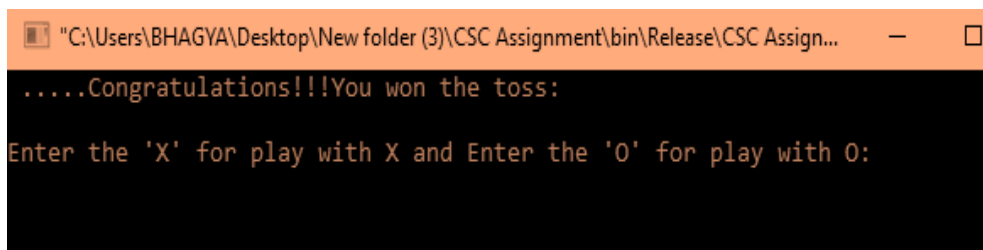


Documentation

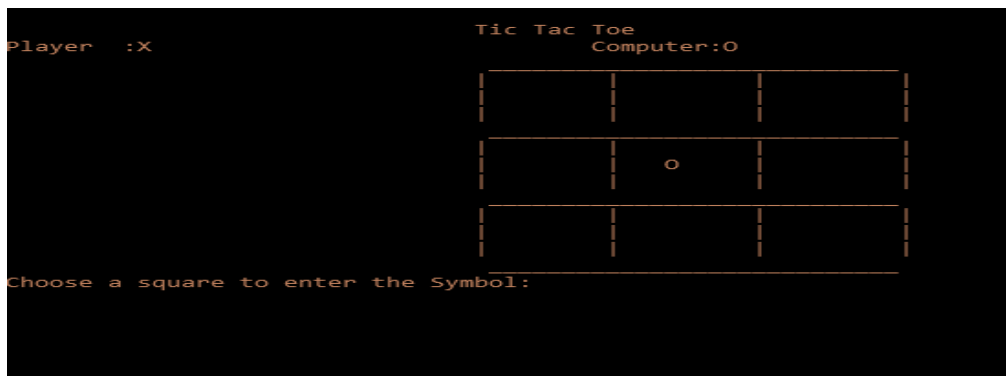
User Manual:

- When you won the toss you allow to select the symbol.
- To play with 'X' press X and to play with 'O' play O:

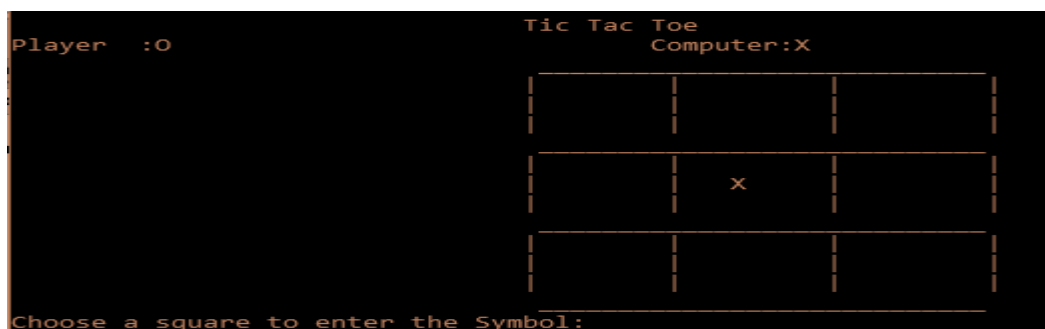


- When you does not won the toss you are unable to select the symbol. Then computer select the symbol and play first.
-

Here computer select O symbol and your symbol is X:



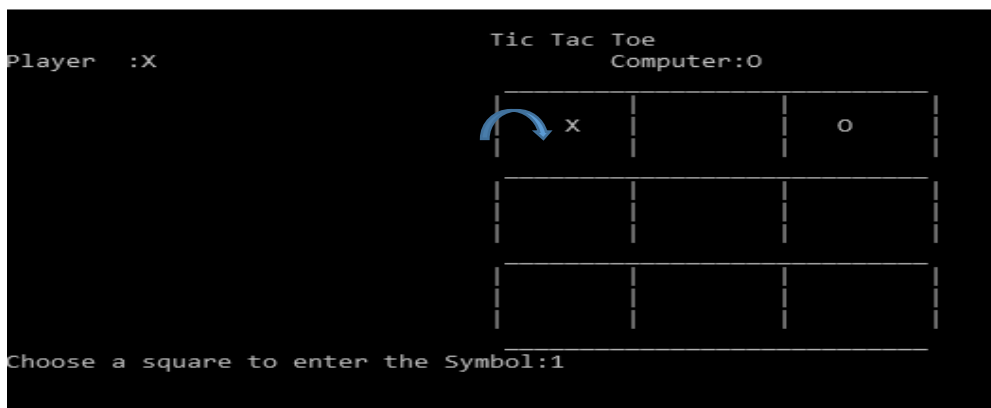
Here computer select X symbol and your symbol is O:



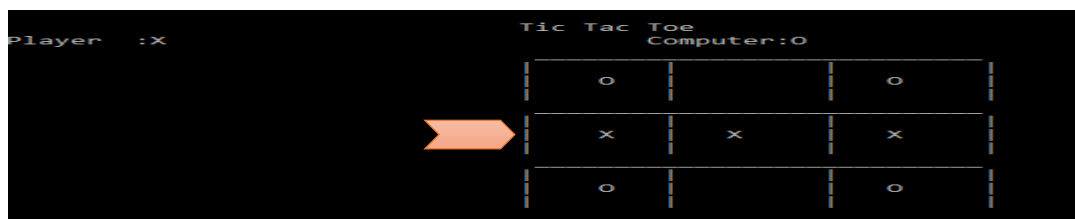
- How to choose a square to enter your symbol:
 - Press 1 → enter the symbol for square 1
 - Press 2 → enter the symbol for square 2
 - Press 3 → enter the symbol for square 3
 - Press 4 → enter the symbol for square 4
 - Press 5 → enter the symbol for square 5
 - Press 6 → enter the symbol for square 6
 - Press 7 → enter the symbol for square 7
 - Press 8 → enter the symbol for square 8
 - Press 9 → enter the symbol for square 9

Example:

.



- When you enter your symbol in to the row or column or diagonal you can win:



....Congratulations You won!!!.....

- To replay the game press 1 :

To replay press 1:

Maintenance Manual:

When the player choose a filled square player will receive a message .It is a one step of maintenance

- Then player should allow to re- select a square.

```
Player :X
Tic Tac Toe
Computer:O
| | | |
| x | x | o |
| | | |
| | | o |
|_|_|_|
Choose a square to enter the Symbol:3
*****Square filled*****
Please re-enter:
Choose a square to enter the Symbol:
```

When player enter an irrelevant number to choose square player will receive a message

- Then player should allow to re-select a square

```
Player :X
Tic Tac Toe
Computer:O
| | | |
| x | x | o |
| | | |
| | | o |
|_|_|_|
Choose a square to enter the Symbol:12
Invalid choice
Please re-enter to the another square:
Choose a square to enter the Symbol:
```

- Player allow to replay the game as many time as desired without having to restart the program

replay ➡ press

results:

In order to determine which evaluator had the best performance, they were printed against each other. The program makes use of maximum value position algorithm. The board is represented as a two dimensional array. A variety different evaluation function are provided. Also include the function for checking if the game has been won.

To detect a win this program simply looks at every possible row, column and diagonal to see if any one player's pieces occupy all 3 spaces. This achieved using a series of conditions. There are 76 different possibilities for win.

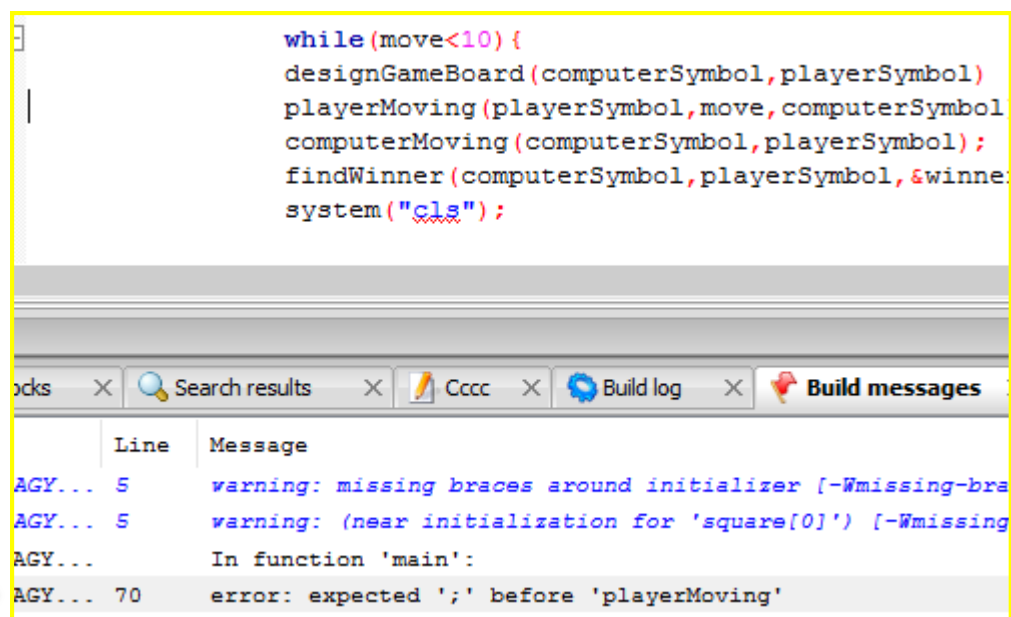
Problems encountered:

When programming this game it is bit difficult to build it without errors. At the first stages of programming there are too many errors such as syntax errors, logical errors and run time errors.

When calling function it is difficult to determine the order. Some functions are repeating. So we have to decide what is the most appropriate loop for it.

It is important to clear the system in each and every move. So we have to use conio.h header file and system("cls").

Those are the problems encountered during the tic tac toe programming.



```
while(move<10){
    designGameBoard(computerSymbol,playerSymbol)
    playerMoving(playerSymbol,move,computerSymbol
    computerMoving(computerSymbol,playerSymbol);
    findWinner(computerSymbol,playerSymbol,&winne
    system("cls");
```

The screenshot shows a code editor with the above code. Below the editor is a window titled 'Build messages' which contains the following messages:

Line	Message
AGY... 5	warning: missing braces around initializer [-Wmissing-br
AGY... 5	warning: (near initialization for 'square[0]') [-Wmissing
AGY...	In function 'main':
AGY... 70	error: expected ';' before 'playerMoving'

An orange arrow points to the error message on line 70.

Working schedule and progress report

	1 st week	2 nd week	3 rd week	4 th week
Read the assignment make algorithms				
Design the game board function				
Design the playerMoving function and computerMoving function				
Build the findWinner function				
Design whole program in to a manner.				
Make the Documentation				

Enhancements

In the future this project could be improved upon mainly by writing better evaluator functions. The current ones still allow the opponent to win sometimes, by falling to block, which could obviously use improvements. This program should run fast.