

Tic-Tac-Toe:

Description:

*My project name is Tic-Tac-Toe game. This game is very popular and is fairly simple. In this game, there is a board with $n * n$ squares. In my game, it is $3 * 3$ squares. The game can be played by two players.*

(1) gamer 1 (2) gamer 2

*The goal of Tic-Tac-Toe game is to be one of the gamers to get three same symbols in a row-horizontally, vertically or diagonally on a $3*3$ grid.*

Requirements:

High level Requirements:

- * players should know the rules of the game.*
- * create a $3 * 3$ grid board.*
- * Declare the winner.*

Low level Requirements:

- * Knowing the game rules*

i) If any gamer is able to draw three Xs or three Ys in the following combinations then that player wins. The combinations are:

- | | |
|-----------------|-----------------|
| <i>a) 1,2,3</i> | <i>e) 2,5,8</i> |
| <i>b) 4,5,6</i> | <i>f) 3,6,9</i> |
| <i>c) 7,8,9</i> | <i>h) 1,5,9</i> |

d)1,4,7

i)3,5,7

** creating a 3 * 3 grid board*

*i)we have to create a 3 * 3 grid board for the game for that I used function grid.*

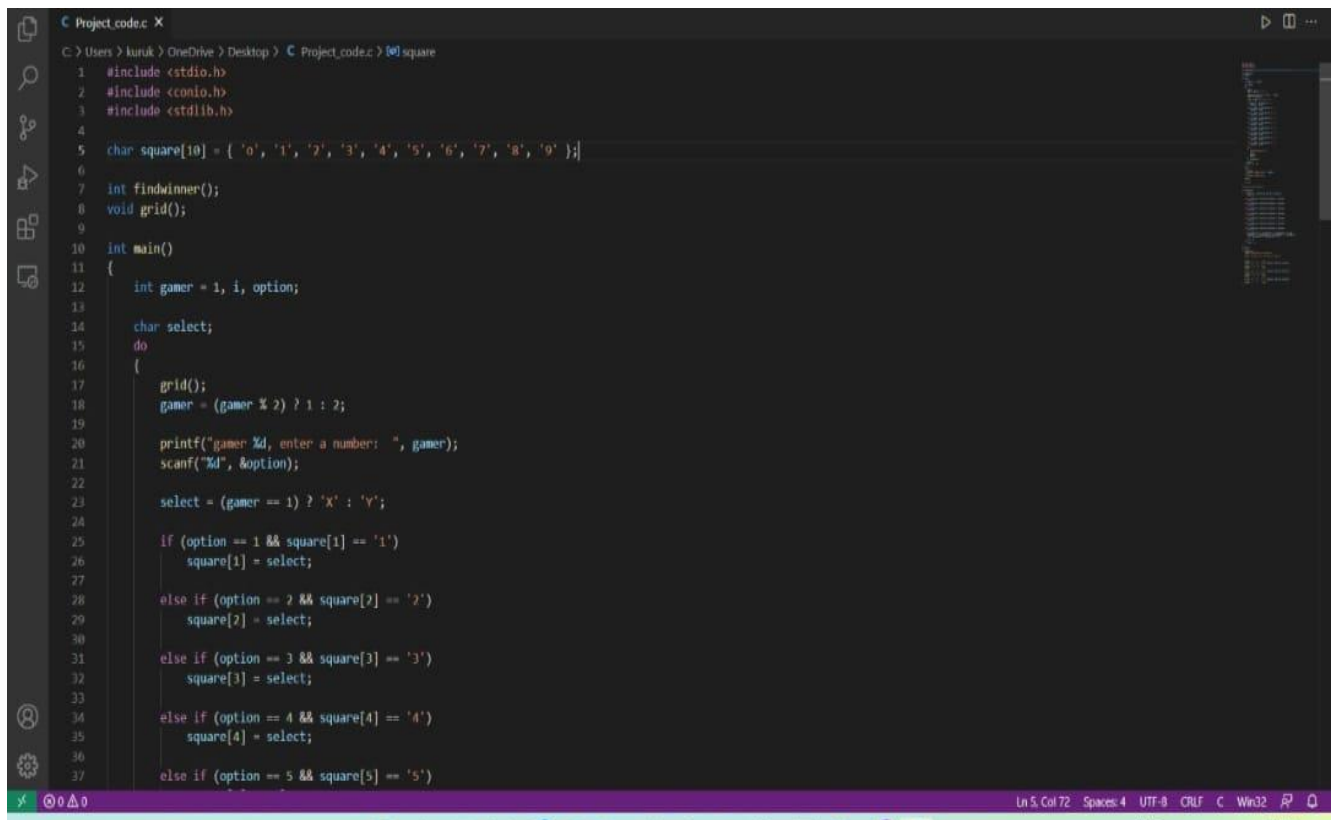
*ii) Nine squares were created as we used 3 * 3 grid and we have to name the squares from 1-9.*

** Declaring the winner*

i)The player who draw the three Xs or three Ys in the above combinations is declared as winner.

ii)TO check winner I used findwinner function.

Source Code:



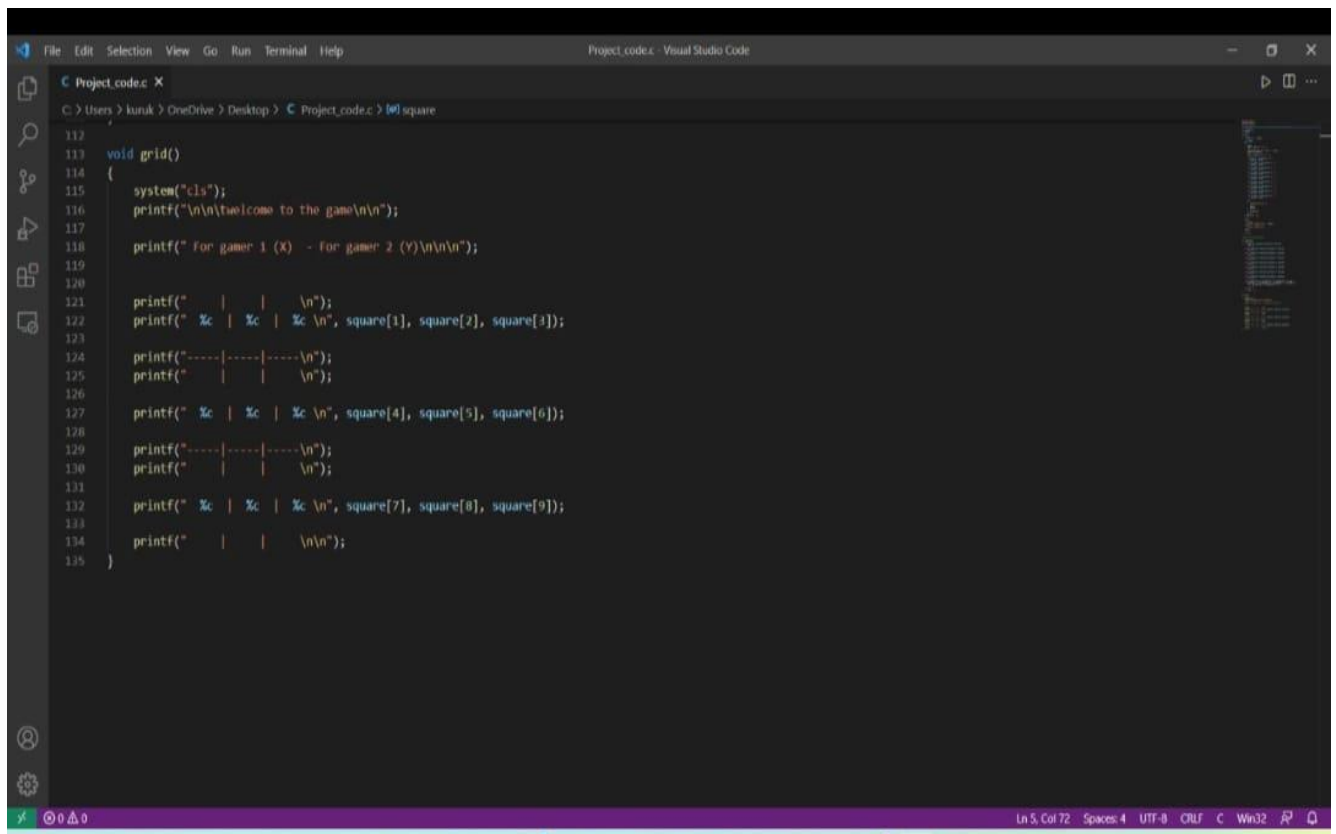
```
C Project_code.c X
C:\Users\kuruk>OneDrive\Desktop>C:\Project_code.c>[0] square
1  #include <stdio.h>
2  #include <conio.h>
3  #include <stdlib.h>
4
5  char square[10] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };
6
7  int findwinner();
8  void grid();
9
10 int main()
11 {
12     int gamer = 1, i, option;
13
14     char select;
15     do
16     {
17         grid();
18         gamer = (gamer % 2) ? 1 : 2;
19
20         printf("gamer %d, enter a number: ", gamer);
21         scanf("%d", &option);
22
23         select = (gamer == 1) ? 'X' : 'Y';
24
25         if (option == 1 && square[1] == '1')
26             square[1] = select;
27
28         else if (option == 2 && square[2] == '2')
29             square[2] = select;
30
31         else if (option == 3 && square[3] == '3')
32             square[3] = select;
33
34         else if (option == 4 && square[4] == '4')
35             square[4] = select;
36
37         else if (option == 5 && square[5] == '5')
```

```
File Edit Selection View Go Run Terminal Help
Project_code.c - Visual Studio Code

C Project_code.c X
C:\Users\kuruk> OneDrive\ Desktop> C:\Project_code.c> 90 square
37     else if (option == 5 && square[5] == '5')
38         square[5] = select;
39
40     else if (option == 6 && square[6] == '6')
41         square[6] = select;
42
43     else if (option == 7 && square[7] == '7')
44         square[7] = select;
45
46     else if (option == 8 && square[8] == '8')
47         square[8] = select;
48
49     else if (option == 9 && square[9] == '9')
50         square[9] = select;
51
52     else
53     {
54         printf("Invalid move ");
55
56         gamer--;
57         getch();
58     }
59     i = findwinner();
60
61     gamer++;
62 } while (i == - 1);
63
64 grid();
65
66 if (i == 1)
67     printf("==>\agamer %d win ", --gamer);
68 else
69     printf("==>\aGame draw");
70
71     getch();
72
73     return 0;
```

```
File Edit Selection View Go Run Terminal Help
Project_code.c - Visual Studio Code

C Project_code.c X
C:\Users\kuruk> OneDrive\ Desktop> C:\Project_code.c> 90 square
76 //FUNCTION TO RETURN GAME STATUS
77
78 int findwinner()
79 {
80     if (square[1] == square[2] && square[2] == square[3])
81         return 1;
82
83     else if (square[4] == square[5] && square[5] == square[6])
84         return 1;
85
86     else if (square[7] == square[8] && square[8] == square[9])
87         return 1;
88
89     else if (square[1] == square[4] && square[4] == square[7])
90         return 1;
91
92     else if (square[2] == square[5] && square[5] == square[8])
93         return 1;
94
95     else if (square[3] == square[6] && square[6] == square[9])
96         return 1;
97
98     else if (square[1] == square[5] && square[5] == square[9])
99         return 1;
100
101     else if (square[3] == square[5] && square[5] == square[7])
102         return 1;
103
104     else if (square[1] != '1' && square[2] != '2' && square[3] != '3' &&
105             square[4] != '4' && square[5] != '5' && square[6] != '6' && square[7]
106             != '7' && square[8] != '8' && square[9] != '9')
107
108         return 0;
109     else
110         return - 1;
111 }
112
```



The screenshot shows the Visual Studio Code editor with a file named 'Project_code.c' open. The code defines a 'grid()' function that initializes a 3x3 game grid. It uses 'system("cls");' to clear the screen and 'printf()' to display the grid state. The grid is represented as a 3x3 array of integers, with values 1 through 9. The function prints the grid state after each move.

```
112  
113 void grid()  
114 {  
115     system("cls");  
116     printf("\n\nwelcome to the game\n\n");  
117     printf(" for gamer 1 (X) - for gamer 2 (Y)\n\n");  
118  
119     printf("   |   |   \n");  
120     printf("  %c | %c | %c \n", square[1], square[2], square[3]);  
121  
122     printf("-----|-----\n");  
123     printf("   |   |   \n");  
124  
125     printf("  %c | %c | %c \n", square[4], square[5], square[6]);  
126  
127     printf("-----|-----\n");  
128     printf("   |   |   \n");  
129  
130     printf("  %c | %c | %c \n", square[7], square[8], square[9]);  
131  
132     printf("   |   |   \n\n");  
133  
134 }
```

Output:



The screenshot shows the terminal output of the program. It displays the welcome message, the prompt for player moves, and the current state of the 3x3 grid. The grid shows 'X' in the top row and 'Y' in the bottom row. The terminal also shows the command 'game 1 win'.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL  
  
welcome to the game  
  
for gamer 1 (X) - For gamer 2 (Y)  
  
X | X | X  
-----  
Y | 5 | 6  
-----  
Y | 8 | 9  
  
game 1 win
```