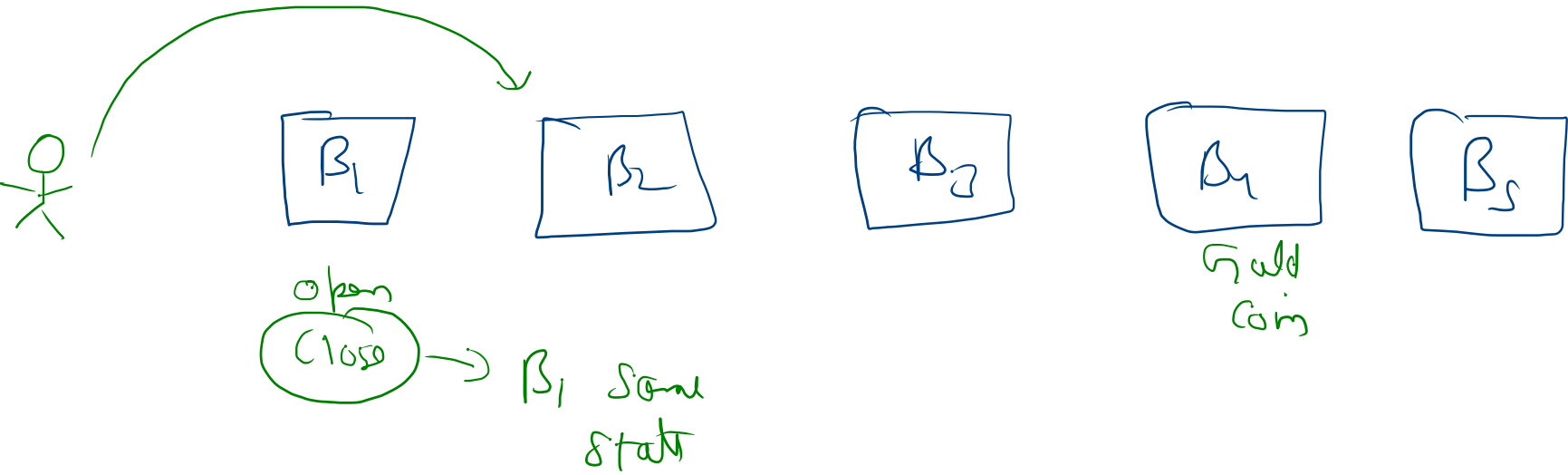
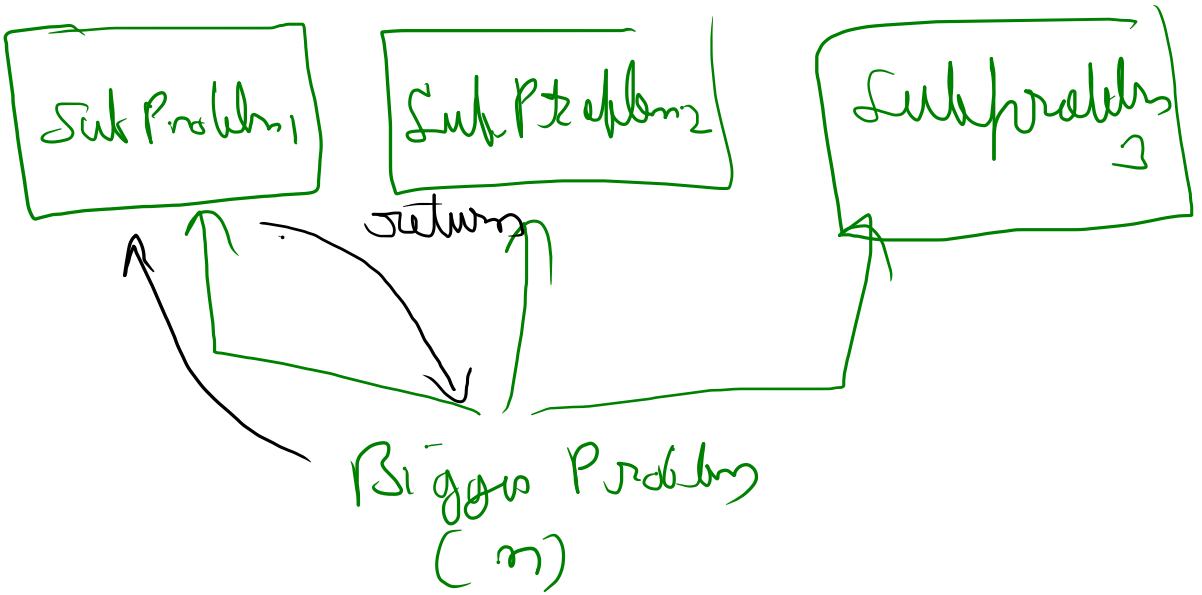
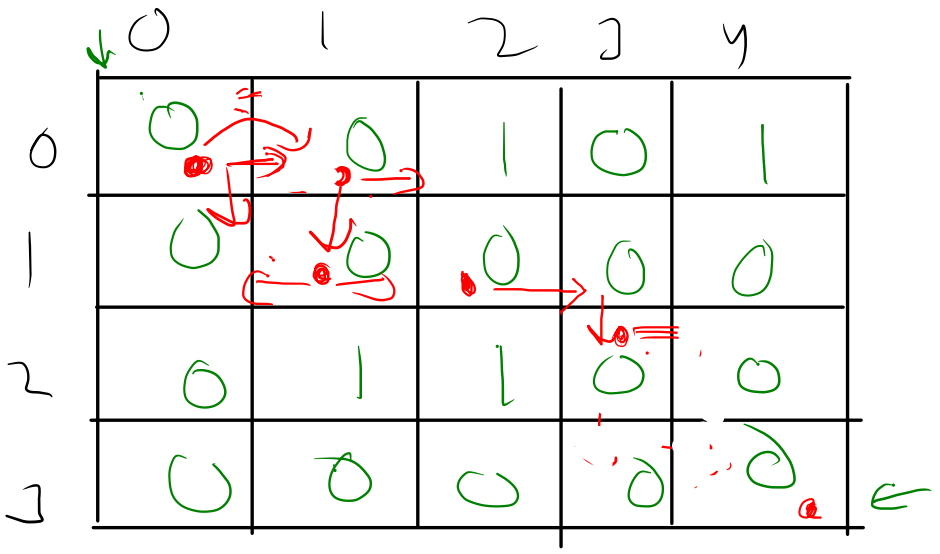
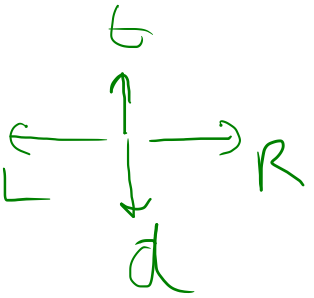
Recursion Backtracking




Sub Problem 1
Sub Problem 2
Sub problem 3
return
Bigger Problem
(n)

Flood Fill (Day 27)

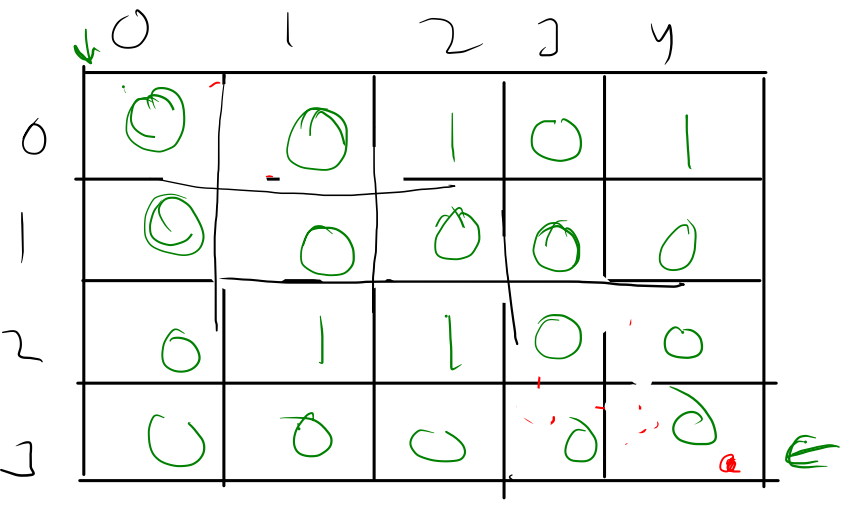


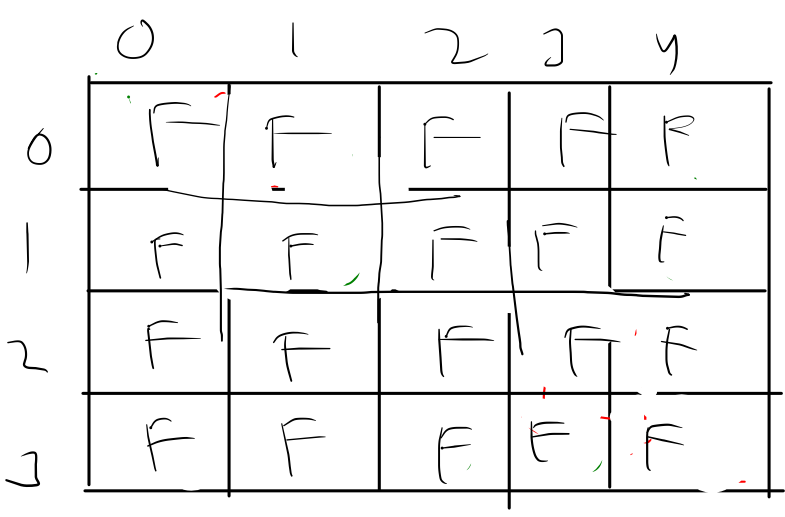


r d r r r d d r

r d r r r d r d

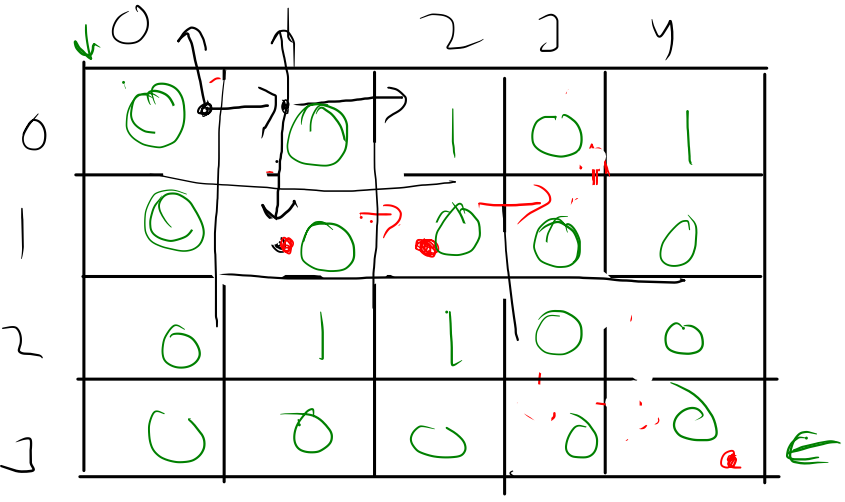
Visited arr





False → Node never visited arr

True → " already visited

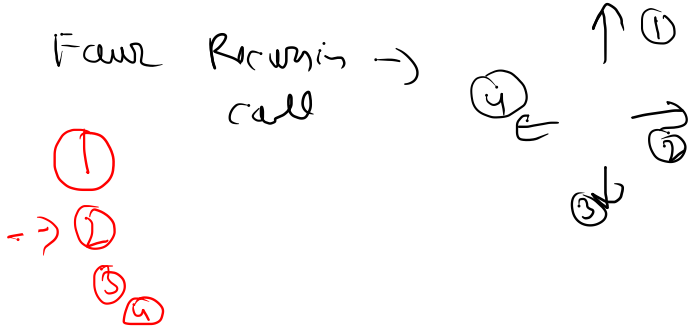


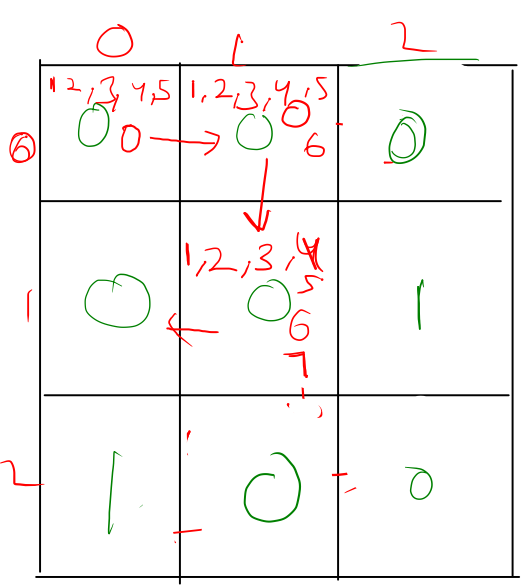
Visited arr

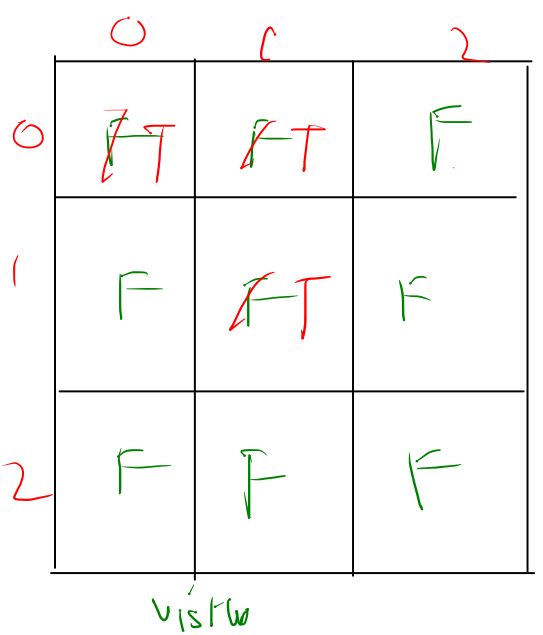
	0	1	2	3	4
0	F T	F T	F	F	F
1	F	F T	F T	F	F
2	F	F	F	F	F
3	F	F	F	F	F

if (in Maze && all == 0 &&
visited [i] [j] == false)

Four Recursion call

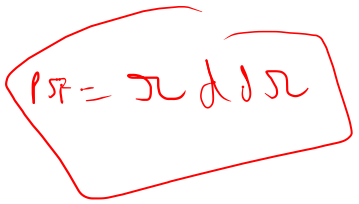




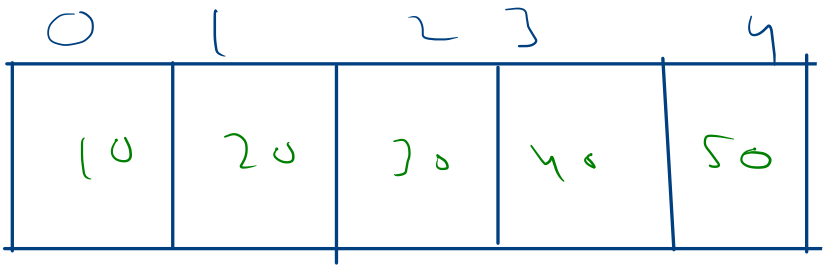



```java
public static void solution(int[][] arr, String psf, int i, int j,

    if (i < 0 || j < 0 || i ≥ arr.length || j ≥ arr[0].length ||
        arr[i][j] == 1 || visited[i][j] == true) {
        return;
    }
    if (i == arr.length - 1 && j == arr[0].length - 1) {
        System.out.println(psf);
        return;
    }
    visited[i][j] = true;
    // top call
    solution(arr, psf: psf + "t", i: i - 1, j, visited);
    // right call
    solution(arr, psf: psf + "r", i, j: j + 1, visited);
    // down call
    solution(arr, psf: psf + "d", i: i + 1, j, visited);
    // left call
    solution(arr, psf: psf + "l", i, j: j - 1, visited);
    // backtracking step
    visited[i][j] = false;
```



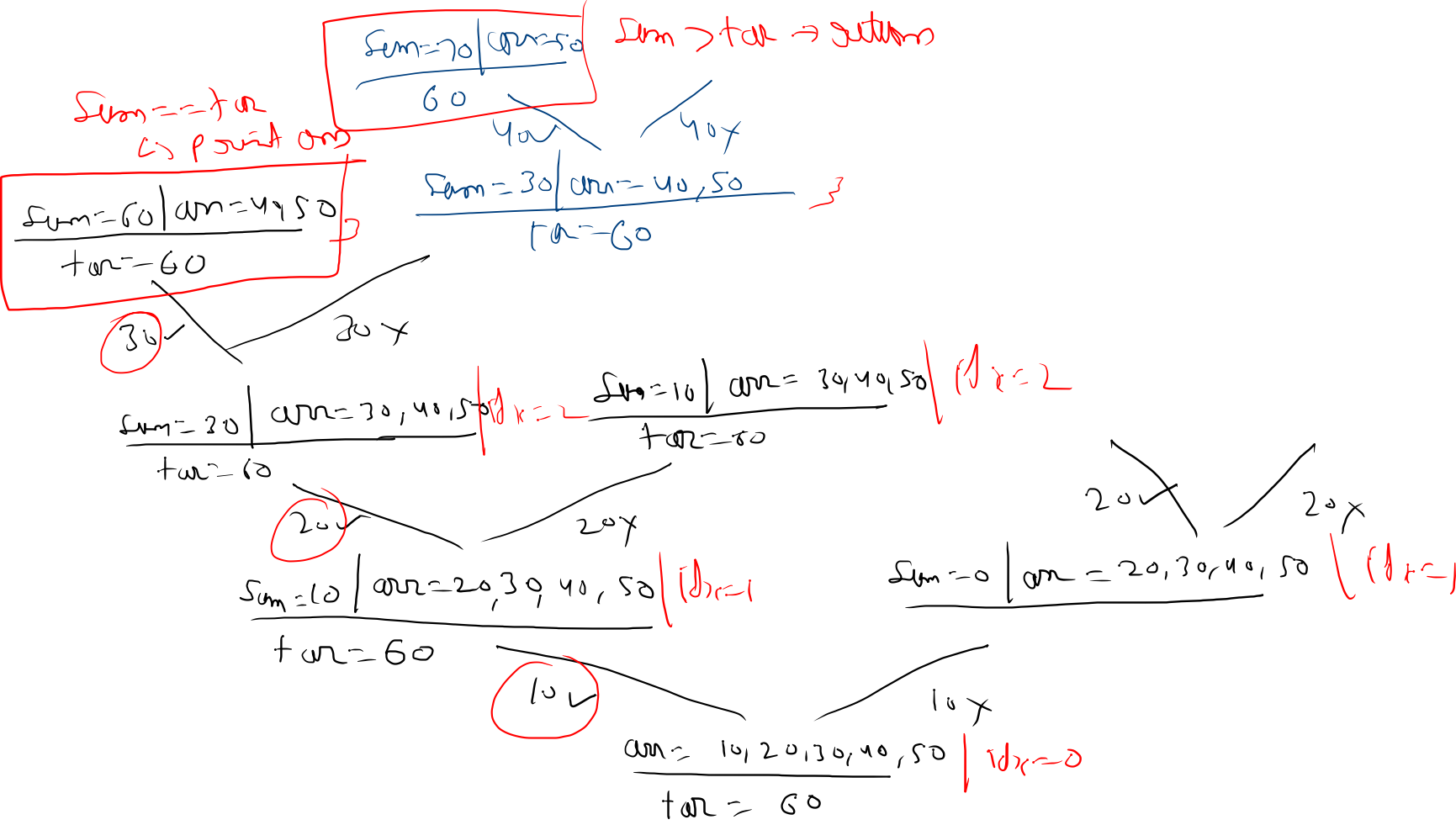
Target Sum Subsets (Day 27)



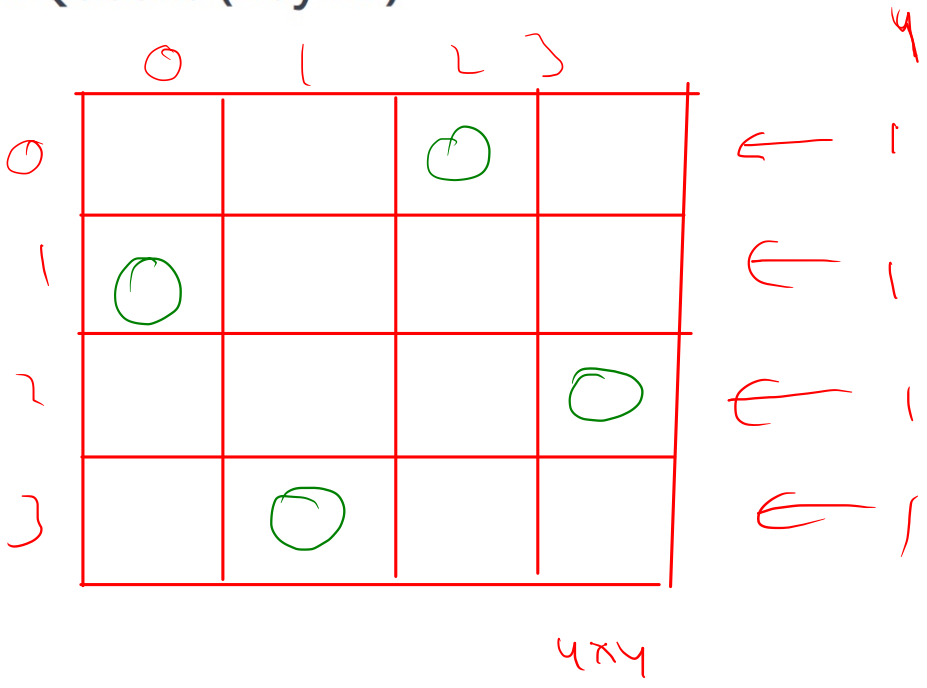
tar = 60

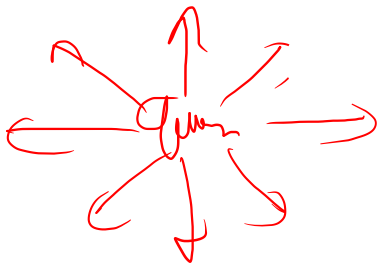
10 20 30
10 50
20 40

] 5 subsets


sum=70 | arr=50
60
sum > tar → subtract
40✓
40✗
sum=30 | arr=40,50
tar=60
sum==tar
is present only
sum=60 | arr=40,50
tar=60
30✓
30✗
sum=30 | arr=30,40,50
tar=60
idx=2
sum=10 | arr=30,40,50
tar=60
idx=2
20✓
20✗
sum=10 | arr=20,30,40,50
tar=60
idx=1
20✓
20✗
sum=0 | arr=20,30,40,50
idx=1
10✓
10✗
arr=10,20,30,40,50
tar=60
idx=0

N Queens (Day 27)





row = 6