B.Tic-Tac-Toe Game

#include<stdio.h>

#include<conio.h>

char square[3][3] = {

{'1','2','3'},

{'4','5','6'},

{'7','8','9'}

};

int board[3][3] = {

{2,2,2},

{2,2,2},

{2,2,2}

};

int player = 0, i ,choice , isEvil = 0 , isEvilMarked = 0 , reset = 0 , flag = 0, withPlayer = 0;

char mark;

void resetBoard();

int checkWinner();

void drawBoard();

void selectPlayer();

void enterChoice();

void selectMark();

void fillMark();

void gameStatus();

int isPlaceEmpty();

void findComputerMove()

{

int row,col,selectedRow , selectedCol,i,j , multi = 1, maxMul =1;

for(row =0 ; row <3 ; row++)

{

for(col =0; col <3; col++)

{

// reset

// maxMul = 1;

if(board[row][col] == 2)

{

board[row][col] = 3;

multi = board[row][0]\* board[row][1]\*board[row][2];

// Computer Win Condition or Saving opponent win

if( multi == 27)

{

square[row][col] = 'X';

return;

} else if(maxMul < multi)

{

maxMul = multi;

selectedRow = row;

selectedCol = col;

}

multi = board[0][col]\* board[1][col]\*board[2][col];

// Computer Win Condition or Saving opponent win

if( multi == 27)

{

square[row][col] = 'X';

return;

} else if(maxMul < multi)

{

maxMul = multi;

selectedRow = row;

selectedCol = col;

}

if(row == col)

{

multi = board[0][0]\* board[1][1]\*board[2][2];

// Computer Win Condition or Saving opponent win

if( multi == 27)

{

square[row][col] = 'X';

return;

}else if(maxMul < multi)

{

maxMul = multi;

selectedRow = row;

selectedCol = col;

}

}

if((row ==0 && col ==2) || (row ==2 && col ==0) || (row == 1 && col ==1))

{

multi = board[0][2]\* board[1][1]\*board[2][0];

// Computer Win Condition or Saving opponent win

if( multi == 27)

{

square[row][col] = 'X';

return;

} else if(maxMul < multi)

{

maxMul = multi;

selectedRow = row;

selectedCol = col;

}

}

board[row][col] = 2;

}

}

}

// check opponent win

for(row =0 ; row <3 ; row++)

{

for(col =0; col <3; col++)

{

if(board[row][col] == 2)

{

board[row][col] = 3;

multi = board[row][0]\* board[row][1]\*board[row][2];

// Computer Win Condition or Saving opponent win

if(multi == 3)

{

reset = 1;

square[row][col] = 'X';

if(isEvil == 0)

return;

else

isEvilMarked = 1;

}

multi = board[0][col]\* board[1][col]\*board[2][col];

if(multi == 3)

{ reset = 1;

square[row][col] = 'X';

if(isEvil == 0)

return;

else

isEvilMarked = 1;

}

if(row == col)

{

multi = board[0][0]\* board[1][1]\*board[2][2];

if( multi == 3)

{ reset = 1;

square[row][col] = 'X';

if(isEvil == 0)

return;

else

isEvilMarked = 1;

}

}

if((row ==0 && col ==2) || (row ==2 && col ==0) || (row == 1 && col ==1))

{

multi = board[0][2]\* board[1][1]\*board[2][0];

if(multi == 3)

{

reset = 1;

square[row][col] = 'X';

if(isEvil == 0)

return;

else

isEvilMarked = 1;

}

}

if(isEvilMarked == 0)

board[row][col] = 2;

isEvilMarked = 0;

}

}

}

if(reset == 0)

{

board[selectedRow][selectedCol] = 3;

square[selectedRow][selectedCol] = 'X';

}

else

reset = 0;

}

int isPlaceEmpty(int row, int col)

{

if(square[row][col] != 'X' && square[row][col] !='O')

return 1;

else

return 0;

}

void enterChoice()

{

printf("Player %d, Enter Marking Place : ",player);

fflush(stdin);

scanf("%d",&choice);

fflush(stdin);

}

void selectPlayer()

{

player = !player;

}

void selectMark()

{

mark = (player == 1) ? 'O' : 'X';

}

void gameStatus()

{

if(i==1){

printf("Player %d won",--player);

}

else {

printf("Game draw");

}

printf("\n\n#####################################################\n\n");

}

int checkWinner(){

// Checking Horizontal match

int row = 0 , col = 0 , placeFilled = 1;

for(row = 0 ; row < 3 ; row++)

{

if(square[row][0] == square[row][1] && square[row][1] == square[row][2])

return 1;

}

// checking vertical match

for(col = 0 ; col < 3 ; col++)

{

if(square[0][col] == square[1][col] && square[1][col] == square[2][col])

return 1;

}

// Checking diagonal match

if(square[0][0] == square[1][1] && square[1][1] == square[2][2])

return 1;

else if(square[0][2] == square[1][1] && square[1][1] == square[2][0])

return 1;

// checking that no place is vacant now

for(row = 0 ; row < 3 ; row++)

{

for(col = 0 ; col < 3 ; col++)

{

if(isPlaceEmpty(row,col))

{

placeFilled = 0;

}

}

}

// placeFilled == 0 means board not filled completely

// Return 0 if board is filled completely else return -1

if(placeFilled == 1)

return 0;

else

return -1;

}

void drawBoard(){

// Clearing Screen for re-drawing new board

int row = 0;

system("cls");

printf("\n\n#################### Tic Tac Toe ####################\n\n");

printf("First Player X \nSecond Player O \n\n\n");

for(row = 0 ; row < 3 ; row++)

{

printf(" | | \n");

printf(" %c | %c | %c \n",square[row][0],square[row][1],square[row][2]);

printf("\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_\n");

}

printf("\n\n#####################################################\n\n");

}

void fillMark()

{

int row = 0 , col = 0;

int place = 0;

int flag = 0;

for(row = 0 ; row < 3 ; row++)

{

for(col = 0 ; col < 3 ; col++)

{

// Formula used to find the place.

// How this formula comes will be explained in class

place = ((row \* 3) + (col +1));

// checking the choice and place is empty or not

if((choice == place) && isPlaceEmpty(row,col))

{

square[row][col] = mark;

board[row][col] = 1;

flag = 1; // Mark done

}

}

}

// Mark not done means Invalid option

if(flag == 0)

{

printf("\n\n#####################################################\n\n");

printf("Place is Either already used or incorrect !");

getch();

if(withPlayer == 0)

player--;

}else{

i = checkWinner();

if(withPlayer == 0)

player++;

else

selectPlayer();

}

}

void resetBoard()

{

int row, col;

char a = '1';

for(row = 0 ; row < 3 ; row++)

{

for(col = 0 ; col < 3 ; col++)

{

square[row][col] = a++;

board[row][col] = 2;

}

}

player = 0,isEvil = 0 , isEvilMarked = 0 , reset = 0 , flag = 0, withPlayer = 0;

}

int main(){

int choice ;

while(1)

{

system("cls");

resetBoard();

printf("Enter your choice\n");

printf("1. Want to play with Smart Computer\n");

printf("2. Want to play with Evil Computer\n");

printf("3. Want to play with player\n");

printf("4. Exit\n");

scanf("%d",&choice);

switch(choice)

{

case 1 :

case 2 : isEvil = (choice == 1 ) ? 0 : 1;

do {

drawBoard();

selectPlayer();

if(player == 0)

{

findComputerMove();

i = checkWinner();

if(i == 1)

player++;

}else{

enterChoice();

selectMark();

fillMark();

}

}while(i == -1);

drawBoard();

gameStatus();

getchar();

break;

case 3 :

withPlayer = 1;

selectPlayer();

selectMark();

do {

drawBoard();

selectMark();

enterChoice();

fillMark();

}while(i == -1);

drawBoard();

gameStatus();

getchar();

break;

case 4 :

exit(0);

break;

default: printf("Invalid choice\n");

}

}

return 0;

}

**OUTPUT**









