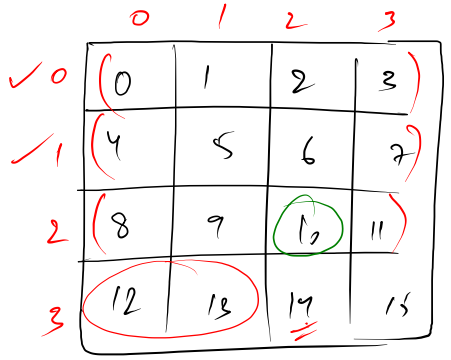
2D Array → Represent 1D



n → no. of els in each row

row = cellno / n

col = cellno % n

cellno. ⇒ (row * n) + col

Eg : cellno. = 10

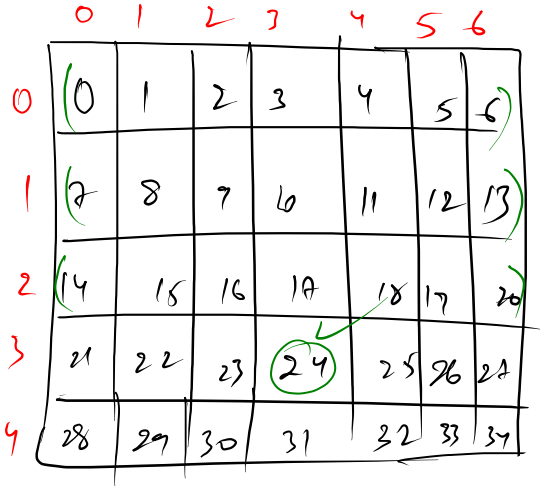
row ← 10/4 = 2

col ← 10 % 4 = 2

cellno. ⇒ (2 * 4) + 2
⇒ 10

row = 14/4 ⇒ 3

col = 14 % 4 = 2



cell ⇒ 24

24 ⇒ 7 + 7 + 7 + 3

row = cell/7 ⇒ 24/7 ⇒ 3

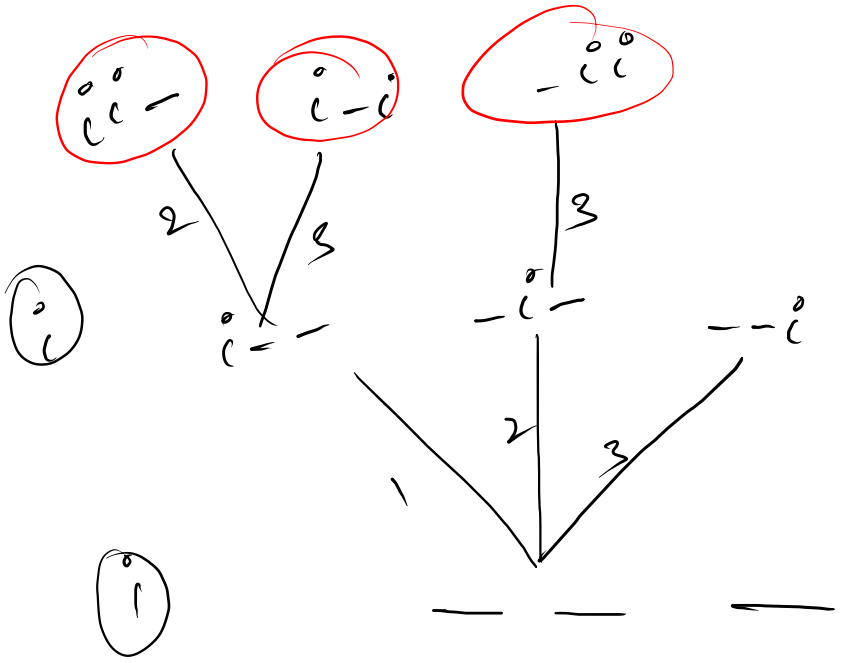
col ⇒ cell % 7 ⇒ 24 % 7 ⇒ 3

Combinations ⇒ object choose

$n = 3, r = 2$ (Similar)

$n = 2$

1	2
3	4



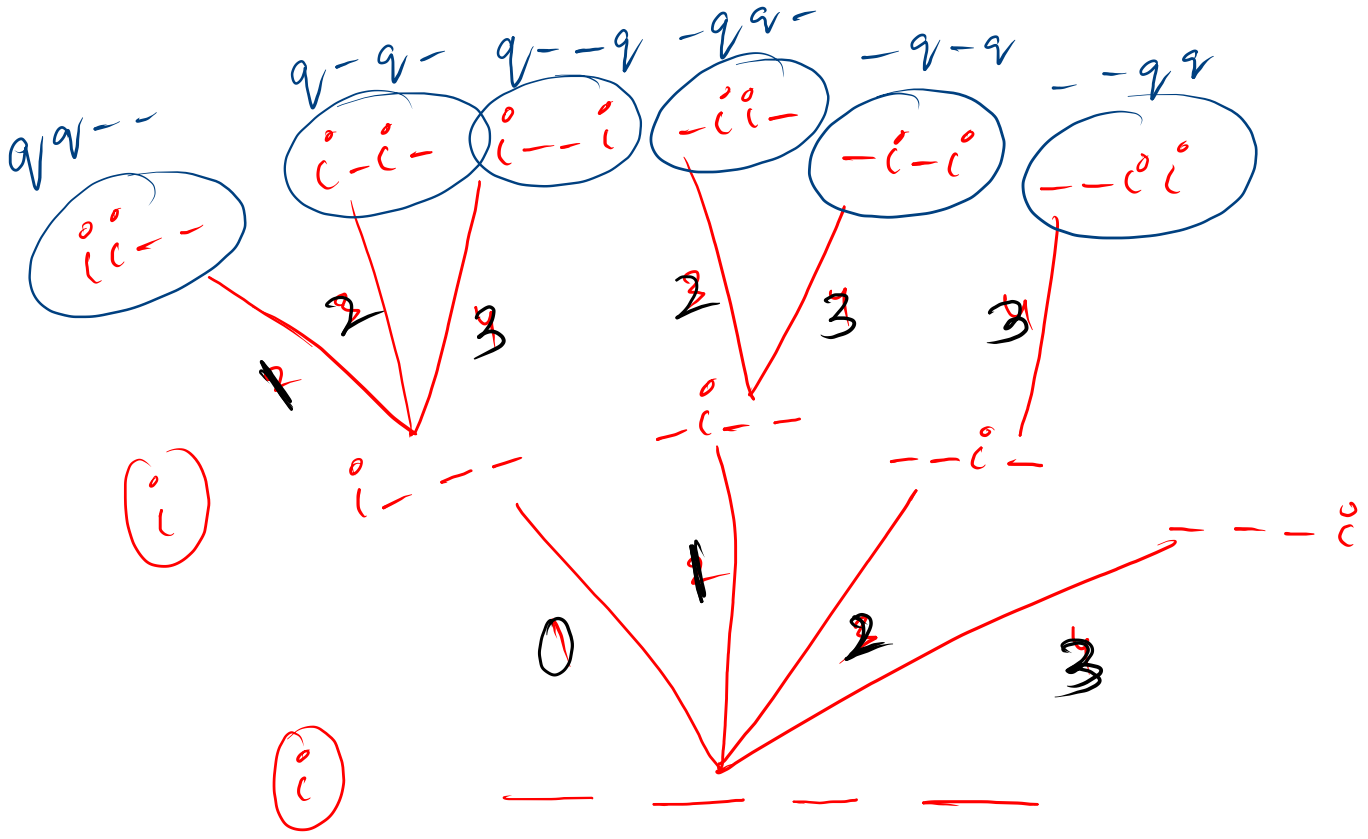
$n \leq 2$

0	1
2	3

$$\begin{pmatrix} q & q \\ - & - \end{pmatrix} \quad \begin{pmatrix} -q \\ q \end{pmatrix}$$

$$\begin{pmatrix} q & - \\ q & - \end{pmatrix} \quad \begin{pmatrix} -q \\ -q \end{pmatrix}$$

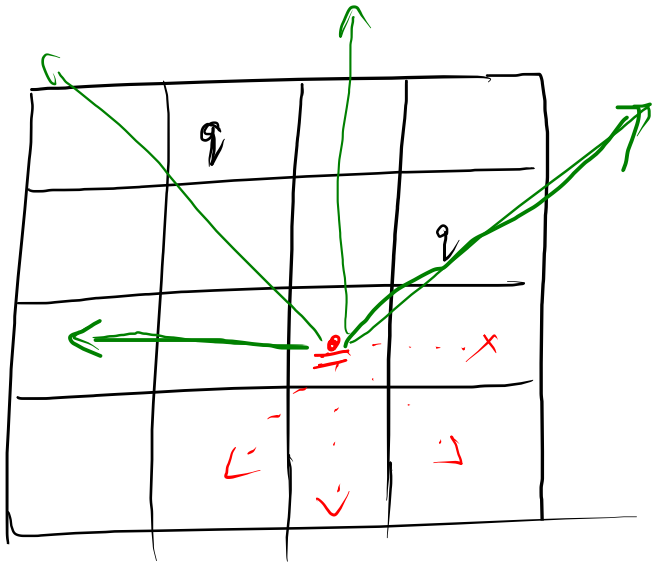
$$\begin{pmatrix} q & - \\ - & q \end{pmatrix} \quad \begin{pmatrix} - & - \\ q & q \end{pmatrix}$$

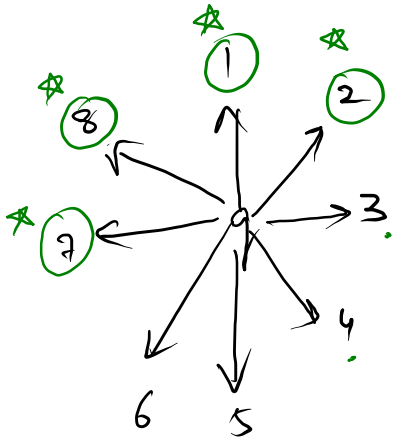


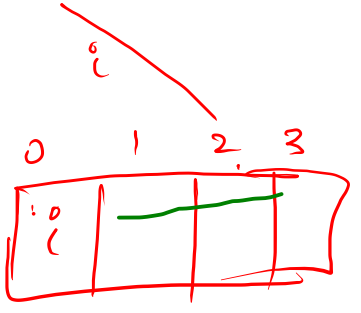
```java
public static void queensCombinations(int qpsf, int tq, boolean[][] chess, int lcno) {
    if(qpsf == tq){
        for(int i = 0 ; i < chess.length ; i++){
            for(int j = 0 ; j < chess[0].length ; j++){
                if(chess[i][j]){
                    System.out.print("q\t");
                }else{
                    System.out.print("-\t");
                }
            }
            System.out.println();
        }
        System.out.println();
        return;
    }

    for(int cell = lcno+1 ; cell < tq*tq ; cell++){
        int row = cell / tq;
        int col = cell % tq;

        chess[row][col] = true;
        queensCombinations(qpsf+1,tq,chess,cell);
        chess[row][col] = false;
    }
}
```

Combination + N queen

n=4







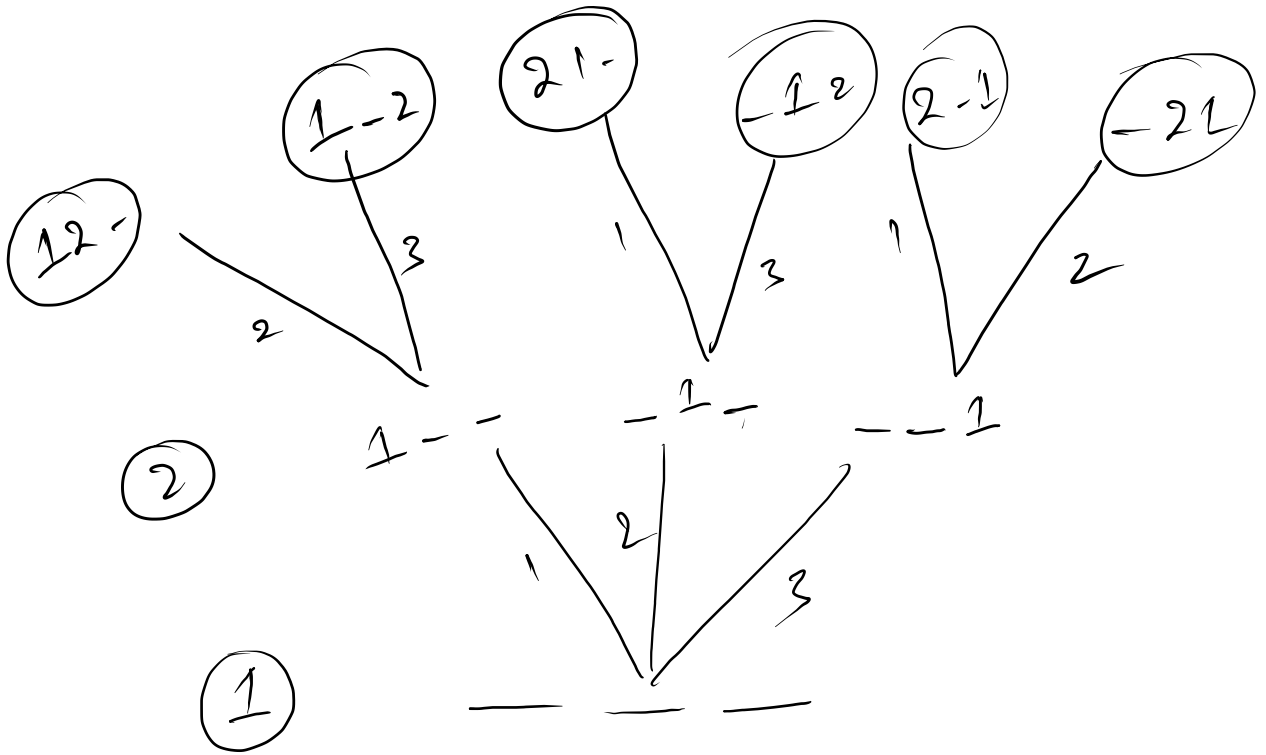
Permutation

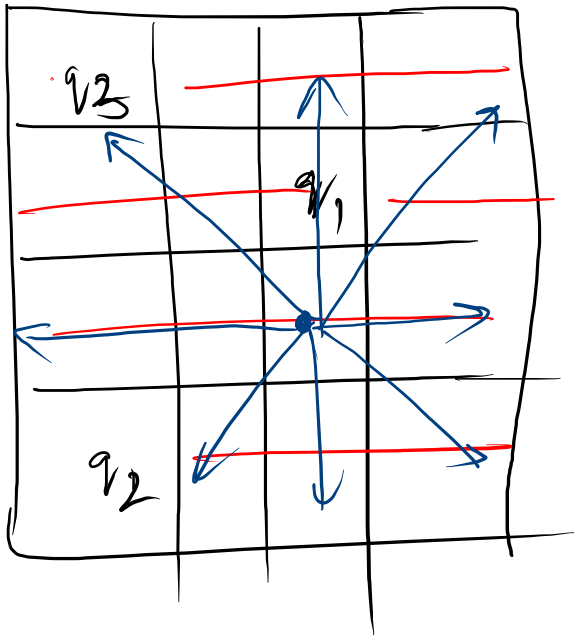
$n = 3$ $r = 2$

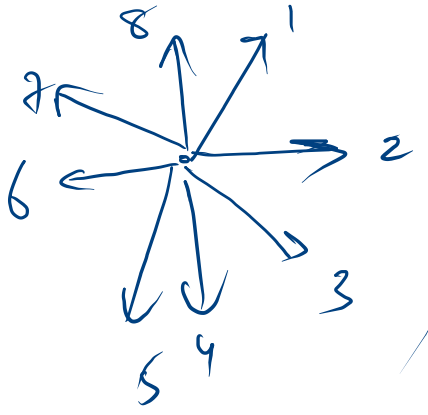
↳ distinct

$n = 2$

0	1 q_1
2	3




q3
q1
q2


8
1
7
2
6
3
5
4

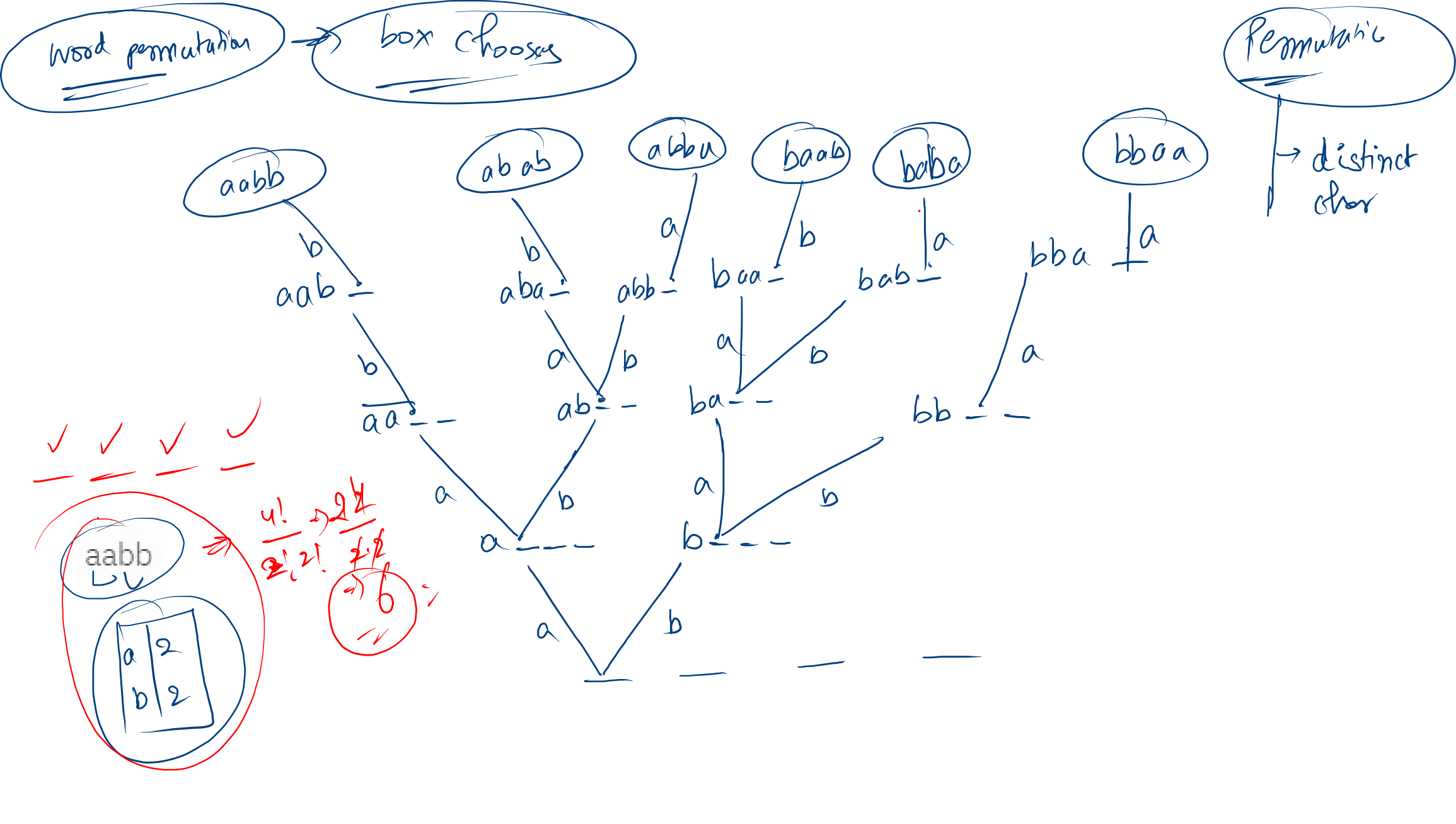
```java
public static boolean IsQueenSafe(int[][] chess, int row, int col) {
    return true;
}
```

How?

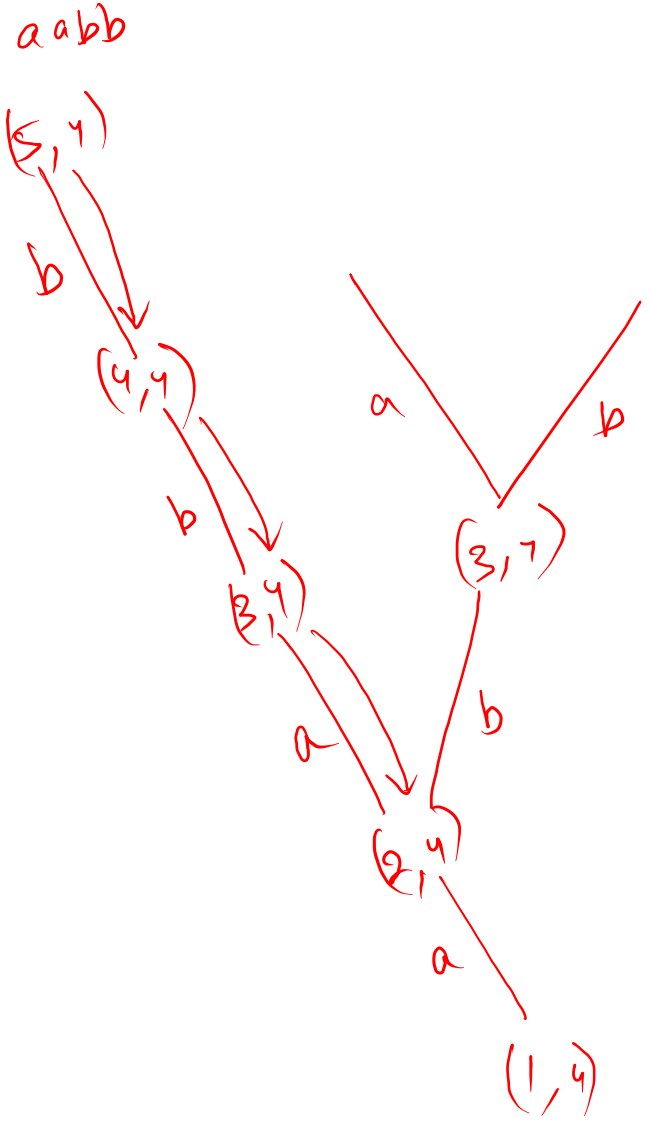
Modify

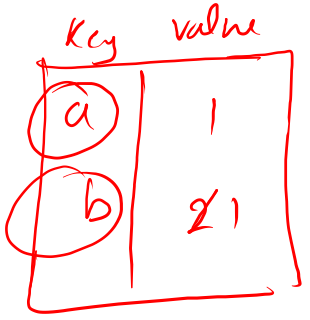
Complete

```java
public static void nqueens(int qpsf, int tq, int[][] chess) {
    if(qpsf == tq){
        for(int i = 0 ; i < chess.length ; i++){
            for(int j = 0 ; j < chess[0].length ; j++){
                if(chess[i][j] != 0){
                    System.out.print("q"+chess[i][j]+"\t");
                }else{
                    System.out.print("-\t");
                }
            }
            System.out.println();
        }
        System.out.println();
        return;
    }

    for(int cell = 0 ; cell < tq*tq ; cell++){
        int row = cell / tq;
        int col = cell % tq;

        if(chess[row][col] == 0 && isSafe(chess,row,col) == true){
            chess[row][col] = qpsf+1;
            nqueens(qpsf+1,tq,chess);
            chess[row][col] = 0;
        }
    }
}
```

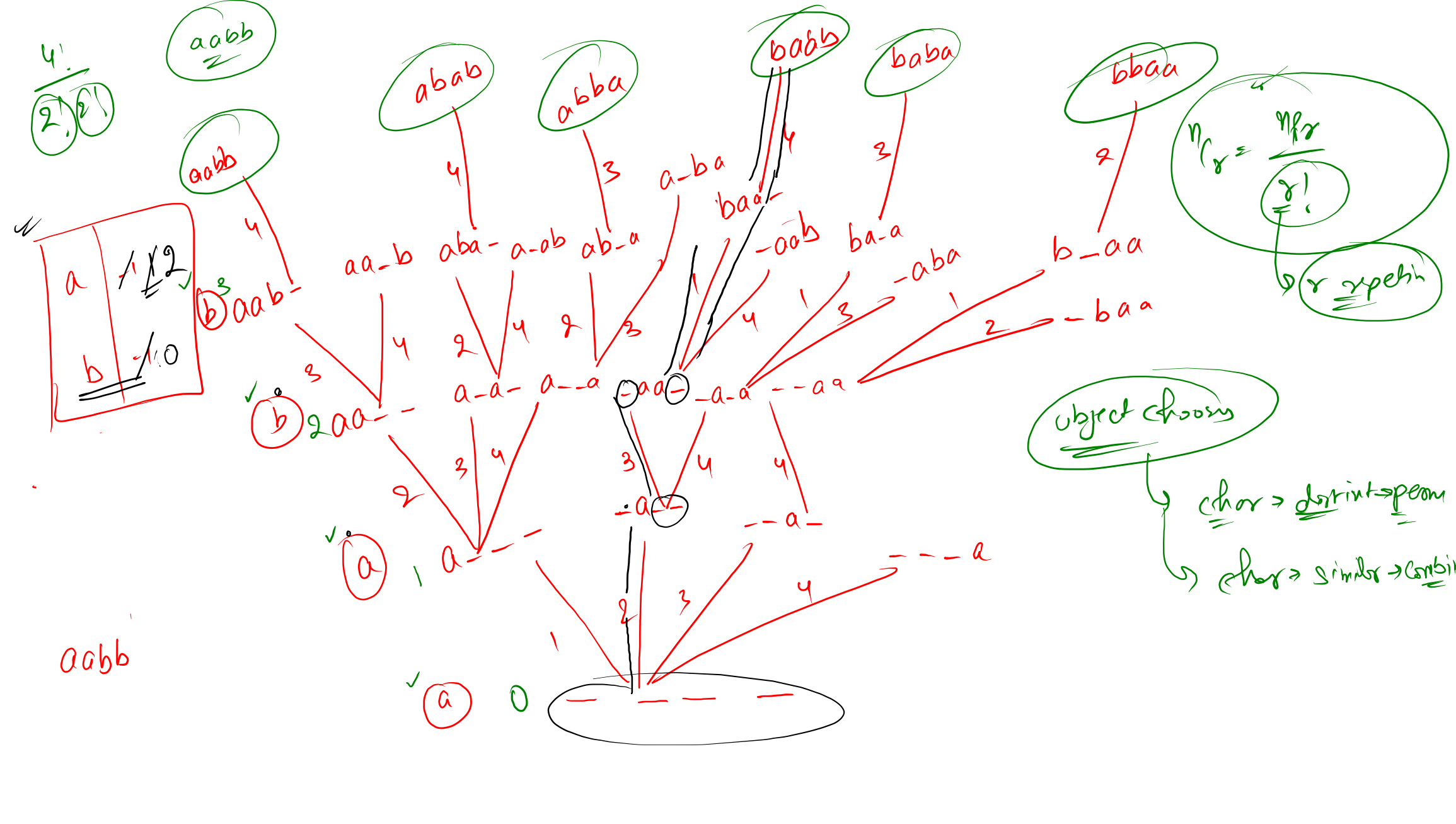

word permutation
box choose
Permutation
distinct char
aabb
abab
abba
baab
baba
bbaa
aab
aba
abb
baa
bab
bba
aa
ab
ba
bb
a
b
aabb
a 2
b 2
4! → 24
2! 2! → 4
6

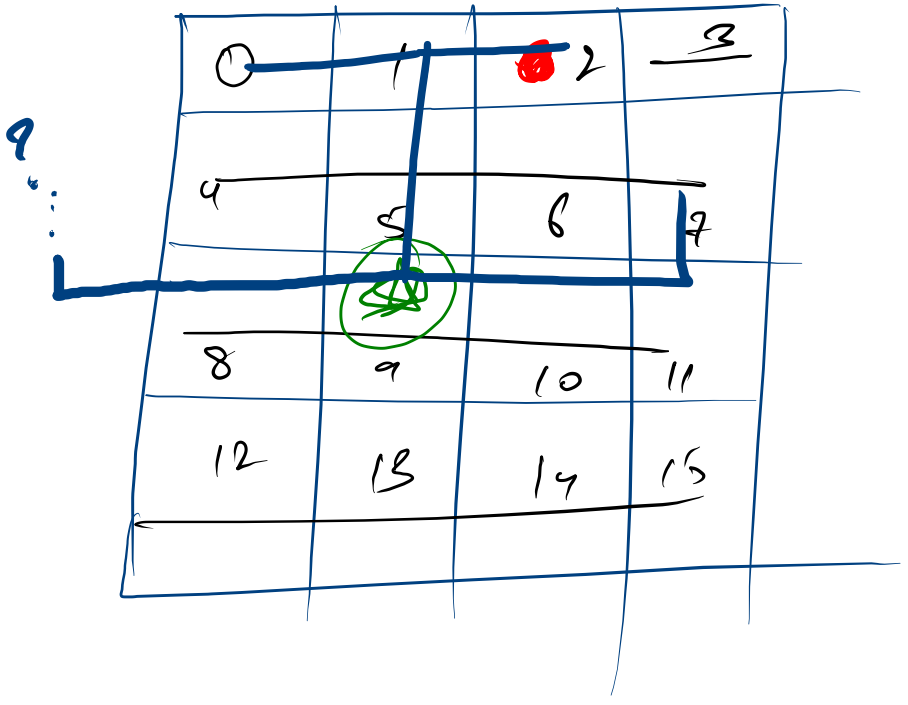
aabb

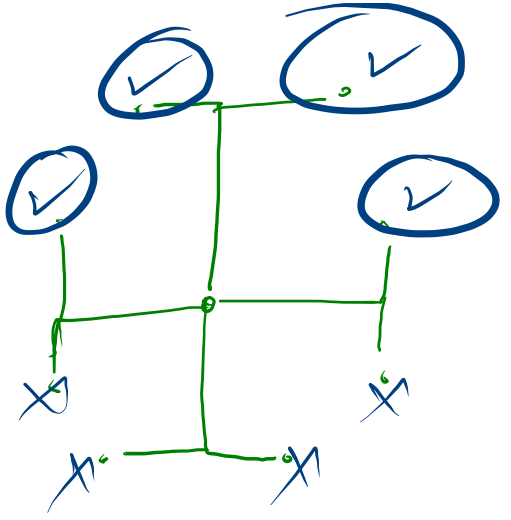





```
public static void generateWords(int cs, int t
  if(cs > ts){
      System.out.println(asf);
      return;
  }

  for(char ch : fmap.keySet()){
      int freq = fmap.get(ch); //2
      if( freq > 0){
          fmap.put(ch,freq-1);
          generateWords(cs+1,ts,fmap,asf+ch);
          fmap.put(ch,freq);
      }
  }
}
```


4!
2!
2!
aabb
aabb
abab
abba
baab
baba
bbaa
a
b
aab
aa-b
aba-
a-ab
ab-a
a-ba
baa-
-aab
ba-a
b-aa
-aba
-baa
a-a-
a--a
-aa-
-a-a
--aa
-a-
--a
---a
aabb
object chooses
char → distinct → perm
char → similar → combi
η(γ) = η!γ / γ!
γ repeti


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15



NOTES

30

Friday

6 days → 2

2 + 2 → 4

Queens Combinations - 2d As 1d - Queen Chooses	done
Nqueens Combinations - 2d As 1d - Queen Chooses	done
Nqueens Permutations - 2d As 1d - Queen Chooses	HW ☆ → Today
Permutations - Words - 1	done
Permutations - Words - 2 ✓	Try
Nknights Combinations - 2d As 1d - Knight Chooses	HW ☆

Nqueen + queens → K knights