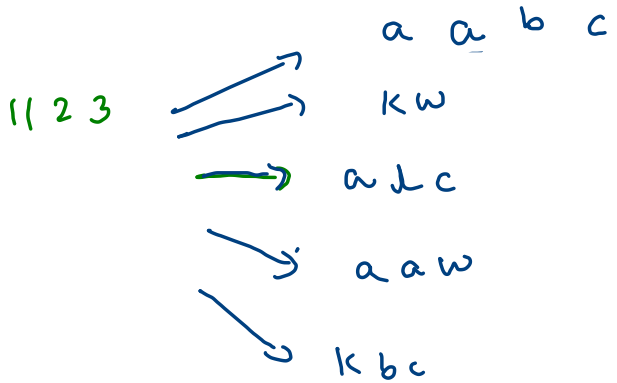
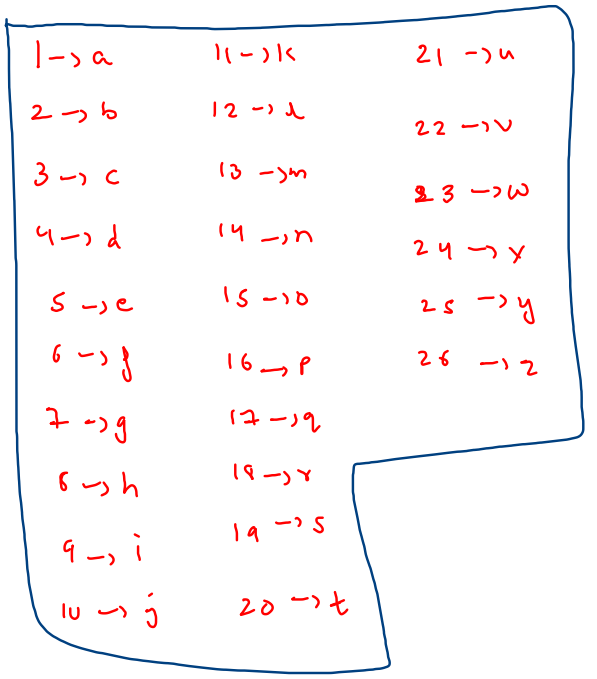
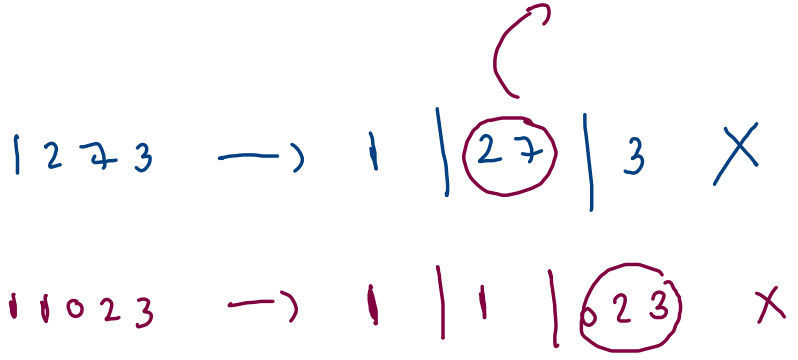
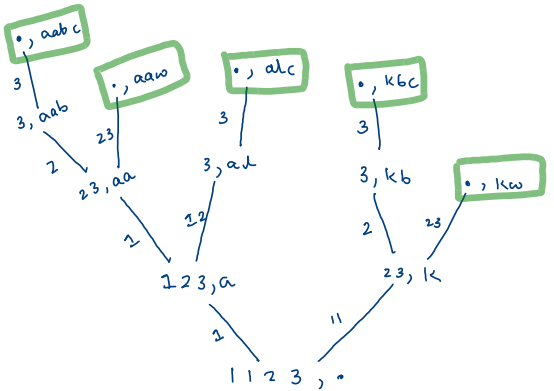

a a b c
1123
kw
akc
aaw
kbc


1 -> a
2 -> b
3 -> c
4 -> d
5 -> e
6 -> f
7 -> g
8 -> h
9 -> i
10 -> j
11 -> k
12 -> l
13 -> m
14 -> n
15 -> o
16 -> p
17 -> q
18 -> r
19 -> s
20 -> t
21 -> u
22 -> v
23 -> w
24 -> x
25 -> y
26 -> z


1273 -> 1 | 27 | 3 X
11023 -> 1 | 1 | 23 X

1 -> a
2 -> b
3 -> c
4 -> d
5 -> e
6 -> f
7 -> g
8 -> h
9 -> i
10 -> j
11 -> k
12 -> l
13 -> m
14 -> n
15 -> o
16 -> p
17 -> q
18 -> r
19 -> s
20 -> t
21 -> u
22 -> v
23 -> w
24 -> x
25 -> y
26 -> z



1, 2, 3, 4 9

enc -> ch - '0' - 1 + 'a' -> ch is a character

10, 11, 12 26

str -> "12"

int up -> str[1] - '0'

int tp -> str[0] - '0'

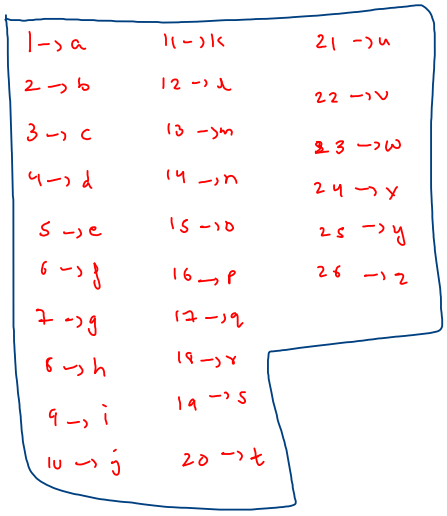
val -> tp * 10 + up

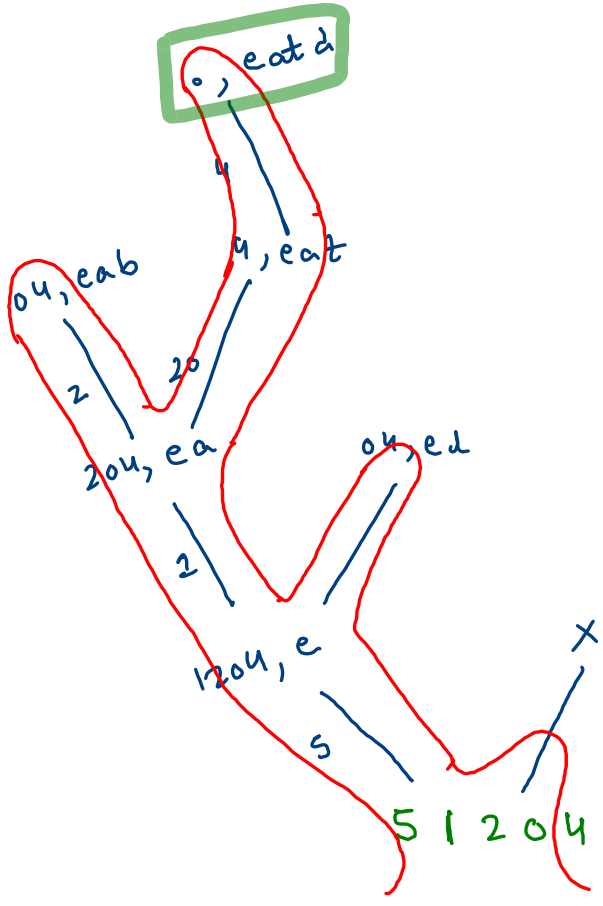
up -> 2

tp -> 1

1 * 10 + 2 = 12

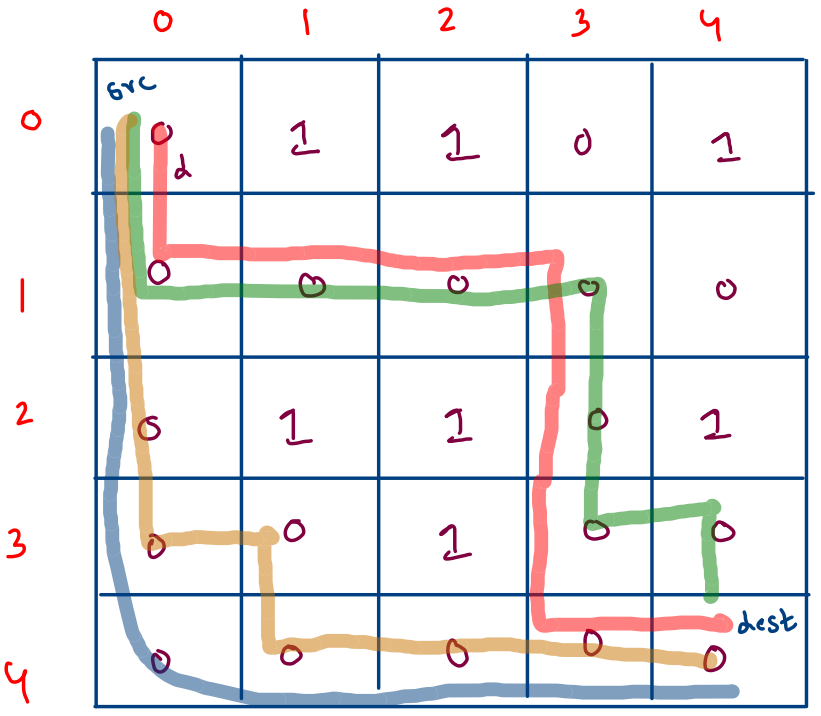
enc = val - 1 + 'a'

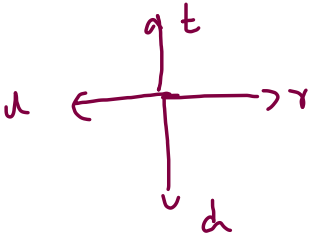




eatd

```java
public static void printEncodings(String str,String asf) {
    if(str.length() == 0) {
        System.out.println(asf);
        return;
    }

    char ch = str.charAt(0);

    if(ch == '0') {
        return;
    }


    //single
    String ros = str.substring(1);
    char enc = (char)(ch - '0' - 1 + 'a');
    printEncodings(ros,asf + enc);

    //clubbing of two chars
    if(str.length() >= 2) {
        int tp = str.charAt(0) - '0';
        int up = str.charAt(1) - '0';

        int val = tp*10 + up;

        if(val >= 10 && val <= 26) {
            String ros2 = str.substring(2);
            char enc2 = (char)(val - 1 + 'a');
            printEncodings(ros2,asf + enc2);
        }
    }
```

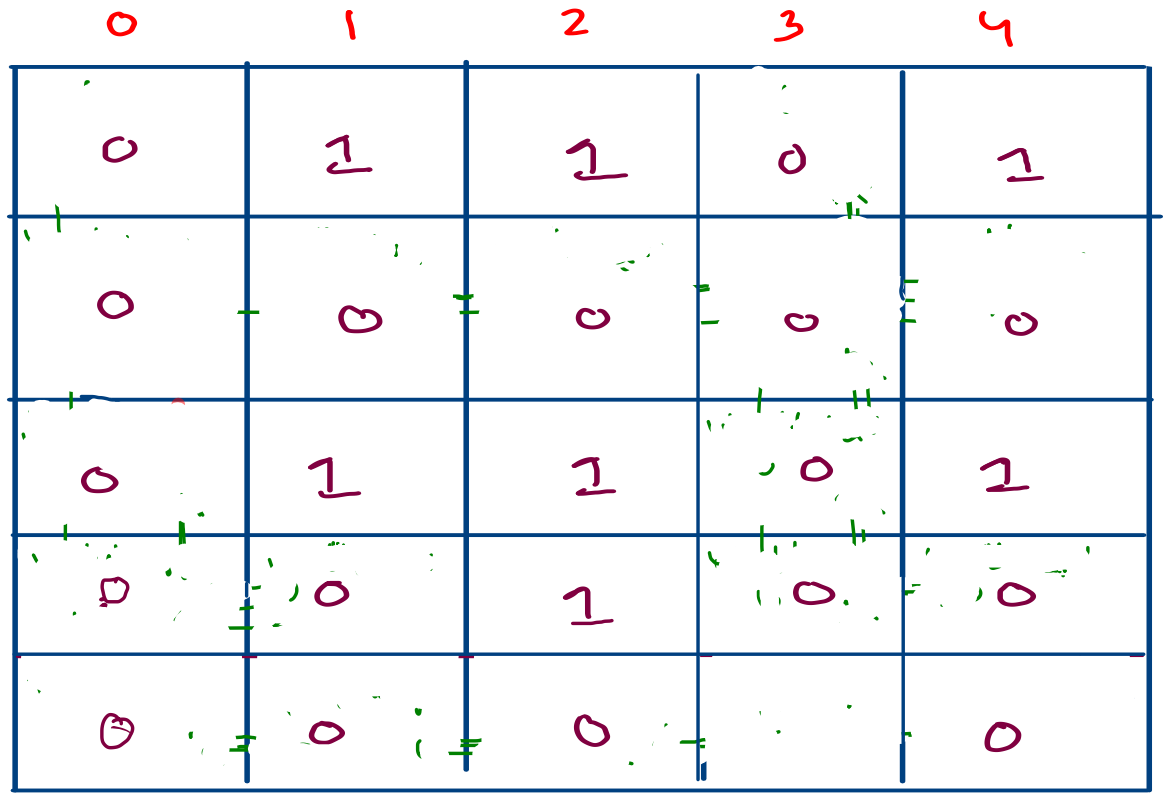


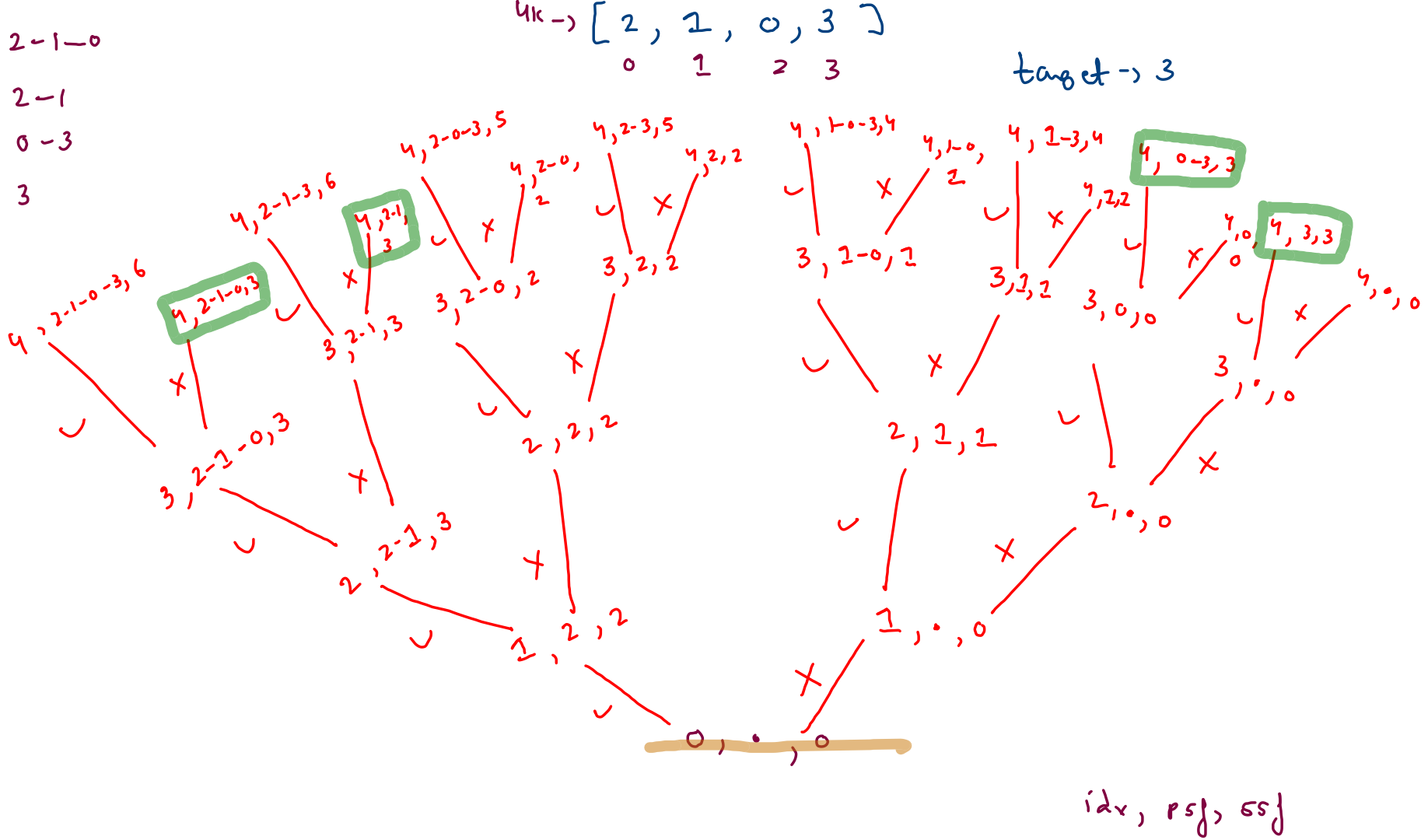


0 -> move

1 -> obstacle

```java
public static void floodfill(int[][] maze, int sr, int sc, String asf,boolean[][]vis) {
    if(sr == maze.length-1 && sc == maze[0].length-1) {
        System.out.println(asf);
        return;
    }

    if(sr < 0 || sc < 0 || sr >= maze.length || sc >= maze[0].length || maze[sr][sc] == 1 || vis[sr][sc] == true) {
        return;
    }

    vis[sr][sc] = true;

    //top
    floodfill(maze,sr-1,sc,asf + "t",vis);

    //left
    floodfill(maze,sr,sc-1,asf + "l",vis);

    //down
    floodfill(maze,sr+1,sc,asf + "d",vis);

    //right
    floodfill(maze,sr,sc+1,asf + "r",vis);

    vis[sr][sc] = false;
}
```




2-1-0
2-1
0-3
3
nk -> [2, 1, 0, 3]
0 1 2 3
target -> 3
4, 2-1-0-3, 6
4, 2-1-0, 3
4, 2-1-3, 6
3, 2-1, 3
4, 2-0-3, 5
4, 2-1, 3
4, 2-0, 2
3, 2-0, 2
4, 2-3, 5
4, 2, 2
3, 2, 2
2, 2, 2
3, 2-1-0, 3
2, 2-1, 3
1, 2, 2
0, •, 0
4, 1-0-3, 4
4, 1-0, 2
3, 1-0, 1
4, 1-3, 4
4, 2, 2
3, 1, 2
2, 1, 1
1, •, 0
4, 0-3, 3
4, 0, 0
3, 0, 0
2, •, 0
4, 3, 3
4, •, 0
3, •, 0
idx, psf, ssf

tar -> 3 [4, 1, 0, 2]

```java
public static void printTargetSumSubsets(int[] arr, int idx, String set, int sos, int tar) {
    if(idx == arr.length) {
        if(sos == tar) {
            System.out.println(set + ".");
        }
        return;
    }

    if(sos > tar) {
        return;
    }

    //arr[idx] -> present
    printTargetSumSubsets(arr,idx+1,set + arr[idx] + ", ",sos + arr[idx],tar);

    //arr[idx] -> absent
    printTargetSumSubsets(arr,idx+1,set,sos,tar);
}
```