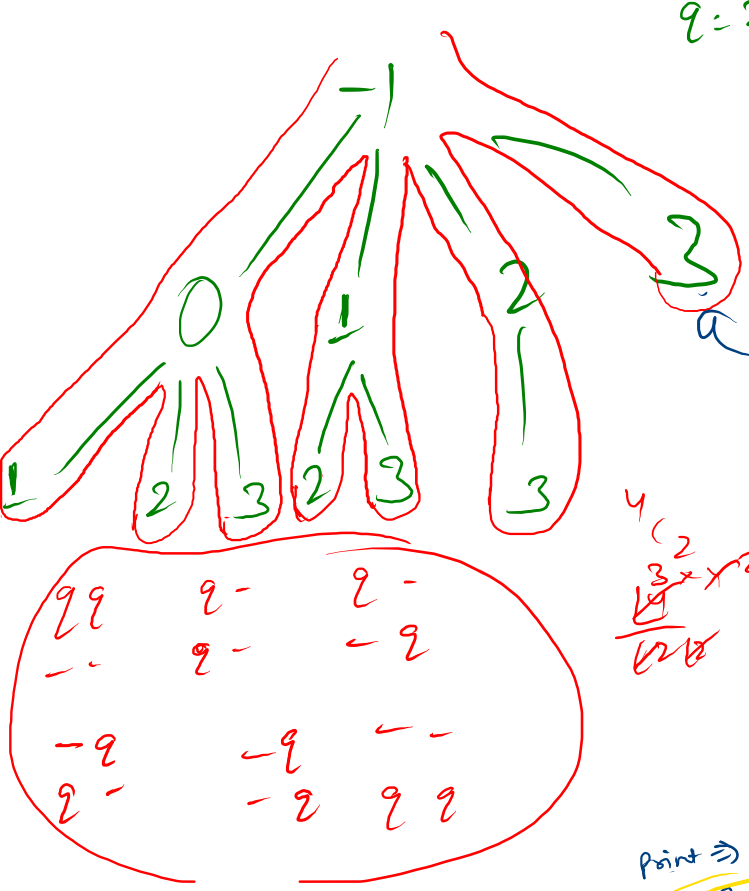
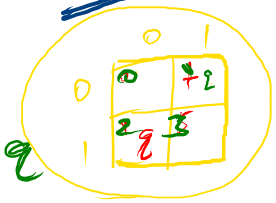


q = 2

arr



row = cellno / 4 (arr[0].length)

col = cellno % 4 (arr[0].length)

```
public static void queensCombinations(int qpsf, int tq, boolean[][] chess, in
no) {
    // write your code here
    if(qpsf == tq){
        for(int i=0;i<tq;i++){
            for(int j=0;j<tq;j++){
                if(chess[i][j] == true){
                    System.out.print("q\t");
                }else{
                    System.out.print("-\t");
                }
            }

            System.out.println();
        }

        System.out.println();
        return;
    }


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


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


Print ⇒ boxes

Permutation → q1 → q2 } boolean[] boxes

level → queen →

Combination → q → q

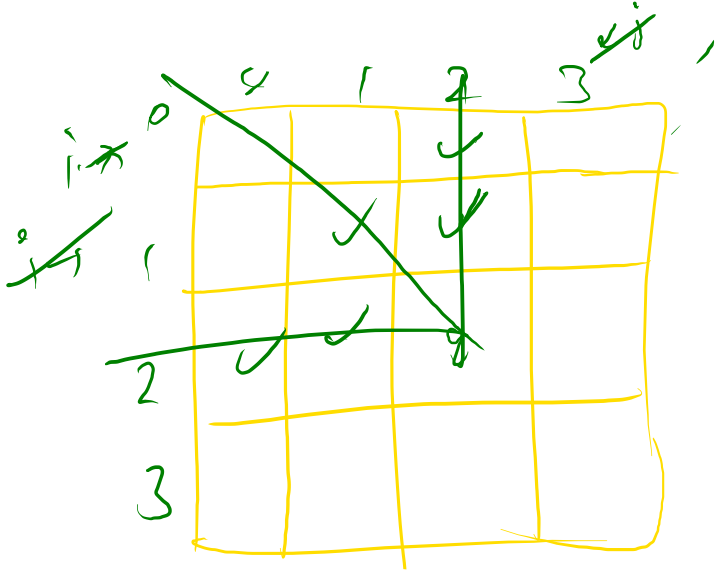
box →

Permutation → a, q1, q2, q3 {a|b}

boolean[] queens

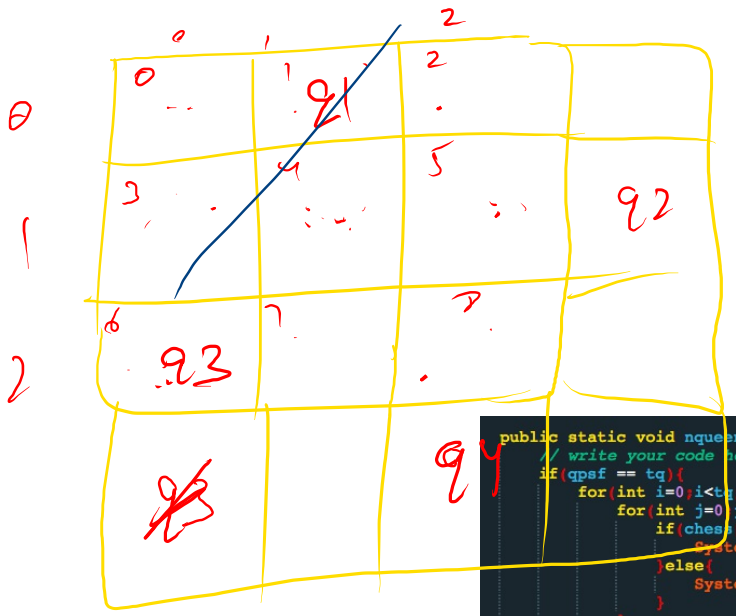
Combination → a, q → a|b

NQueen




```java
public static boolean IsQueenSafe(boolean[][] chess, int row, int col) {
    // write your code here

    for(int j = col-1;j>=0;j--){
        if(chess[row][j] == true){
            return false;
        }
    }


    for(int i = row-1;i>=0;i--){
        if(chess[i][col] == true){
            return false;
        }
    }

    for(int i = row-1,j = col-1;i>=0 && j>=0;i--,j--){
        if(chess[i][j] == true){
            return false;
        }
    }

    for(int i=row-1,j=col+1;i>=0&&j<chess.length;i--,j++){
        if(chess[i][j] == true){
            return false;
        }
    }
```

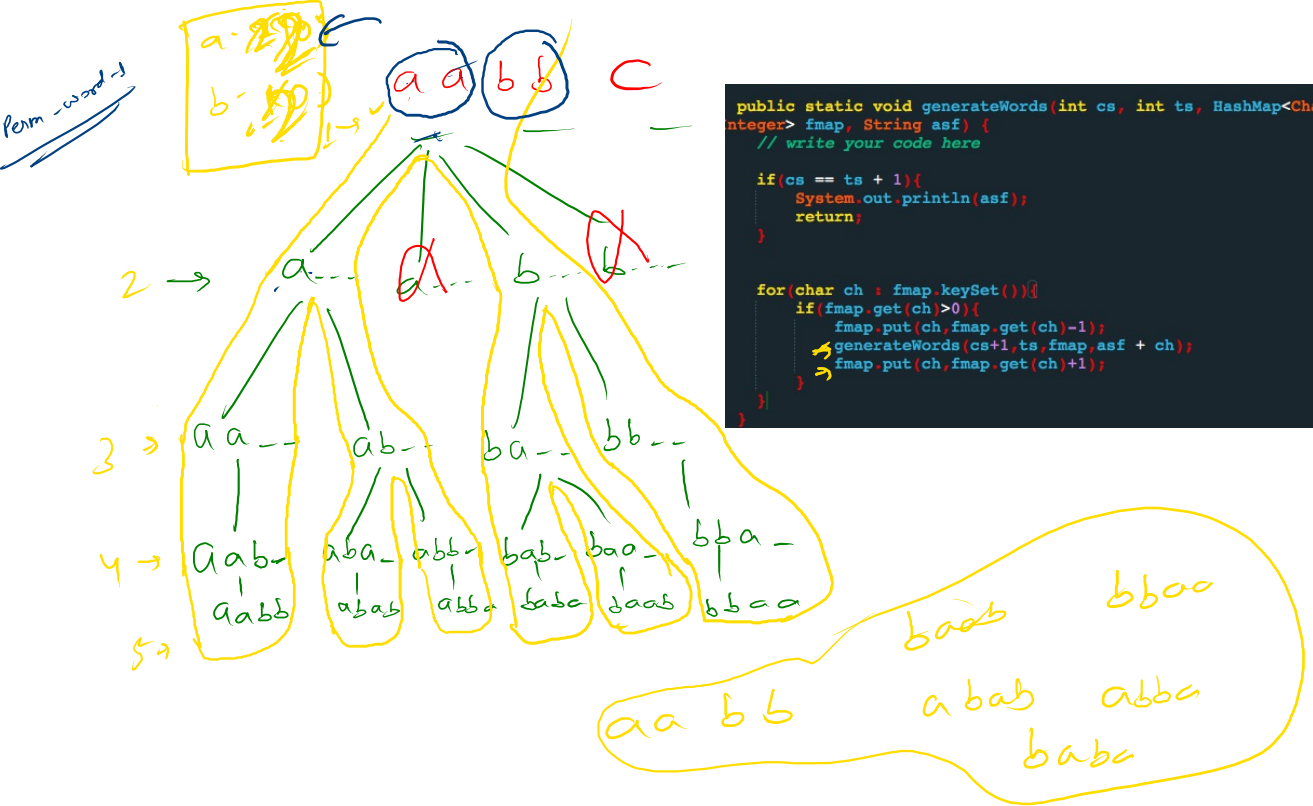
N-queen Rev



q1 q2 q3

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

        System.out.println();
        return;
    }

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

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

```
for(int j =0;j<n;j++){
    if(chess[row][j] != 0){
        return false;
    }
}
for(int i=0;i<n;i++){
    if(chess[i][col] != 0){
        return false;
    }
}
for(int i =row-1,j=col-1;i>=0&&j>=0;i--,j--){
    if(chess[i][j] != 0){
        return false;
    }
}
  for(int i =row+1,j=col+1;i<n&&j<n;i++,j++){
    if(chess[i][j] != 0){
        return false;
    }
}
  for(int i =row-1,j=col+1;i>=0&&j<n;i--,j++){
    if(chess[i][j] != 0){
        return false;
    }
}
  for(int i =row+1,j=col-1;i<n&&j>=0;i++,j--){
    if(chess[i][j] != 0){
        return false;
    }
}

return true;
```



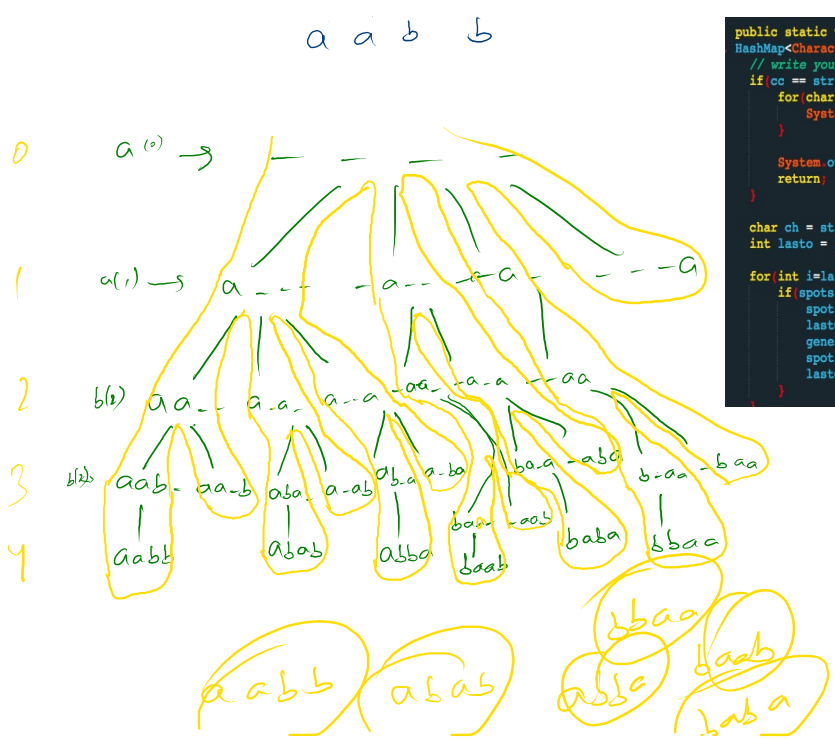

```
public static void generateWords(int cs, int ts, HashMap<Character,
Integer> fmap, String asf) {
    // write your code here

    if(cs == ts + 1){
        System.out.println(asf);
        return;
    }


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

2 9 9

5

1

0




```java
public static void generateWords(int cc, String str, Character[] sp
HashMap<Character, Integer> lastOccurence) {
  // write your code here
  if(cc == str.length()){
      for(char temp: spots){
          System.out.print(temp);
      }

      System.out.println();
      return;
  }

  char ch = str.charAt(cc);
  int lasto = lastOccurence.get(ch);

  for(int i=lasto+1;i<str.length();i++){
      if(spots[i] == null){
          spots[i] = ch;
          lastOccurence.put(ch,i);
          generateWords(cc+1,str,spots,lastOccurence);
          spots[i] = null;
          lastOccurence.put(ch,lasto);
      }
  }
```