'&&a[8]=='X'&&a[9]=='9')return 9;
if(a[7]=='X'&&a[9]=='X'&&a[8]=='8')return 8;
if(a[8]=='X'&&a[9]=='X'&&a[7]=='7')return 7;

//Checks for 2 vertical spots held by player and blocks
if(a[1]=='X'&&a[4]=='X'&&a[7]=='7')return 7;
if(a[1]=='X'&&a[7]=='X'&&a[4]=='4')return 4;
```

```
    if(a[4]=='X'&&a[7]=='X'&&a[1]=='1')return 1;

    if(a[2]=='X'&&a[5]=='X'&&a[8]=='8')return 8;
    if(a[2]=='X'&&a[8]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[8]=='X'&&a[2]=='2')return 2;

    if(a[3]=='X'&&a[6]=='X'&&a[9]=='9')return 9;
    if(a[3]=='X'&&a[9]=='X'&&a[6]=='6')return 6;
    if(a[6]=='X'&&a[9]=='X'&&a[3]=='3')return 3;

    //Checks for 2 diagonal spots held by player and blocks
    if(a[1]=='X'&&a[5]=='X'&&a[9]=='9')return 9;
    if(a[1]=='X'&&a[9]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[9]=='X'&&a[1]=='1')return 1;

    if(a[3]=='X'&&a[5]=='X'&&a[7]=='7')return 7;
    if(a[3]=='X'&&a[7]=='X'&&a[5]=='5')return 5;
    if(a[5]=='X'&&a[7]=='X'&&a[3]=='3')return 3;

    //if above conditions are not met, random select available
    int aisp=rand()%9+1;
    return aisp;
}

//determine which player's turn it is
char plyrTrn(int &p){
     if (p%2==1){
        p=1;
        return 'X';
     }
     else if (p%2==0){
        p=2;
        return 'O';
     }
}

void rslt(Player &p1,Player &p2,int s,int p,int &d){
    //Show end results of game
    if (s==1){
      if(p%2==1){
          cout<<p1.getName()<<" is the winner! Good job!"<<endl;
          p1.incWins();
          p2.incLosses();
      }
      if(p%2==0){
          cout<<p2.getName()<<" is the winner! Good job!"<<endl;
          p1.incLosses();
          p2.incWins();
      }
    }
    if (s==2){
        cout<<"The game has ended in a draw!"<<endl;
        d++;
    }
}

void rcrdScr(Player &p1,Player &p2,int g,int d){
    //outputs final overall stats and stores them in a file designated by the user
    string rsltFl; //file name to store results to file

    //get file name for output file
    cout<<endl;
    cout<<"Please specify a filename that you would like"<<endl
        <<"to output the results to (ex:'results.txt'): ";
    cin>>rsltFl;

    //declare the outfile and open
    ofstream outputFile;
    outputFile.open(rsltFl.c_str());

    //final results output to screen
    cout<<endl<<"You can find the file "<<rsltFl<<" within"<<endl
        <<"the program folder."<<endl<<endl;
    cout<<"Games Played = "<<(g-1)<<" Draws = "<<d<<endl;
    cout<<p1.getName()<<"'s score: "<<endl<<p1.getWins()<<"W "<<p1.getLosses()<<"L"<<endl;
    cout<<p2.getName()<<"'s score: "<<endl<<p2.getWins()<<"W "<<p2.getLosses()<<"L"<<endl;

    //output results to file
    outputFile<<"Games Played = "<<(g-1)<<" Draws = "<<d<<endl;
    outputFile<<p1.getName()<<"'s score: "<<endl<<p1.getWins()<<"W
"<<p1.getLosses()<<"L"<<endl;
    outputFile<<p2.getName()<<"'s score: "<<endl<<p2.getWins()<<"W
"<<p2.getLosses()<<"L"<<endl;

    //close file
    outputFile.close();
}
```

player.h

```
/*
 * File:   player.h
 * Author: Joe
 *
 * Created on June 6, 2014, 3:28 PM
 */

#include <iostream>
#include <string>
```

```
using namespace std;

#ifndef PLAYER_H
#define         PLAYER_H

class Player{
protected:
    string name;   //holds player name
    char mrkr;     //holds x's or o's
    int wins;      //holds wins
    int losses;    //holds losses
public:
    //Constructors
    Player();
    Player(string);
    //Copy Constructor
    Player(const Player &);
    //Accessors
    string getName();
    char getMrkr();
    int getWins();
    int getLosses();
    //Mutators
    void setName(string);
    void setMrkr(char);
    void setWins(int);
    void incWins();
    void setLosses(int);
    void incLosses();
    //Overloaded = operator
    void operator = (const Player &);
};

#endif /* PLAYER_H */
```

player.cpp

```
#include <string>
#include "player.h"
using namespace std;

//Constructors
Player::Player(){
    name="Player";
    mrkr='-';
    wins=0;
    losses=0;
}

Player::Player(string n){
    name=n;
    mrkr='-';
    wins=0;
    losses=0;
}

//Copy Constructor
Player::Player(const Player &obj){
    name=obj.name;
    mrkr=obj.mrkr;
    wins=obj.wins;
    losses=obj.losses;
}

//Accessors
string Player::getName(){
    return name;
}

char Player::getMrkr(){
    return mrkr;
}

int Player::getWins(){
    return wins;
}

int Player::getLosses(){
    return losses;
}

//Mutators
void Player::setName(string n){
    name=n;
}

void Player::setMrkr(char m){
    mrkr=m;
}

void Player::setWins(int w){
    wins=w;
}

void Player::incWins(){
    wins+=1;
}

void Player::setLosses(int l){
    losses=l;
}

void Player::incLosses(){
    losses+=1;
}

//overloaded = operator
void Player::operator = (const Player &right){
    name=right.name;
    mrkr=right.mrkr;
    wins=right.wins;
    losses=right.losses;
}
```

computer.h

```
/*
 * File:   computer.h
 * Author: Joe
 *
 * Created on June 6, 2014, 4:07 PM
 */

#include <iostream>
#include <string>
#include "player.h"
using namespace std;

#ifndef COMPUTER_H
#define	COMPUTER_H

class Computer : public Player{
private:
    int diff;  //stores difficulty setting
public:
    //constructor
    Computer() : Player(){
        diff=1;
        name="Easy AI";
    }

    Computer(int d) : Player(){
        diff=d;
        if(diff==1)name="Easy AI";
        if(diff==2)name="Advanced AI";
    }
    int getDiff(){
        return diff;
    }
};

#endif /* COMPUTER_H */
```