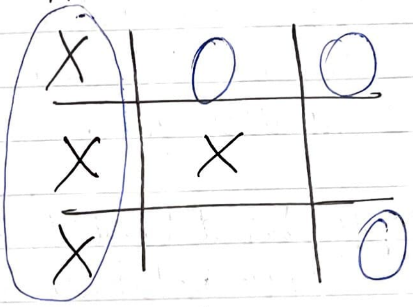
Final Exam Project . Topic 2 . Tic Tac Toe Game.

Whats Tic Tac Toe.
It's a two player game!.

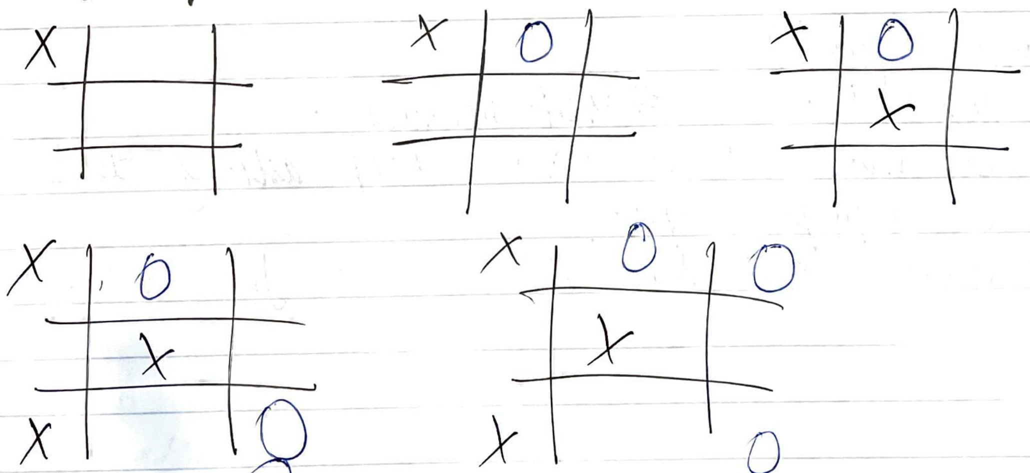
We have say... a 3 x 3 plot of land.
lets say you and your opponent are both farmers.

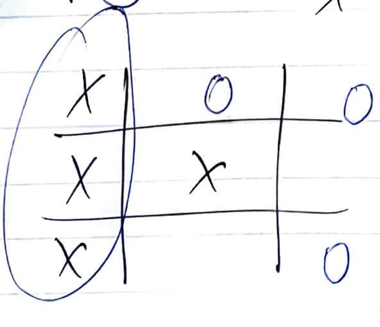


You must plant your crop in a diagonal or horizontal /vertical line.
And the same goes for your opponent.

You plant X crop and your opponent Y crop
But you only get to plant one at a time.

As you began planting your opponent also plants after you!





But here using your strategy you have managed to win!.

Now, we shall code this to be a 2 player game.

Step 1 : We write a function that can print out a board.

```
# assert : We set up our board as a list.
# We get a 3x3 board representation
# Each index 1-9 corresponds with a number
# on a number pad.

def display_board (board):
# assert    print statement with board[ ] (indexes).
```

Step 2 : We write a function that can take in player input and assign their marker (or the "crop they are planting") as 'X' or 'O'

```
    # INV : While loop to continually ask until we get a
# correct answer.

    def player_input()
        marker = ' '          # assert marker.
# INV   while not 'O' or 'X'   keep asking the
# user   to give a proper input.
          # assert  'X,O' for 'X', 'O,X' for 'O'.
```

Step 3: We write a function that takes the board list object, 'X' 'O' a desired position number (1-9) & assigns it to the board.

```
def place_marker(board, marker, position):
    board[position] = marker
```

Step 4: We write a function that takes in board and checks to see if someone has won it if theres an XXX or OOO in a diagonal or vertical/horizontal row/line together.

```
def win_check(board, mark):
    #assert : all possible winning combinations.
```

Step 5: Now personally when I play with my friends 'X' always goes first. I don't think that's fair. ∴
We will write a function using the random module to randomly decide which player goes first.

```
#assert : import random module.
def choose_first():
#INV   random b/w (0,1) if other, else:
```

step 6: We write a function that returns a boolean indicating whether a space is freely available.

```
def space_check(board, position):
    return board[position] == ' '
```

Step ~~6~~7: We cannot plant/place one players move over anothers.

∴

```
# We check if the board is full and returns a boolean
value.
    # INV return False for False else True
def full_board_check(board):
    # assert : to check in the range of 1 → 10.
```

Step 8: Once one players Turn is over, for the next player to plant/place there 'X' or 'O' we need to check if ~~there~~ ~~is~~ ~~a~~. the selected position is a free position using space_check.

~~def space_check~~.

```
def player_choice(board):
    # Assert position = 0 -
    # INV : Using while to check position to not be
    # in [1,2,3,4,5,6,7,8,9] or not to be in
    # space_check(board, position)
```

Step 9: Had Fun ?!
Well why only stop at one game. Now, we write a function asking the user if they'd like to play another game!

```
def replay():
    # assert : return for input to be Yes or No and
    # depending on so to continue or stop.
```