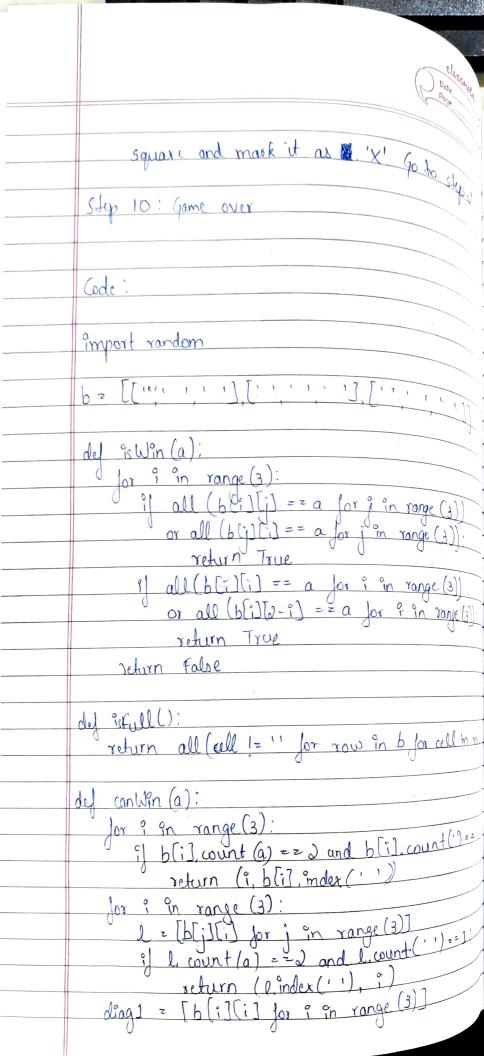
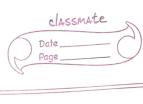


Step 2: Let the user play the first move, move = int (input ("Enter the no of Equare")) Step 3: Mark the Create a set of all the win condition Step 4! Mark the user move in the board as 'o' Step 5: Check of the user has won I so display 6top 5: det check Almost ( a) wer wins and go to step to Steps: Check if the board is full of it display tie. Step 5: Check of the computer can win the game on this move. Check of any row column or diagonals have two 'x' and one space', If there is, make that move and display computer wins Go to step 10 Step 7: Check if the player can win the game on the next move Check if any row column or diagonals have two 'O's and one space 'Af there is, mark that space as 'X'. Go to step 2 Sty &: If the middle square is empty, mark the move as 'X'. Go to step 2

Step 9: Out of the remaining spaces, pick a random





if diagl count (a) == 2 and diagl. count ('') == 1: return (diagl. index(''), diagl.index('')) dlag 2 = [b[i][3-i] for in range (3)]

if dlag 2 = [b[i][3-i] for in range (3)] return (diag). index (''), & -diago. index (''')) return (-1, -1) des display (). while True: display () x = int (input ("Enter row: ")) y = int (input ("Enter column!")) print ("Invalid move!") if is full (): if is Win ('0'): display () print (" You Win! ") (p, q) = canWin ('x') display ()
prant ("(omputer Wins!")

	Dat Samu
	Date Page
	34
	break (p,q) = canWin ('0')
	plbsdj. X,
	continue
	0 5 5 5 5 5 6 5 6 6 6 6 6 6 6 6 6 6 6 6
	b[1][1] = 'X'
	else:
	l = [(°, ')] for ° in range (3) for j on range  ?   b(')['] == ']
	ran = random choîce (e) b[ran [o]][ran [i]] = 'x'
	b(ran (o))(ran [1]) z 'X'
-	
-	
-	
-	
-	
-	
-	
-	
,	