

Queen combinations - 2D as 2D - Box choosed:

In $n \times n$ board, n identical queens [behaviour is not as chess queen], print all possible ways to place queens in chessboard.

level - Box
options + choice of queens.

$$2^n = {}^nC_0 + {}^nC_1 + {}^nC_2 + \dots + {}^nC_n$$

↑
idea from subseq.

Result →

$n=4 \rightarrow$ place

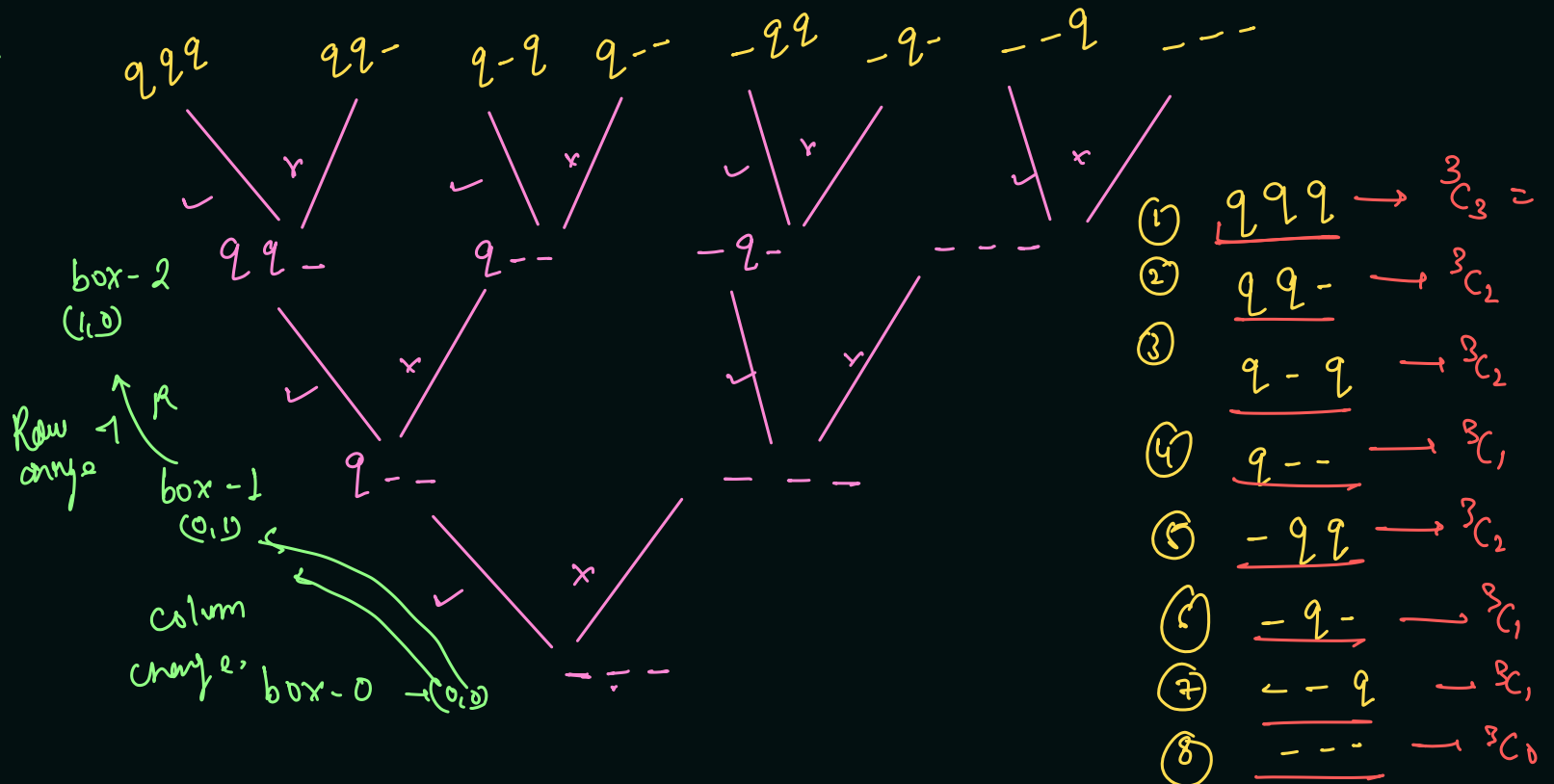
$r=2 \rightarrow$ item.

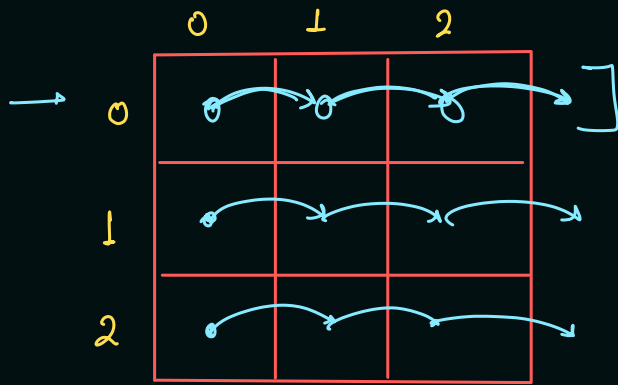
$$2^4 = {}^4C_0 + {}^4C_1 + {}^4C_2 + {}^4C_3 + {}^4C_4$$

$n=3$
 $r=2$

$$2^3 = {}^3C_0 + {}^3C_1 + {}^3C_2 + {}^3C_3$$

↓
⑥ $1 + 3 + 3 + 1 = 8$





column == total column
 ↳ Increment in Row
 & column begin
 from zero

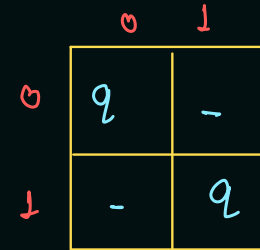
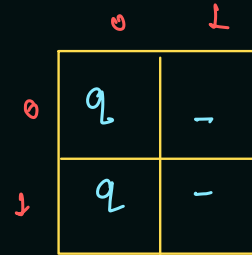
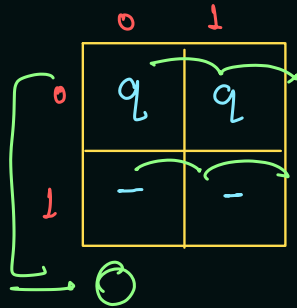
Chess board, $2 \times 2 = 4$ box.

no. of kings/queen: 2 → need to place.

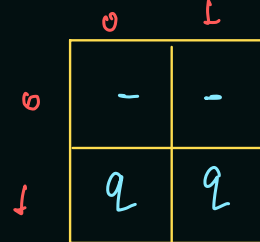
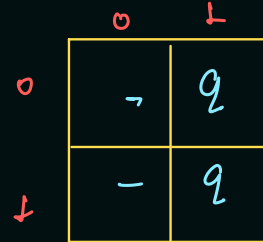
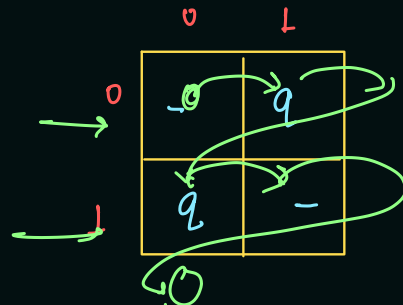
$$\text{Identical items} = {}^4C_2 = \frac{4!}{2!2!} = \frac{4 \times 3 \times 2!}{2! \times 2!} = \frac{4 \times 3}{2 \times 1} = 6$$



Row == total Row



'\n' → new line



string → str = "Shreesh\ntripathi";

⇒ qq\n

syso(str);
 → Shreesh
 tripathi

"q-\n"

q-\n

-q-\n

$$n = 2.$$

Board $\rightarrow 2 \times 2$

feminized

no. of queens = $1q = 2$

$$\frac{[g, r]}{r, c}$$

feminist

4; 99

99
3.9-

99
-9

29

3, 99

2-
2-

9-9 9-

3, 9, 9

$$\begin{pmatrix} -9 \\ 9 \end{pmatrix}$$

$\begin{pmatrix} -9 & -9 \\ -9 & -9 \end{pmatrix}$

29

$\begin{matrix} - & - \\ 9 & - \end{matrix}$

A diagram of a cell with a blue arrow pointing to a red structure.

```

public static void queensCombinations(int qpsf, int tq, int row, int col, String asf) {
    if(row == tq) {
        if(qpsf == tq) {
            System.out.println(asf);
        }
        return;
    }

    if(col + 1 < tq) {
        // mauka no. 1 -> only increment in column
        // yes call
        queensCombinations(qpsf + 1, tq, row, col + 1, asf + "q");
        // no call
        queensCombinations(qpsf, tq, row, col + 1, asf + "-");
    } else {
        // mauka no. 2 -> increment row, and column begin from 0
        // yes call
        queensCombinations(qpsf + 1, tq, row + 1, col: 0, asf + "q\n");
        // no call
        queensCombinations(qpsf, tq, row + 1, col: 0, asf + "-\n");
    }
}
}

```

[illegible]
$${}^4C_3 = 4$$
$${}^4C_2 = 6$$
$${}^4C_1 = 4$$
$$q_{C_0} = 1$$
$$\begin{array}{cc} & \text{H} \\ & | \\ \text{H} - & \text{C} - \text{H} \\ & | \\ & \text{H} \end{array}$$
$$\begin{array}{cc} & 4 \\ & C_3 \\ q & q \\ q & - \end{array}$$
$$\begin{array}{r} 4 \\ 99 \\ -9 \end{array}$$

4
C₃
9-
99

$q - \frac{y}{2}$
 $q -$

$$\begin{array}{r} 2 \\ -9 \end{array}$$
$$a - \frac{y}{c_1}$$
$$\begin{array}{r} 4 \\ -9 \end{array} \begin{array}{l} C_3 \\ 99 \end{array}$$

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

$$\begin{pmatrix} -9 \\ -9 \end{pmatrix} \quad \begin{matrix} 4 \\ C_2 \end{matrix}$$
$$- 2^4 C_1$$

A handwritten diagram of a circle. Inside the circle, there are two horizontal dashed lines at the top and two vertical lines on the right side. The vertical lines are labeled k_1 and k_2 in blue ink.

$$\begin{array}{r} 92 \\ - 46 \\ \hline 46 \end{array}$$
$$\begin{array}{r} \text{---} \text{---} 4 \\ \text{---} 9 \end{array}$$
[illegible]

queens permutation - 2D as 2D - Queen chooses!

$n \times n \rightarrow$ chess board, $n \rightarrow$ non identical items/queen

print all possible ways to place n -items in chess board.

level \rightarrow queen/item.

$$nPr = (n-0) \times (n-1) \times (n-2) \times \dots \times (n-(r-1))$$

no. of possible ways to place

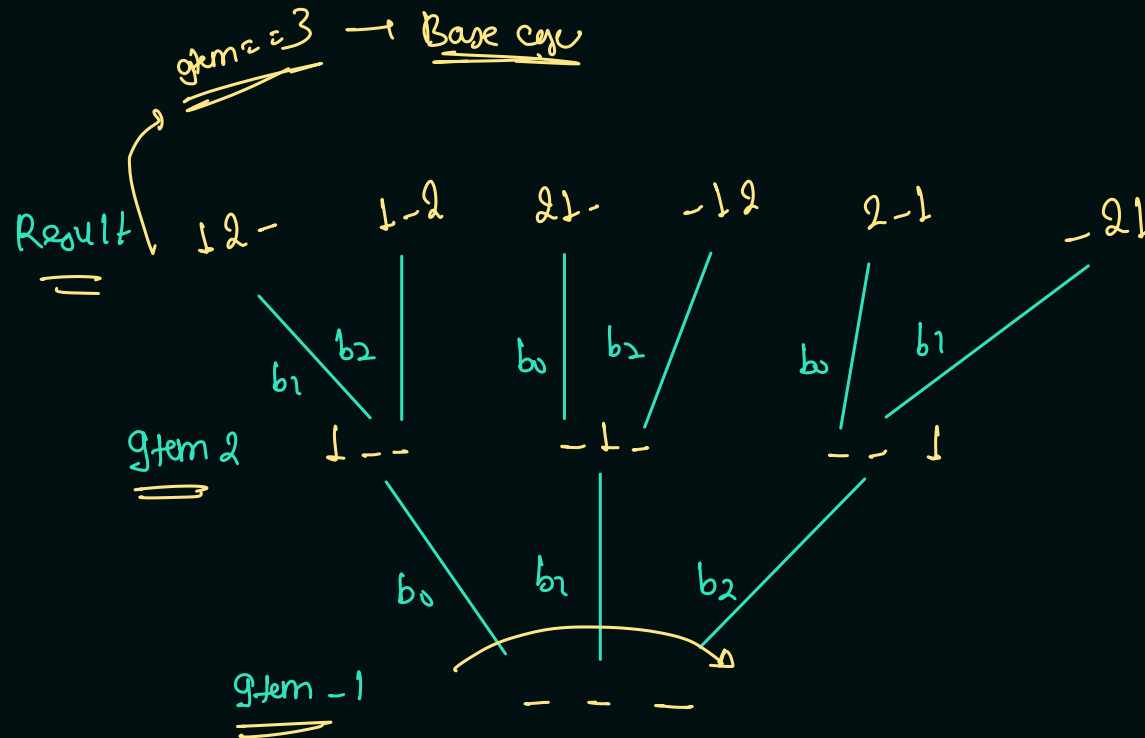
Options \rightarrow boxes

$n=3$ - boxes

$r=2 \rightarrow$ items

2 items on

3 boxes.



1 2 -

1 - 2

2 1 -

- 1 2

2 - 1

- 2 1

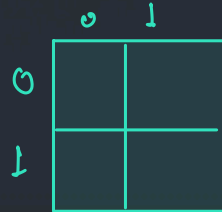
$n=2$, Board = 2×2 .

no. of non-identical items / queen = 2

$${}^4P_2 = \frac{4!}{2!} = \frac{4 \times 3 \times 2}{2} = \underline{\underline{12}}$$

// qpsf -> queen place so far, tq -> total queen, level -> queen, optio

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



// option -> box

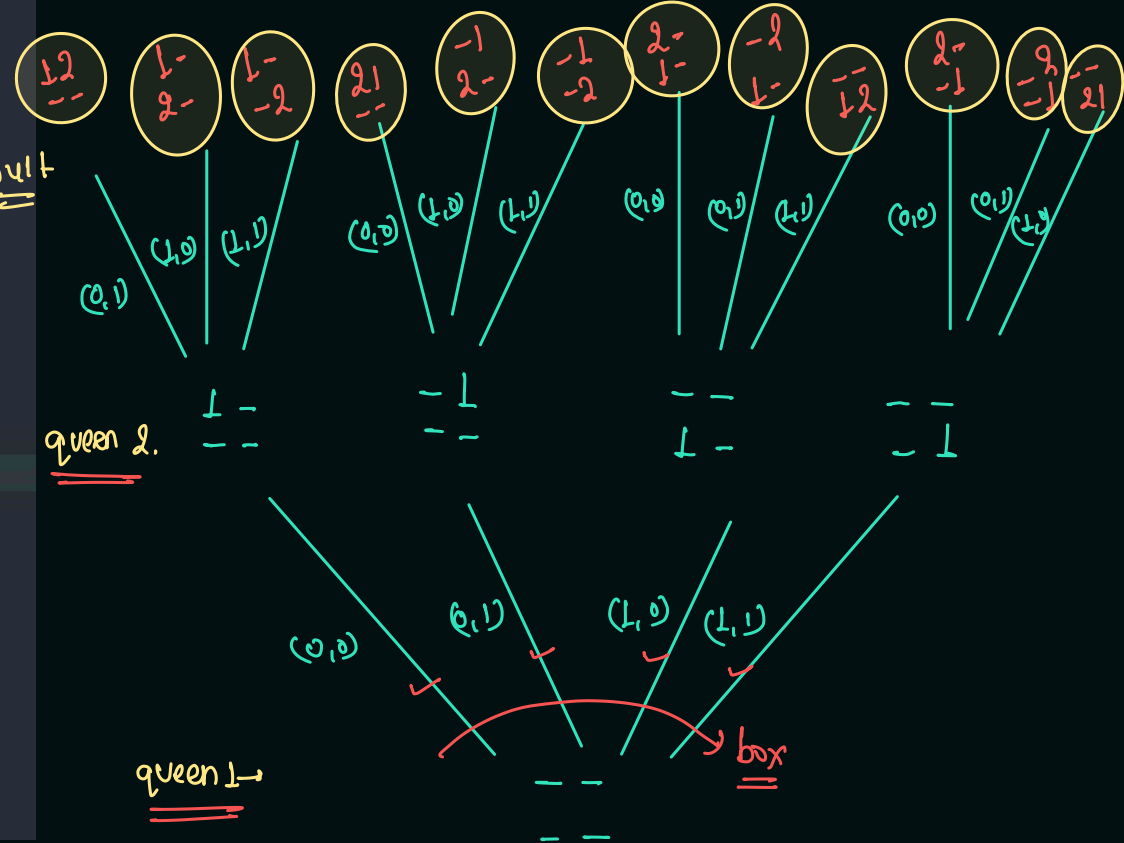
```
for(int i = 0; i < chess.length; i++) {
    for(int j = 0; j < chess.length; j++) {
        if(chess[i][j] == 0) {
            chess[i][j] = qpsf + 1;
            queensPermutations(qpsf + 1, tq, chess);
            chess[i][j] = 0;
        }
    }
}
```

1 2 1 - 1 - 2 1 - 1 - 1
- - 2 - - 2 - - 2 - - 2

2 - - 2 - - 2 - - 2 - -
1 - 1 - 1 2 - 1 - 1 2 1

Result

result



queen permutation 2D as 2D - Box choosen → $n \times n$ → chess board, n → non identical items/queen
 print all possible ways to place n - items in chess board.

level → Box

option → queens.

$${}^n C_r = \frac{n!}{(n-r)! r!}$$

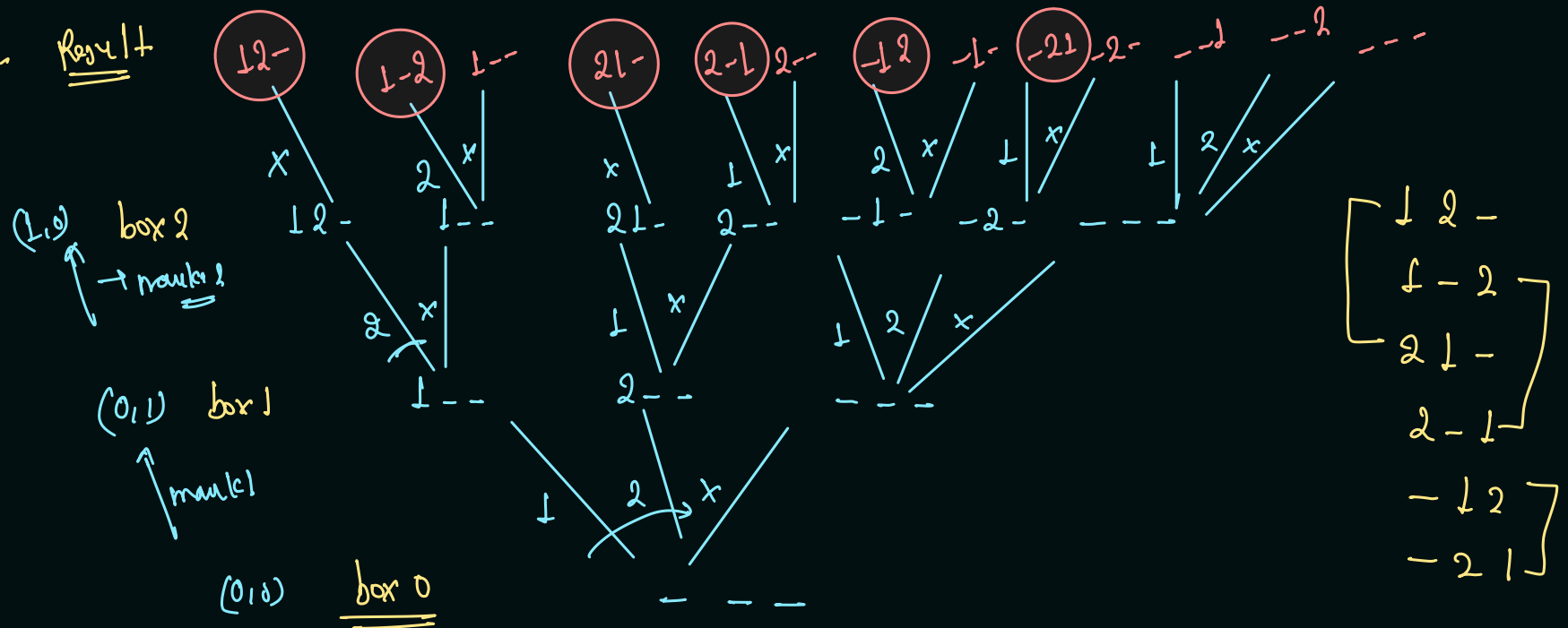
$${}^n P_r = \frac{n!}{(n-r)!}$$

$${}^n P_r = \left[{}^n C_r \times r! \right]$$

$\xrightarrow{\text{Yes or No}}$ $\xrightarrow{\text{loop on ! item}}$

$n=3$ → box

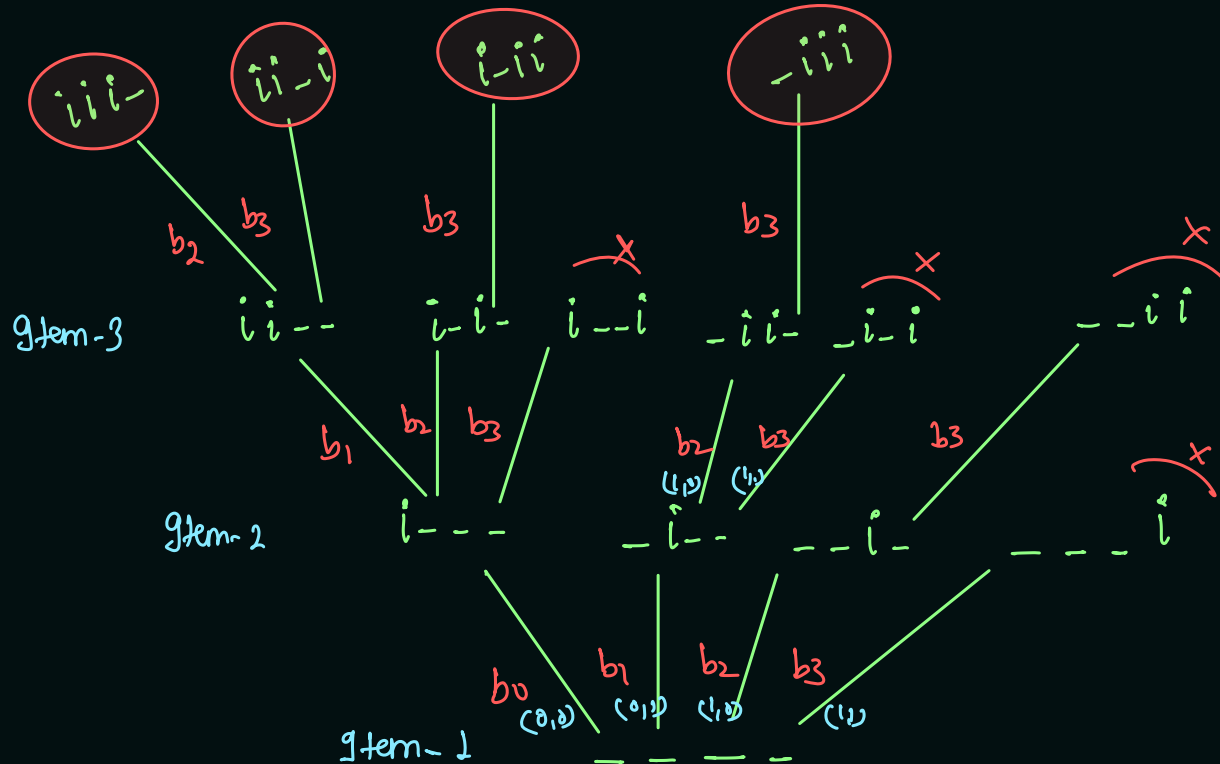
$r=2$ → items Result



n - identical items / given.

$${}^4C_3 = \frac{4!}{3! \times 1!} = \frac{4 \times 3 \times 2 \times 1}{3 \times 2 \times 1 \times 1} = 4$$
$$\gamma = 3 \rightarrow 9 \text{ km.}$$

Rem:-


$$\left\{ \begin{array}{cccc} i & i & i & - \\ i & i & - & i \\ i & - & i & i \\ - & i & i & i \end{array} \right.$$

Row
 $(0, 1)$
 $(0, -1)$

	0	1	2	3	4
0					
1					
2					
3					
4					

to present the
sorted order calling
protocol.

current Row
(1, 1) → Row

2, 0
↓
complete loop

// finish current row

```
for(int c = j + 1; c < chess.length; c++) {
    int r = 1;
    chess[r][c] = true;
    queensCombinations(qpsf + 1, tq, chess, r, c);
    chess[r][c] = false;
}
```

in next Row, from zeroth column

// finish remaining board

```
for(int r = i + 1; r < chess.length; r++) {
    for(int c = 0; c < chess.length; c++) {
        chess[r][c] = true;
        queensCombinations(qpsf + 1, tq, chess, r, c);
        chess[r][c] = false;
    }
}
```