N-Queens Problem

N = 4

Q			
	Q		
		Q	
			Q

↓

By this arrangement every row and coulum have one Queen

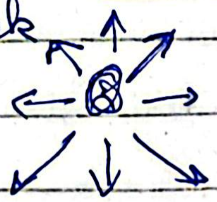
Rules

→ Every row should have Q

→ Every column should have one Queen

→ None of Quee should attack each other.

A Queen can attack in 8 direction



But in this case it can attack so not good arrangement.

So a better arrangements are

①

	Q		
			Q
Q			
		Q	

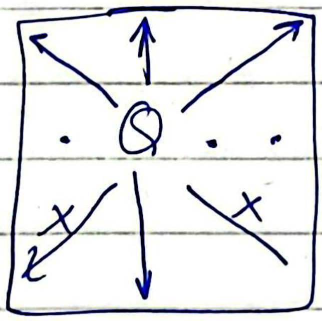
②

		Q	
Q			
			Q
	Q		

On every function call we increase row

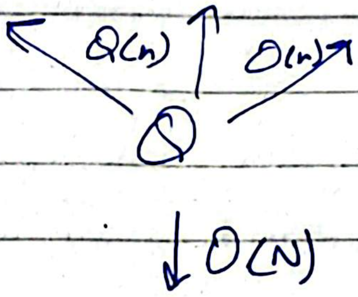
```
backtrack ( board, row)

and
  for col in range(n):
      if is_safe (board, row, col):
          then bard= Q
          backtrack (board, row+1)
          then bord = "."
```

and in-safe is checking
upper left diagonal.
upper right diagonal.
and column



these two not
cuz we are not
adding in them
yet.

But it is not effecient cuz on every adding of Queen



thats why O(N) + O(N) + O(N

O(N) * O([illegible])

OPTIMIZED SOLUTION:

we can reduce checking again and again by having hash sets of cols, diag neg, diag pos.

	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9
3	3	4	5	6	7	8	9	10
4	4	5	6	7	8	9	10	11
5	5	6	7	8	9	10	11	12
6	6	7	8	9	10	11	12	13
7	7	8	9	10	11	12	13	14

as we can see the pattern.

if .

if I place a Queen at
lets say 5 and 7

So 5+7=12 is added in set
this is for left row and
~~#~~ lower diagonal.
top rigth to bottom letf

for upper diagonal

formula ~~(n-1) + (col-row~~) (row - col)

	0	1	2	3	4	5	6	7
0	7	8	9	16	11	12	13	14
1	6	7	8	9	10	11	12	13
2	5	6	7	8	9	10	11	12
3	4	5	6	7	8	9	10	11
4	3	4	5	6	7	8	9	10
5	2	3	4	5	6	7	8	9
6	1	2	3	4	5	6	7	8
7	0	1	2	3	4	5	6	7

let say i fill board[5][6]=Q
then insted of checking
all i can just 5-6=1
which is also 2-1=1
in set then not safe
same 6-7=1 is also so the
Queen is not safe if it
is in that block.

Same for row + col

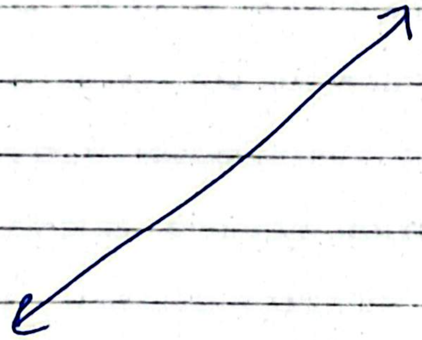
if we see ~~5+7=12~~ then

4 + 4 = 8

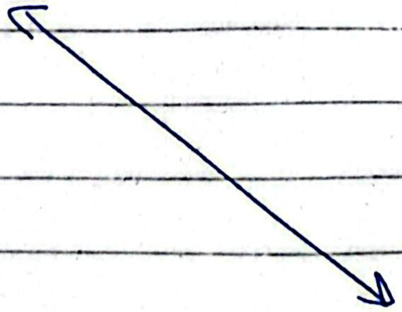
same as 3+5 = 8 same as

2+6 = 8

So row + col check



row - col check



and one sel for col