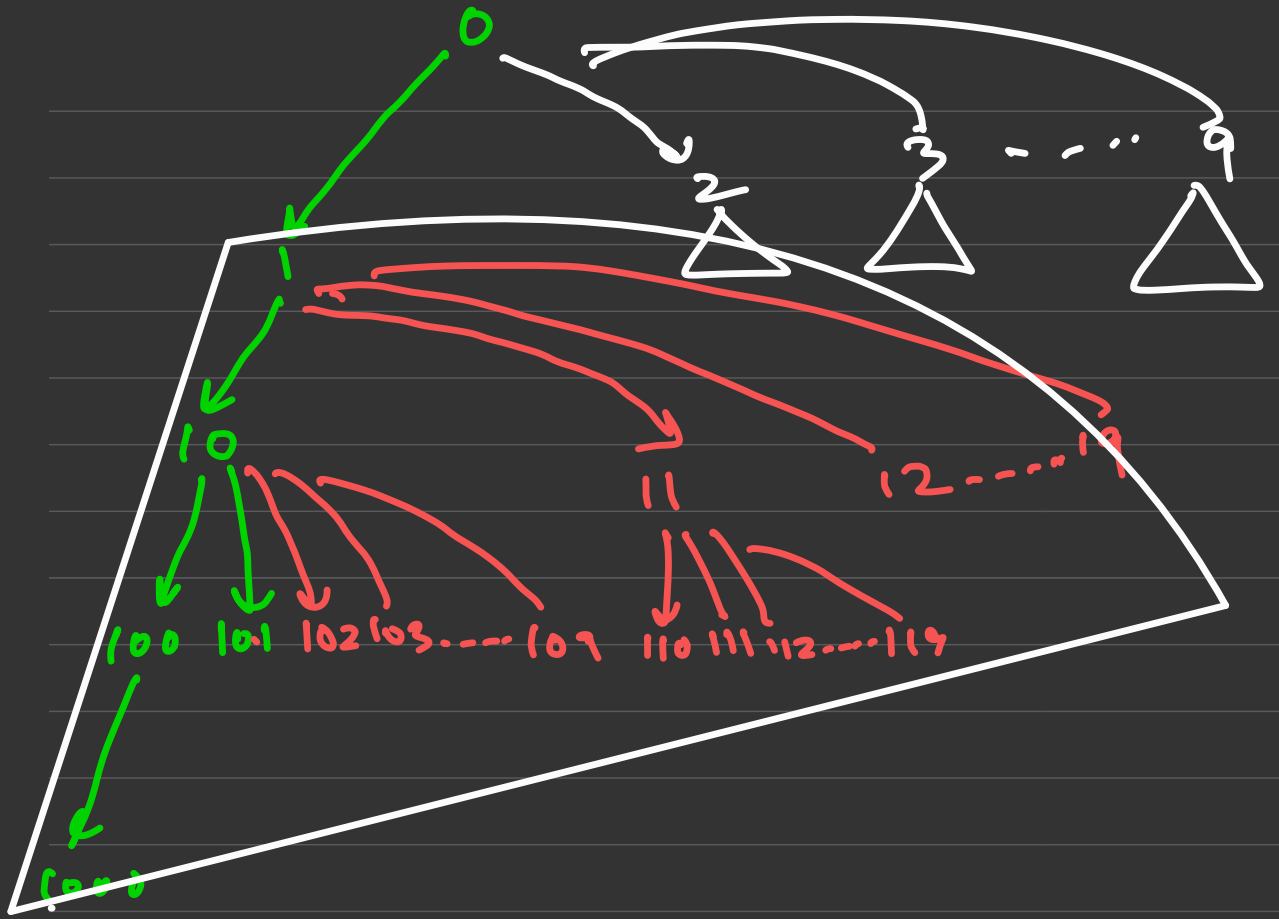
Subsequence → optimum for subsequence

recursively



x

0 - A
1 - B
2 - C
3 - D
4 - E
⋮ ⋮

Note

00 → 0

AA XX

A

AB → 01 → 1

ABA → 10

ABAA → 100

ABAAA → 1000

ABAB → 101

ABAC → 102

⋮

$f(i)$ = print(i) → $f(10*i+j)$ $\forall j \in [0,9]$

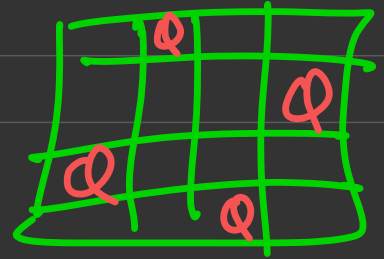
$f(i)$

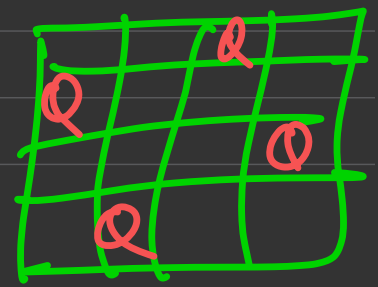
return all the nos starting with i in lexicographical order

N queens

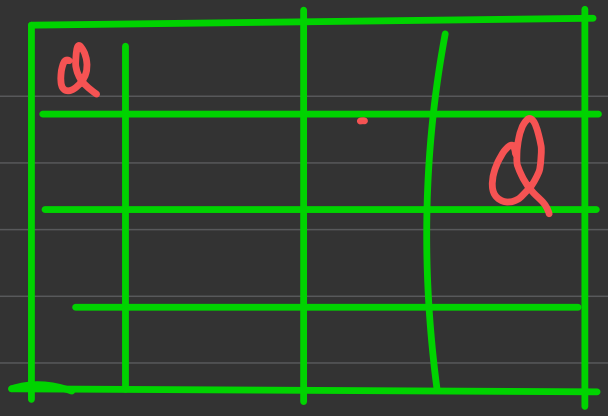
Q. Given a value of N, Consider a N×N board, where you've to place N-queens such that no queen attacks the other. Count the total no. of ways in which you can achieve such configuration.

N=4 → 2





↳

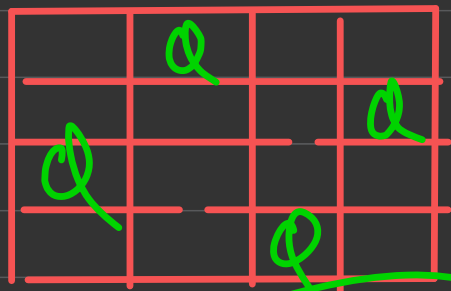


19mm

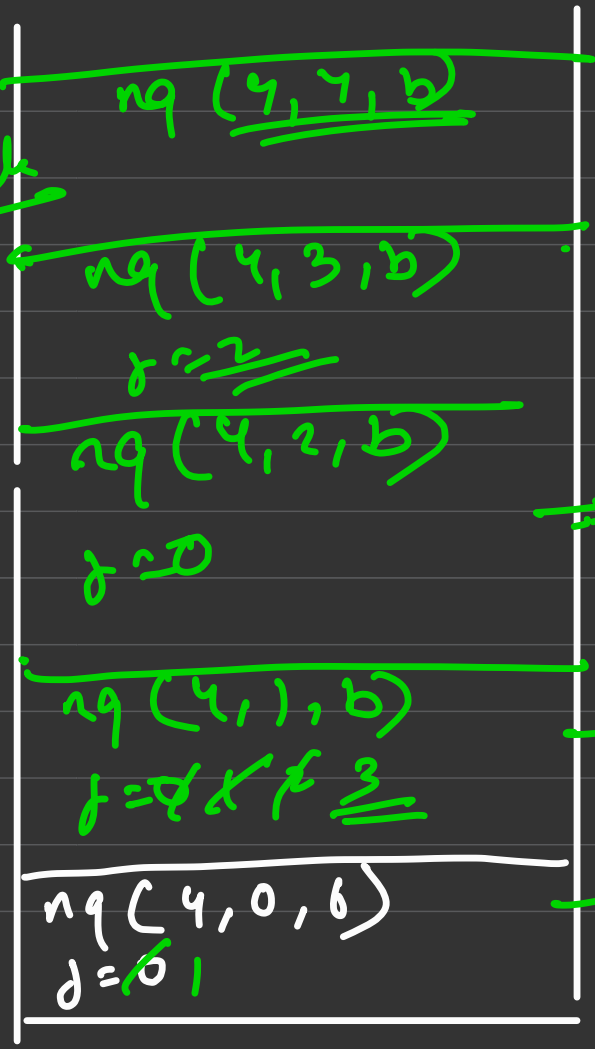
```cpp
void nq  int i : n, int i, std::vector<std::vector<bool > > &board) {
    if(i == n) {

        ways++;
        return;
    }
    for(int j = 0; j < n; j++) {
        if(isSafe(board, i, j)) {
            board[i][j] = true;
            nqueen(n, i+1, board);
            board[i][j] = false;
        }
    }
}
```

setup for backtrack

step of backtrack



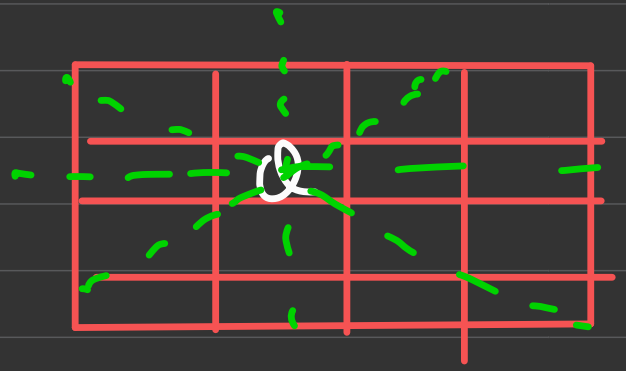
ways = 1



$$T(n) = n(n + T(n-1))$$

$O(n!)$

nqueen using bitmasking



0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3

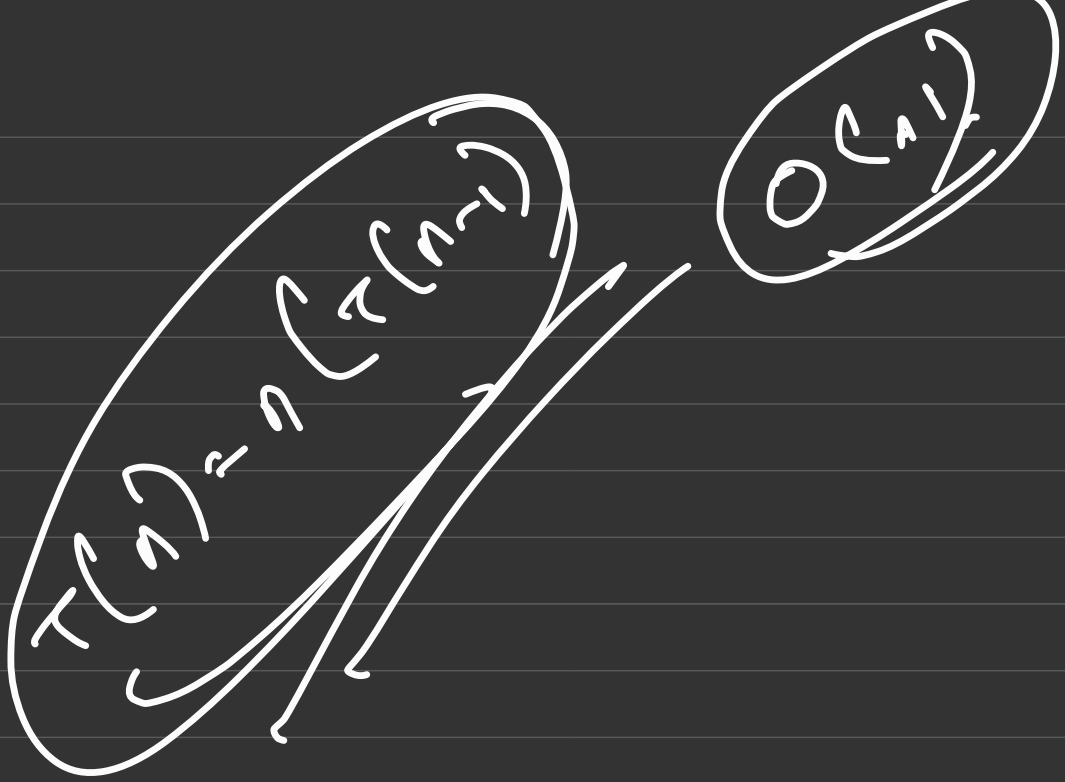
dt[i-j+n-1] = T

col → ∅

col[]

col[j] = T

dt[i+j] = T




N Knights