“Tic Tac Toe”

In c language

Ansari Arif(3117001)

Memon Shaziya(3117026)

Mishra Saloni(3117027)

Second Year Computer Engineering(Sem-IV)

Subject: Computer Graphics(CG)

Prof. Lutful Islam

Department of Computer Engineering

M. H. Saboo Siddik College of Engineering

8,Saboo Siddik Polytechnic Road, New Nagpada, Byculla,

Mumbai,Maharashtra 400008

2018

Index

Sr. No.	Topic
1.	Problem Statement
2.	Theory
3.	Explanation
4.	Program
5.	Output

Problem Statement

Implementation of Tic-Tac-Toe game in c language using graphics.

Theory

Introduction

TIC-TAC-TOE is the most popular and easiest game. It is a two players (X and O) game, where each player taking turns marks the spaces in a 3×3 grid. I have developed this game in TurboC graphics with interfacing keyboard event and have also shown some interactive animation. This tip will be helpful for anyone who wants to develop this type of game using TurboC graphics. So let's get started...

History:

Games played on three-in-a-row boards can be traced back to ancient Egypt,[5] where such game boards have been found on roofing tiles dating from around 1300 BCE.[6]

An early variation of tic-tac-toe was played in the Roman Empire, around the first century BC. It was called *terni lapilli* (*three pebbles at a time*) and instead of having any number of pieces, each player only had three, thus they had to move them around to empty spaces to keep playing.[7] The game's grid markings have been found chalked all over Rome.

Another closely related ancient game is Three Men's Morris which is also played on a simple grid and requires three pieces in a row to finish,[8] and Picaria, a game of the Puebloans.

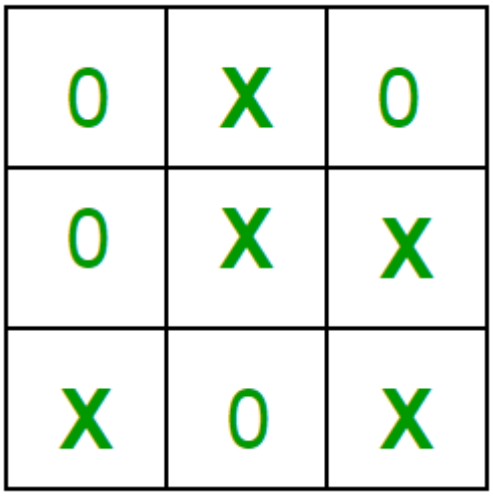
The different names of the game are more recent. The first print reference to "noughts and crosses" (nought being an alternative word for zero), the British name, appeared in 1858, in an issue of *Notes and Queries*.[9] The first print reference to a game called "tick-tack-toe" occurred in 1884, but referred to "a children's game played on a slate, consisting in trying with the eyes shut to bring the pencil down on one of the numbers of a set, the number hit being scored". "Tic-tac-toe" may also derive from "tick-tack", the name of an old version of backgammon first described in 1558. The US renaming of "noughts and crosses" as "tic-tac-toe" occurred in the 20th century.[10]

In 1952, *OXO* (or *Noughts and Crosses*), developed by British computer scientist Alexander S. Douglas for the EDSAC computer at the University of Cambridge, became one of the first known video games.[11][12] The computer player could play perfect games of tic-tac-toe against a human opponent.[11]

In 1975, tic-tac-toe was also used by MIT students to demonstrate the computational power of Tinkertoy elements. The Tinkertoy computer, made out of (almost) only Tinkertoys, is able to play tic-tac-toe perfectly.[13] It is currently on display at the Museum of Science, Boston.

Rules of the Game

- The game is to be played between two people (in this program between HUMAN and COMPUTER).
- One of the player chooses ‘O’ and the other ‘X’ to mark their respective cells.
- The game starts with one of the players and the game ends when one of the players has one whole row/ column/ diagonal filled with his/her respective character (‘O’ or ‘X’).
- If no one wins, then the game is said to be draw.



Implementation

In our program the moves taken by the computer and the human are chosen randomly. We use rand() function for this.

What more can be done in the program?

The program is in not played optimally by both sides because the moves are chosen randomly. The program can be easily modified

so that both players play optimally (which will fall under the category of Artificial Intelligence). Also the program can be modified such that the user himself gives the input (using scanf() or cin).

The above changes are left as an exercise to the readers.

Winning Strategy – An Interesting Fact

If both the players play optimally then it is destined that you will never lose ("although the match can still be drawn"). It doesn't matter whether you play first or second.In another ways – " Two expert players will always draw ".

Explanation

While making a Tic Tac Toe game using C language, it is important to make use of arrays. The Xs and Os are kept in different arrays, and they are passed between several functions in the code to keep track of how the game goes. With the code here you can play the game choosing either X or O against the computer.

This has been done this way because it is just a console application without graphics designed in C language. The gotoxy function has been used to print text in any part of the screen.

I have divided this project into many functions, and below is a list of those functions. I have only described the gotoxy function

in detail. Just go through the source code once, and other functions used are simple and easy to understand.

1. **void gotoxy (int x, int y)** : You need to understand this function as it is one of the most important one used in Tic Tac Toe in C. This function allows you to print text in any place of the screen. It can be directly used in Turbo C.
2. **void logic():** This function is used to print 'X' or 'O' according to the user.
3. **void makex(int,int):** This function is used to print the 'X' in the respective cell according to the coordinate.
4. **void makeO(int,int):** This function is used to print the 'O' in the respective cell according to the coordinate.

5. **int check(int,int,int,int,int,int,int,int,int):** This function is used to check the winning condition, if one whole row/ column/ diagonal filled with his/her respective character ('O' or 'X').

Requirements

To run and develop TurboC graphics code, you need the following:

1. TurboC compiler to execute and run the code.
2. Graphic mode functions require a graphics monitor and adapter card such as CGA, EGA and VGA.

3. To start graphics programming, you need two files which are *GRAPHICS.H* and *GRAPHICS.LIB*. Especially, these files are provided as part of TurboC compile.

Program:

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
#include "DDA~1.CPP"
#include "CIRCLE.CPP"
#include <stdlib.h>
void logic();
void makex(int,int);
void make0(int,int);
int check(int,int,int,int,int,int,int,int,int);
int count;
void main()
{
int gdriver=DETECT,gmode;
initgraph(&gdriver,&gmode,"C:\\TURBOC3\\BGI");
DDALine(100,20,100,350,WHITE);
DDALine(200,20,200,350,WHITE);
DDALine(10,100,300,100,WHITE);
DDALine(10,250,300,250,WHITE);
logic();
getch();

}
void makex(int x,int y)
{
        int x1=x,y1=y;
        DDALine(x1-20,y1-20,x1,y1,RED);
        DDALine(x1,y1,x1+20,y1+20,RED);
        DDALine(x1,y1,x1-20,y1+20,RED);
        DDALine(x1+20,y1-20,x1,y1,RED);
}
void make0(int x,int y)
{
        int xc=x,yc=y;
        midpoint(xc,yc,25);
}

void logic()
{
int win=0,play=0;
int entered[10],q,w;
int n,turn=0,v,ans=0;
int b1=0,b2=0,b3=0,b4=0,b5=0,b6=0,b7=0,b8=0,b9=0;
int x=350,y=250;
for(q=0;q<=10;q++)
```

```
{
entered[q]=0;
}
while(win==0)
{
		printf("\n\t\t\tEnter the block");
while (play==0)
{
		scanf("%d",&n);
		if(n==1||n==2||n==3||n==4||n==5||n==6||n==7||n==8||n==9)
		{
		if(turn==0||turn==2||turn==4||turn==6||turn==8)
		{
		if(entered[n]==0)
		{
		 entered[n]=1;
		switch(n)
		{
		case 1:
		makex(45,50);
		b1=1;
		count++;
		turn++  ;
		break;

		case 2:
		makex(150,50);
		b2=1;
		count++;
		turn++   ;
		break;

		case 3:
		makex(250,50);
		b3=1;
		count++;
		turn++;
		break;

		case 4:
		makex(45,175);
		b4=1;
		count++;
		turn++;
		break;

		case 5:
		makex(150,175);
		b5=1;
		count++;
		turn++;
		break;

		case 6:
		makex(250,175);
		b6=1;
		count++;
		turn++;
```

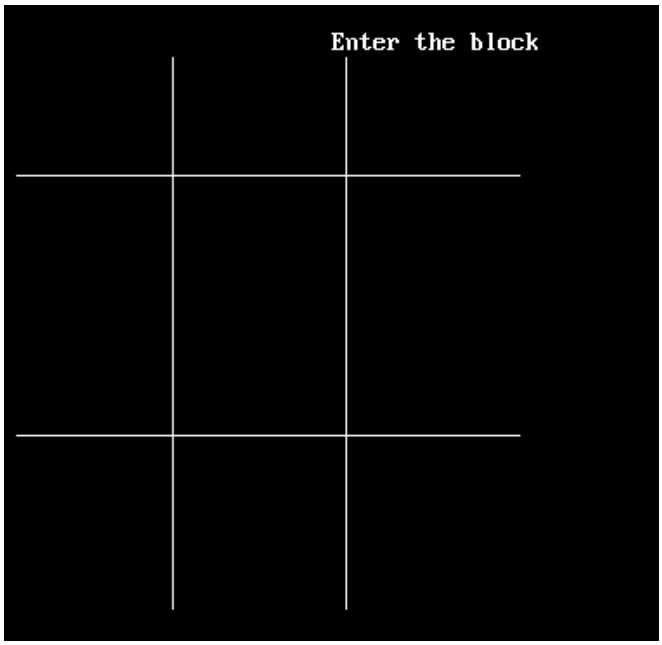
```
break;

case 7:
makex(45,300);
b7=1;
count++;
turn++;
break;

case 8:
makex(150,300);
b8=1;
count++;
turn++;
break;

case 9:
makex(250,300);
b9=1;
count++;
turn++;
break;

default:
break;
}
ans=check(b1,b2,b3,b4,b5,b6,b7,b8,b9);
}
}
else
{
if(entered[n]==0)
{
entered[n]=1;
 switch(n)
 {
 case 1:
makeO(45,50);
b1=2;
count++;
turn++ ;
break;

case 2:
makeO(150,50);
b2=2;
count++;
turn++;
break;

case 3:
makeO(250,50);
b3=2;
count++;
turn++;
break;

case 4:
```

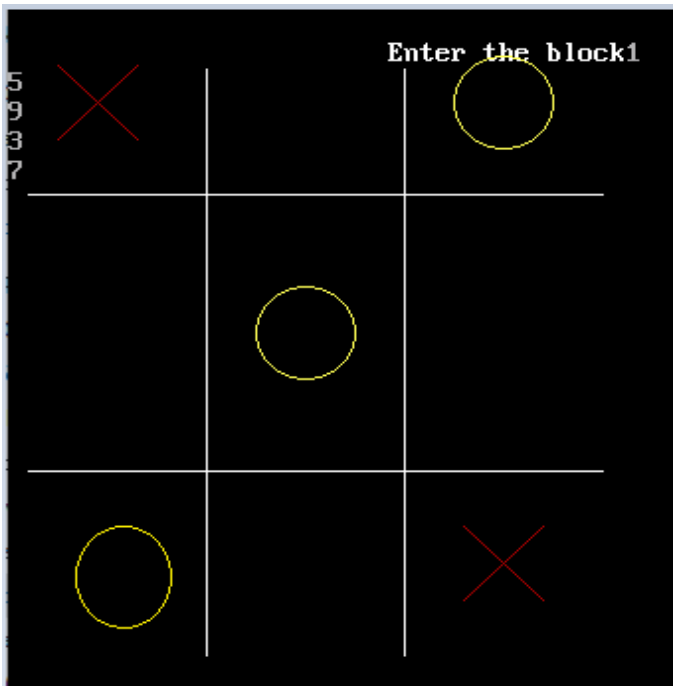
```
make0(45,175);
b4=2;
count++;
turn++;
break;

case 5:
make0(150,175);
b5=2;
count++;
turn++;
break;

case 6:
make0(250,175);
b6=2;
count++;
turn++;
break;

case 7:
make0(45,300);
b7=2;
count++;
turn++;
break;

case 8:
make0(150,300);
b8=2;
count++;
turn++;
break;

case 9:
make0(250,300);
b9=2;
count++;
turn++;
break;

default:
break;
}
ans=check(b1,b2,b3,b4,b5,b6,b7,b8,b9);
}
}
}

else
{
outtextxy("You've entered the wrong block");
y=y+50;

}
        if(count==9||count==18||count==27||count==36)
{
```

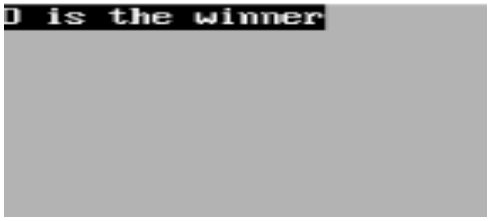
```
main();
 }
                if(ans==1)
                {
                clrscr();
                printf("X is the winner");
                count=9;
//              logic();
delay(1000);
main();
                win=1;
                }
                if(ans==2)
                {
                clrscr();
                printf("O is the winner");
                count=9;
      //        logic();
      delay(1000);
      main();
                win=1;
                }
                 }
 }
// clrscr();
 printf("Press 1 to continue 0 to exit");
 scanf("%d",&v);
 if(v==1)
 {
 play=0;
 }
 else
 {
 play=1;
 }
 //turn++;
 }

 int check(int b1,int  b2,int  b3 ,int  b4,int b5,int b6,int b7,int b8,int b9)
{ if(b1==1&&b2==1&&b3==1)
 return 1;
 if(b1==1&&b4==1&&b7==1)
 return 1;
 if(b2==1&&b5==1&&b6==1)
 return 1;
if(b3==1&&b6==1&&b9==1)
return 1;
if(b4==1&&b5==1&&b6==1)
return 1;
if(b7==1&&b8==1&&b9==1)
return 1;
if(b1==1&&b5==1&&b9==1)
return 1;
if(b3==1&&b5==1&&b7==1)
return 1;

 if(b1==2&&b2==2&&b3==2)
```

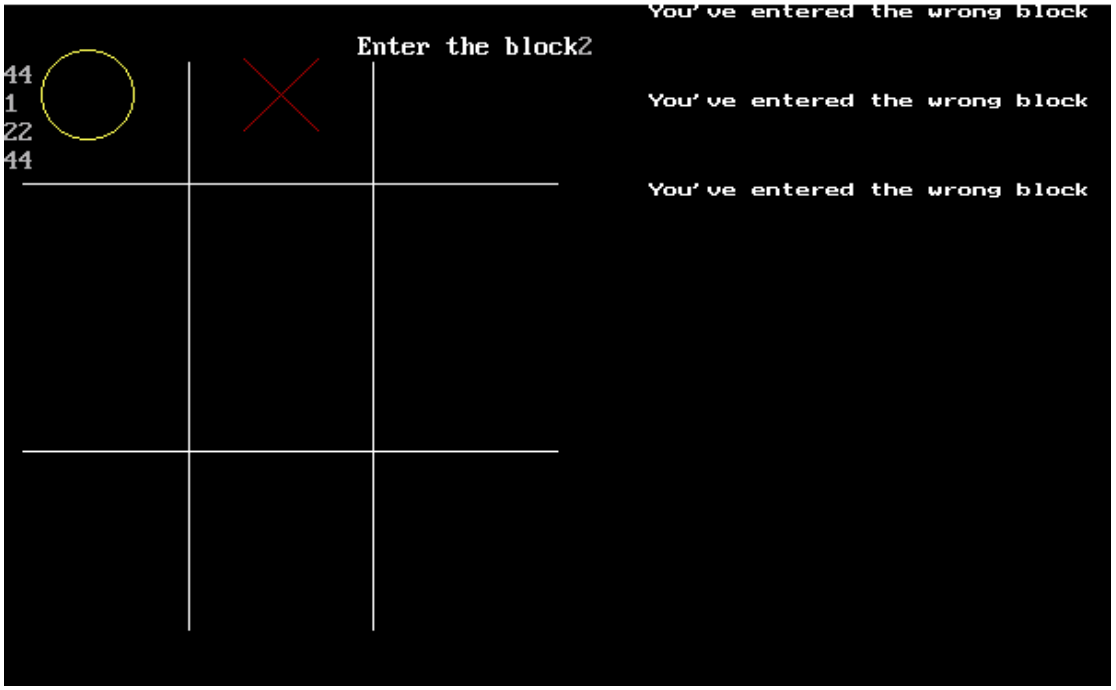
```
 return 2;
if(b1==2&&b4==2&&b7==2)
 return 2;
if(b2==2&&b5==2&&b6==2)
 return 2;
if(b3==2&&b6==2&&b9==2)
return 2;
if(b4==2&&b5==2&&b6==2)
return 2;
if(b7==2&&b8==2&&b9==2)
return 2;
if(b1==2&&b5==2&&b9==2)
return 2;
if(b3==2&&b5==2&&b7==2)
return 2;
return 3;
}
```

Output:




Enter the block1
5
9
3
7


O is the winner


You've entered the wrong block
Enter the block2
44
1
22
44
You've entered the wrong block
You've entered the wrong block