① Flood fill.

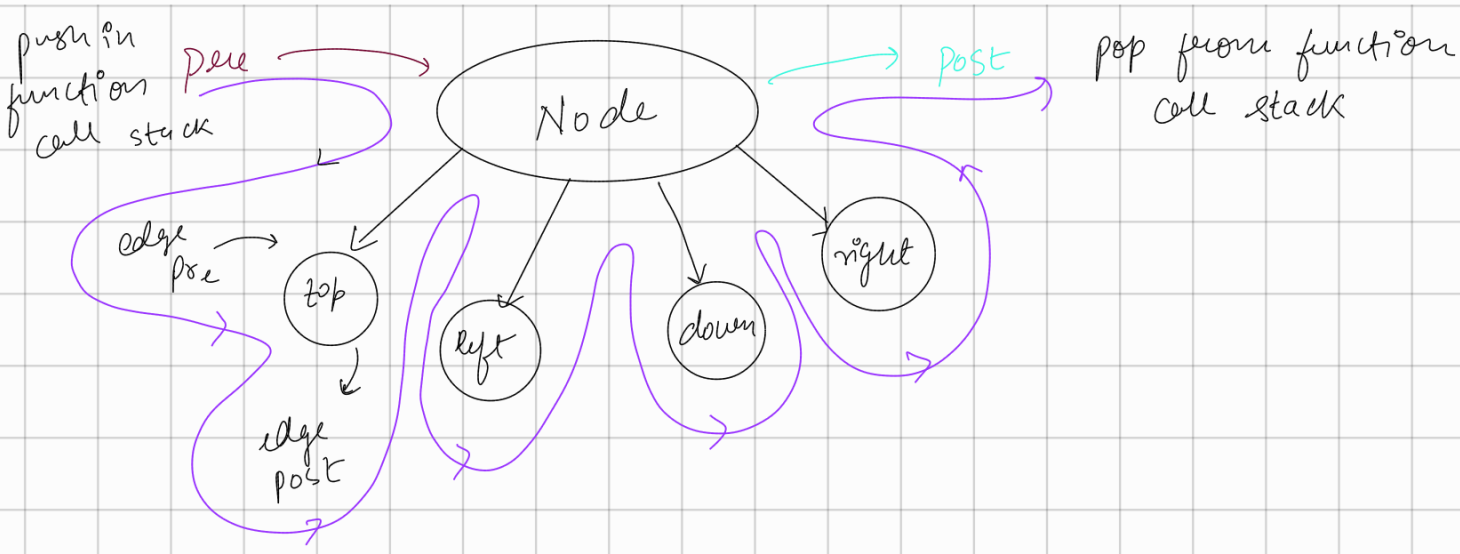
CODE:

```
flood fill( int[][] maze, int sr, int sc, String psf, boolean[][] vis){
    int dr= maze. length -1, dc = maze[0]. length -1;
    if ( sr >dr || sc >dc || sr <0 || sc <0 || maze[sr][sc]==1 || vis[sr][sc] == true){
        // negative base case
        return;
    }

    vis [sr] [sc] = true;
    if ( sr == dr && sc == dc){
        // positive base case
        System. out. println( psf);
        vis [sr] [sc] = false;
        return;
    }

    floodfill ( maze, sr-1, sc, psf + "t" , vis) ;
    floodfill (maze, sr, sc-1, psf+"l" , vis);
    floodfill (maze , sr+1 , sc, psf+ "d", vis);
    floodfill ( maze, sr, sc+1 , psf+ "r" , vis);
    vis [sr][sc] = false;                    // backtracking
}
```

visited array facilitates cycle detection




★ when to use backtracking

- Print all paths
- " configurations
- " permutations & combinations

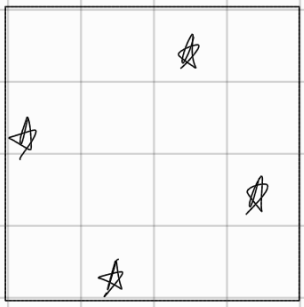
- Count all paths
  " configurations } dp/greedy
  " P&C

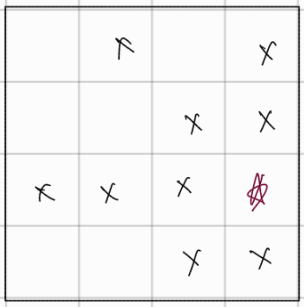
from constraints.

Recursion & Backtracking ⇒ very small values, because exponential TC.

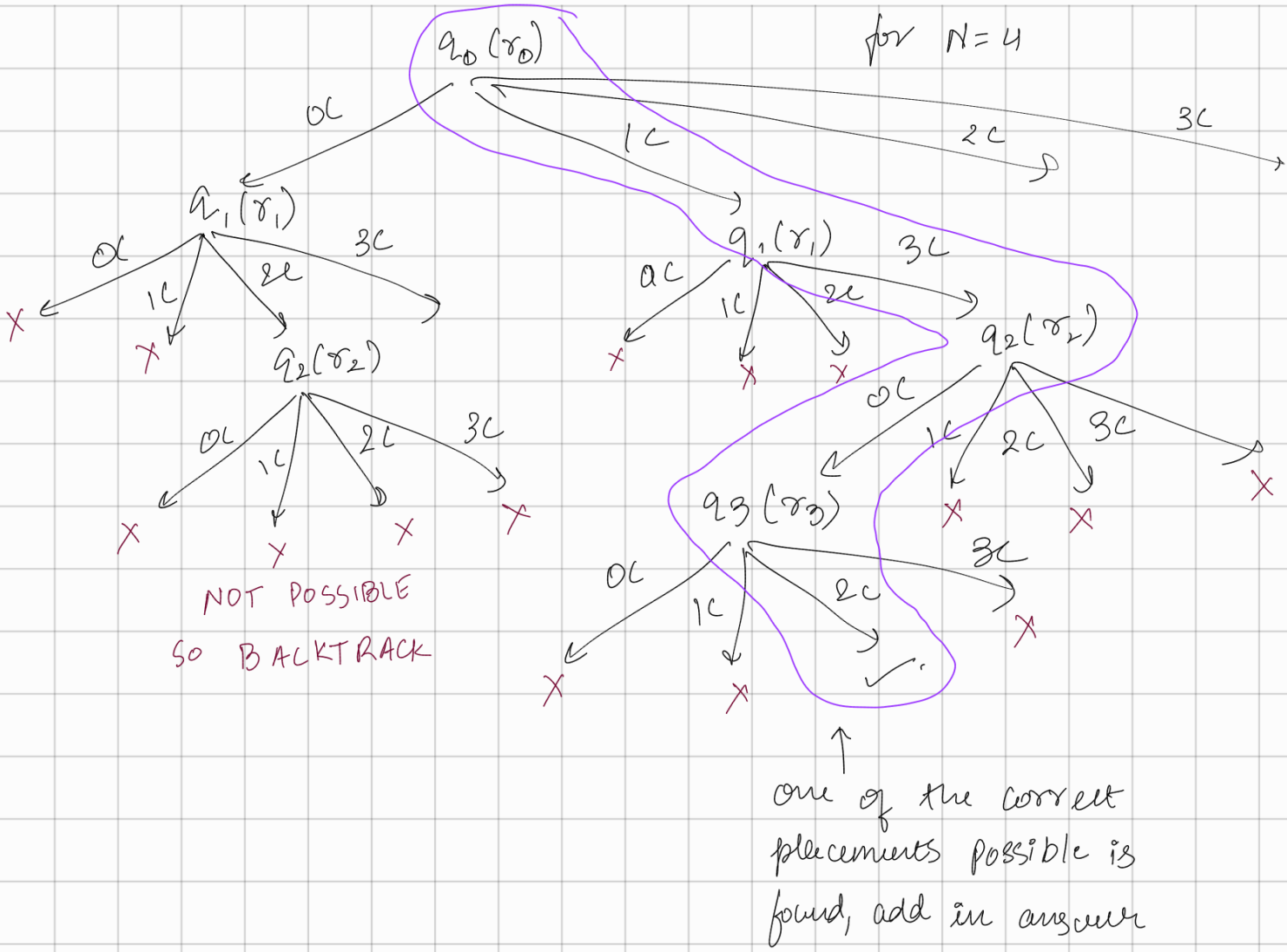
② N-Queen Problem

for N = 4,





print all configurations of n queens in n×n matrix such that none kills any other




∴ CODE:

```
printNQueens(boolean[][] chess, String qsf, int row) {
    if(row == chess.length) {
        System.out.println(qsf);
        return;
    }

    for(int col = 0; col < chess[0].length; col++) {
        if(isSafe(chess, row, col) == true) {
            chess[row][col] = true;
            printNQueens(chess, qsf + row + "-" + col + ", ", row+1);
            chess[row][col] = false;
        }
    }
}
```

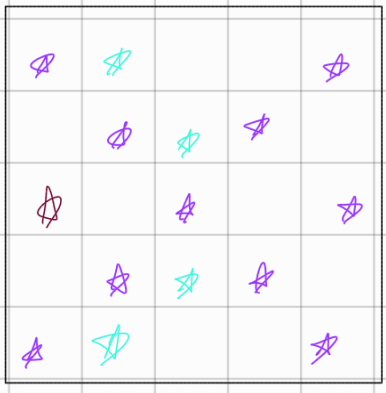
```
boolean isSafe(boolean[][] chess, int row, int col){
    int n = chess.length;
    for(int i=0; i<n; i++)
        if(chess[i][col] == true)
            return false;

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

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

    return true;
}
```

③ Knight's Tour




without repeating

Kings Tour

| | |
|---|---|
| 1 | 2 |
| 3 | 4 |

1 2 / 3 4

1 2 / 4 3

1 4 / 2 3

1 3 / 2 4

Faith → clockwise