**MINI PROJECT (GAME DESIGN)**

**TIC TAC TOE**

**PROBLEM STATEMENT:**

Design a game OR a small application using appropriate Data Structure.

**DESIGN**

Students can form a group of 3 or 4. Students can choose any system, for example library data, medical shop data,

College admission data etc.

Students can list multiple functions associated with problem statement.

One problem statement can be divided into multiple problem statement modules (functions) students can list

Input and output required for each function.

Any data structure can be used.

Students may also design a game using any data structure.

**DELIVERABLES:**

1. Problem Statement: Design the game TIC TAC TOE using array data structure.

2. Details About Designing: Classes-- only a single main class was used by us.

Functions-- we used functions 1. Board to draw the tic tac toe board, 2.TogglePlayer to alternate play chance for X and O, 3.Input functions to take the positions of X and O and 4.Score function to reveal the winner or to declare a Tie.

3. Game Design in Brief:

This is a two player game where one player chooses X and the other chooses O. There are 9 blocks in total.

Input position of X and O is taken from respective Player.

When Vertical/Horizontal/Diagonal places appear with the same sign then the Score is Revealed (i.e. x wins or O wins) else a TIE is declared.

4. Short Note On the Data Structure Used:

We used 2-DIMENSIONAL ARRAY

Definition-Multi-dimensional array is an array that is more than just a single row of elements. It may have rows and columns (a two-dimensional array or 2D array), or may have three dimensions. All elements must be of the same data type. Rows and Columns are included in a 2D array. We know for this game 9 positions are needed hence Hard coded values from 1 to 9 is given to the array.

There are only 9 positions at which we can enter data, hence we went on replacing the hardcoded positions with X and O.

***CPP CODE:***

#include<iostream>

#include<conio.h>

using namespace std;

char matrix[3][3]={'1','2','3','4','5','6','7','8','9'};

int score;

int count=0;

char player='X';

void Board() //This function creates our game Board.

{

cout<<"\n\n\n\t\t\t\tTIC TAC TOE"<<endl;

for(int i=0;i<3;i++)

{

for(int j=0;j<3;j++)

{

cout<<" | "<<matrix[i][j];

}

cout<<endl;

}

}

void Input() //This function takes the input for X and O.

{

int a;

cout<<"\n\tPRESS THE NUMBER FOR THE FIELD 1 TO 9:";

cin>>a;

switch(a)

{

case 1:matrix[0][0]=player;

count++;

break;

case 2:matrix[0][1]=player;

count++;

break;

case 3:matrix[0][2]=player;

count++;

break;

case 4:matrix[1][0]=player;

count++;

break;

case 5:matrix[1][1]=player;

count++;

break;

case 6:matrix[1][2]=player;

count++;

break;

case 7:matrix[2][0]=player;

count++;

break;

case 8:matrix[2][1]=player;

count++;

break;

case 9:matrix[2][2]=player;

count++;

break;

}

}

int Score()

{

if(matrix[0][0]=='X' && matrix[0][1]=='X' && matrix[0][2]=='X')

return 2;

if(matrix[1][0]=='X' && matrix[1][1]=='X' && matrix[1][2]=='X')

return 2;

if(matrix[2][0]=='X' && matrix[2][1]=='X' && matrix[2][2]=='X')

return 2;

if(matrix[0][0]=='X' && matrix[1][0]=='X' && matrix[2][0]=='X')

return 2;

if(matrix[0][1]=='X' && matrix[1][1]=='X' && matrix[2][1]=='X')

return 2;

if(matrix[0][2]=='X' && matrix[2][1]=='X' && matrix[2][2]=='X')

return 2;

if(matrix[0][0]=='X' && matrix[1][1]=='X' && matrix[2][2]=='X')

return 2;

if(matrix[0][2]=='X' && matrix[1][1]=='X' && matrix[2][0]=='X')

return 2; //Returns 2 if any of the above conditions are true for X to win

if(matrix[0][0]=='O' && matrix[0][1]=='O' && matrix[0][2]=='O')

return 4;

if(matrix[1][0]=='O' && matrix[1][1]=='O' && matrix[1][2]=='O')

return 4;

if(matrix[2][0]=='O' && matrix[2][1]=='O' && matrix[2][2]=='O')

return 4;

if(matrix[0][0]=='O' && matrix[1][0]=='O' && matrix[2][0]=='O')

return 4;

if(matrix[0][1]=='O' && matrix[1][1]=='O' && matrix[2][1]=='O')

return 4;

if(matrix[0][2]=='O' && matrix[2][1]=='O' && matrix[2][2]=='O')

return 4

if(matrix[0][0]=='O' && matrix[1][1]=='O' && matrix[2][2]=='O')

return 4;

if(matrix[0][2]=='O' && matrix[1][1]=='O' && matrix[2][0]=='O')

return 4; ;//Returns 4 if any of the above conditions are true for O to win

}

void TogglePlayer() //This functions ensures alternate play for X and O.

{

if(player=='X')

{

player='O';

}

else

{

player='X';

}

}

int main()

{

int n;

clrscr;

Board();

while(1)

{

Input();

Board();

n=Score();

if(n==2)

{

cout<<"\n\t\*\*\*\*\*X WINS!\*\*\*\*\*";

break;

}

if(n==4)

{

cout<<"\n\t\*\*\*\*\*O WINS!\*\*\*\*\*";

break;

}

if(count>=9)

{

cout<<"\n\t\*\*\*\*\*MATCH TIED!\*\*\*\*\*";

}

TogglePlayer();

}

return 0;

}

END OF THE CODE.

OUTPUT

TIC TAC TOE

| 1 | 2 | 3

| 4 | 5 | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:1

TIC TAC TOE

| X | 2 | 3

| 4 | 5 | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:2

TIC TAC TOE

| X | O | 3

| 4 | 5 | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:3

TIC TAC TOE

| X | O | X

| 4 | 5 | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:4

TIC TAC TOE

| X | O | X

| O | 5 | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:5

TIC TAC TOE

| X | O | X

| O | X | 6

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:6

TIC TAC TOE

| X | O | X

| O | X | O

| 7 | 8 | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:8

TIC TAC TOE

| X | O | X

| O | X | O

| 7 | X | 9

PRESS THE NUMBER FOR THE FIELD 1 TO 9:9

TIC TAC TOE

| X | O | X

| O | X | O

| 7 | X | O

PRESS THE NUMBER FOR THE FIELD 1 TO 9:7

TIC TAC TOE

| X | O | X

| O | X | O

| X | X | O

\*\*\*\*\*X WINS!\*\*\*\*\*