## * Problem statement :-

→ Tic tac toe is played on a 3x3 matrix where two players A & B play their moves.

→ Player A is always the first player in every game.

→ A : X & B : O

→ X & O is played only on empty squares & are never overwritten.

→ We are given a matrix called moves where moves[i] = [row_i, col_i] indicates that the $i^{th}$ move will be played on grid[row_i, col_i]

→ Return the winner of game ie A or B.

→ If no winner → "Draw"

→ If game can still continue → "Pending"

→ A move is only valid if grid[row][col] = " "

→ If all squares are non-empty & no-winner → "Draw"

## * Examples :-

1) moves = [[0,0], [2,0], [1,1], [2,1], [2,2]]

Output = A

2) moves = [[0,0], [1,1], [0,1], [0,2], [1,0], [2,0]]

Output = B

3) moves = [[0,0], [1,1], [2,0], [1,0], [1,2], [2,1], [0,1], [0,2], [2,2]]

Output = "Draw"

## * Solution :-

### * Winning condition :-

1) Row check (Assume player = "A")

grid[0][0] && grid[0][1] && grid[0][2] = player OR
grid[1][0] && grid[1][1] && grid[1][2] = player OR
grid[2][0] && grid[2][1] && grid[2][2] = player

} Then return true for that player.

2) Col check (Assume player = "A")

grid[0][0] && grid[1][0] && grid[2][0] = player OR
grid[0][1] && grid[1][1] && grid[2][1] = player OR
grid[0][2] && grid[1][2] && grid[2][2] = player

} Then return true for that player.

3) Main diagonal

grid[0][0] && grid[1][1] && grid[2][2] == player

Anti diagonal

grid[0][2] && grid[1][1] && grid[2][0] == player

Return main diagonal || anti diagonal

* Playing the game :-

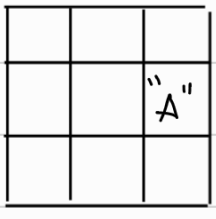
1) Create a grid of 3x3 & initialize it with " ".
2) Loop over the rows of moves
   i) moves[i] → currentMove
   ii) x = currentMove[0] (x coordinate on grid)
   iii) y = currentMove[1] (y coordinate on grid)
   iv) Since "A" is the first player & we are starting from i=0, every even $i^{th}$ value = A's chance.

   ∴ player = i % 2 == 0 ? "A" : "B"

   v) grid[x][y] = player

   So, if current move = [1,2],

   x = 1, y = 2, grid[1][2] = player




   vi) Check the winning conditions after each move.
   vii) If any winning condition returns true, return that player.
   viii) Else check if total moves == 9, then return "Draw" else return "Pending"

Time complexity :- $O(n)$ where $n$ is total moves played

Space complexity :- $O(1)$