***PROJECT NAME: TIC TAC TOE GAME***

***PROGRAMMIG FUNDAMENTAL***

GROUP MEMBERS:

1): ZAIN ZAHID (11515)

2): Syed Muneeb Hasan Kazmi (10937)

3): MUZAMIL KHAN (11515)

***CODING:***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace TicTacToeGame

{

class Program

{

public static char playerSignature = ' ';

static int turns = 0; //Will count each turn. Once == 10 then the game is a draw.

static char[] ArrBoard =

{

'1', '2', '3','4', '5', '6','7', '8', '9'

}; //Global char array variable to store the players input.

public static void BoardReset() //If this method is called then the game resets.

{

char[] ArrBoardInitialize =

{

'1', '2', '3','4', '5', '6','7', '8', '9'

};

ArrBoard = ArrBoardInitialize;

DrawBoard();

turns = 0;

}

public static void DrawBoard()

{

Console.Clear();

Console.WriteLine(" -------------------------");

Console.WriteLine(" | | | |");

Console.WriteLine(" | {0} | {1} | {2} |", ArrBoard[0], ArrBoard[1], ArrBoard[2]);

Console.WriteLine(" | | | |");

Console.WriteLine(" -------------------------");

Console.WriteLine(" | | | |");

Console.WriteLine(" | {0} | {1} | {2} |", ArrBoard[3], ArrBoard[4], ArrBoard[5]);

Console.WriteLine(" | | | |");

Console.WriteLine(" -------------------------");

Console.WriteLine(" | | | |");

Console.WriteLine(" | {0} | {1} | {2} |", ArrBoard[6], ArrBoard[7], ArrBoard[8]);

Console.WriteLine(" | | | |");

Console.WriteLine(" -------------------------");

} //Draws the player board to terminal.

public static void Introduction()

{

Console.Title = "MK TIC TAC TOE GAME(SIMPLE GAME)";

Console.BufferHeight = 100;

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\n\n\t\t\t\t\t(A SIMPLE GAME CREATED BY MK TEAM)");

Console.ResetColor();

Console.ForegroundColor = ConsoleColor.Cyan;

Console.WriteLine("\n\n\n\t\t\t\t\t WELCOME TO MK TIC TAC TOE GAME");

Console.WriteLine("\n\n\n\t\t\t\t\t PRESS ANY KEY TO START THE GAME");

Console.ResetColor();

Console.ReadKey();

Console.Clear();

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\n\n\t\t\t\t\t\tRULES OF GAME");

Console.ResetColor();

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("\n\n\n\t\t\tPlayers are represented with a unique signature" + "\n\t\t\tPlayer 1 = O. Player 2 = X");

Console.WriteLine("\n\t\t\tThe first player to score three signatures in a row is the winner" + "\n\t\t\tGood luck players!");

Console.ForegroundColor = ConsoleColor.Cyan;

Console.WriteLine("\n\n\n\t\t\t\t\tPRESS ANY KEY TO BEGIN THE GAME");

Console.ReadKey();

Console.ResetColor();

} //This method covers the game rules. Method setup in an effort to keep the code tidy.

static void Main(string[] args)

{

int player = 2; // Player 1 Starts

int input = 0;

bool inputCorrect = true;

Introduction();

do //Alternates player turns.

{

if (player == 2)

{

player = 1;

XorO(player, input);

}

else if (player == 1)

{

player = 2;

XorO(player, input);

}

DrawBoard();

turns++;

//Check Game Status.

HorizontalWin();

VerticalWin();

DiagonalWin();

if (turns == 10)

{

Draw();

}

do

{

Console.WriteLine("\nReady Player {0}: It's your move!", player);

try

{

input = Convert.ToInt32(Console.ReadLine());

}

catch

{

Console.WriteLine("Please enter a number!");

}

if ((input == 1) && (ArrBoard[0] == '1'))

inputCorrect = true;

else if ((input == 2) && (ArrBoard[1] == '2'))

inputCorrect = true;

else if ((input == 3) && (ArrBoard[2] == '3'))

inputCorrect = true;

else if ((input == 4) && (ArrBoard[3] == '4'))

inputCorrect = true;

else if ((input == 5) && (ArrBoard[4] == '5'))

inputCorrect = true;

else if ((input == 6) && (ArrBoard[5] == '6'))

inputCorrect = true;

else if ((input == 7) && (ArrBoard[6] == '7'))

inputCorrect = true;

else if ((input == 8) && (ArrBoard[7] == '8'))

inputCorrect = true;

else if ((input == 9) && (ArrBoard[8] == '9'))

inputCorrect = true;

else

{

Console.WriteLine("Whoops, I didn't get that. \nPlease try again...");

inputCorrect = false;

}

} while (!inputCorrect);

} while (true);

} //Gameplay loop. Controls player turns & overrides the array with players input.

public static void XorO(int player, int input)

{

if (player == 1) playerSignature = 'X';

else if (player == 2) playerSignature = 'O';

switch (input)

{

case 1: ArrBoard[0] = playerSignature; break;

case 2: ArrBoard[1] = playerSignature; break;

case 3: ArrBoard[2] = playerSignature; break;

case 4: ArrBoard[3] = playerSignature; break;

case 5: ArrBoard[4] = playerSignature; break;

case 6: ArrBoard[5] = playerSignature; break;

case 7: ArrBoard[6] = playerSignature; break;

case 8: ArrBoard[7] = playerSignature; break;

case 9: ArrBoard[8] = playerSignature; break;

}

} //Controls if the player is X or O.

public static void HorizontalWin()

{

char[] playerSignatures = { 'X', 'O' };

foreach (char playerSignatue in playerSignatures)

{

if (((ArrBoard[0] == playerSignatue) && (ArrBoard[1] == playerSignatue) && (ArrBoard[2] == playerSignatue))

|| ((ArrBoard[3] == playerSignatue) && (ArrBoard[4] == playerSignatue) && (ArrBoard[5] == playerSignatue))

|| ((ArrBoard[6] == playerSignatue) && (ArrBoard[7] == playerSignatue) && (ArrBoard[8] == playerSignatue)))

{

Console.Clear();

if (playerSignatue == 'X')

{

Console.WriteLine("Congratulations Player 1.\nYou have a achieved a horizontal win! " +

"\nYou're the Tic Tac Toe Master!\n" +

"\nTurns taken{0}", turns);

}

else if (playerSignatue == 'O')

{

Console.WriteLine("Congratulations Player 2.\nYou have a achieved a horizontal win! " +

"\nYou're the Tic Tac Toe Master!\n" +

"\nTurns taken{0}", turns);

}

WinArt();

Console.WriteLine("Please press any key to reset the game");

Console.ReadKey();

BoardReset();

break;

}

}

} //Method is called to check for a horizontal win.

public static void VerticalWin()

{

char[] playerSignatures = { 'X', 'O' };

foreach (char playerSignatue in playerSignatures)

{

if (((ArrBoard[0] == playerSignatue) && (ArrBoard[3] == playerSignatue) && (ArrBoard[6] == playerSignatue))

|| ((ArrBoard[1] == playerSignatue) && (ArrBoard[4] == playerSignatue) && (ArrBoard[7] == playerSignatue))

|| ((ArrBoard[2] == playerSignatue) && (ArrBoard[5] == playerSignatue) && (ArrBoard[8] == playerSignatue)))

{

Console.Clear();

if (playerSignatue == 'X')

{

Console.WriteLine("Player 1, that was Fantastic.\nA vertical win!\nYou're the Tic Tac Toe Master!\n");

}

else

{

Console.WriteLine("Player 2, that was Fantastic.\nA vertical win!\nYou're the Tic Tac Toe Master!\n");

}

WinArt();

Console.WriteLine("Please press any key to reset the game");

Console.ReadKey();

BoardReset();

break;

}

}

} //Method is called to check for a vertical win.

public static void DiagonalWin()

{

char[] playerSignatures = { 'X', 'O' };

foreach (char playerSignatue in playerSignatures)

{

if (((ArrBoard[0] == playerSignatue) && (ArrBoard[4] == playerSignatue) && (ArrBoard[8] == playerSignatue))

|| ((ArrBoard[6] == playerSignatue) && (ArrBoard[4] == playerSignatue) && (ArrBoard[2] == playerSignatue)))

{

Console.Clear();

if (playerSignatue == 'X')

{

Console.WriteLine("WOW!, player 1 that's a diagonal win! " +

"\nExcellently played, it's one for the ages! " +

"\nYou're the Tic Tac Toe Legend!\n \n \n");

}

else

{

Console.WriteLine("WOW!, player 2 that's a diagonal win! " +

"\nExcellently played, it's one for the ages! " +

"\nYou're the Tic Tac Toe Legend!\n \n \n");

}

WinArt();

Console.WriteLine("Please press any key to reset the game");

Console.ReadKey();

BoardReset();

break;

}

}

} //Method is called to check for a diagonal win.

public static void Draw()

{

{

Console.WriteLine("Aw gosh... it's a draw." +

"\nPlease press any key to reset the game and try again!");

Console.ReadKey();

BoardReset();

}

} //Method is called to check if the game is a draw.

public static void WinArt()

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Congratulations You Won !!!!!!");

Console.ResetColor();

} //ASCII Art setup in it's own method to help keep the code tidy.

}

}

***Screenshot Of Output:***

