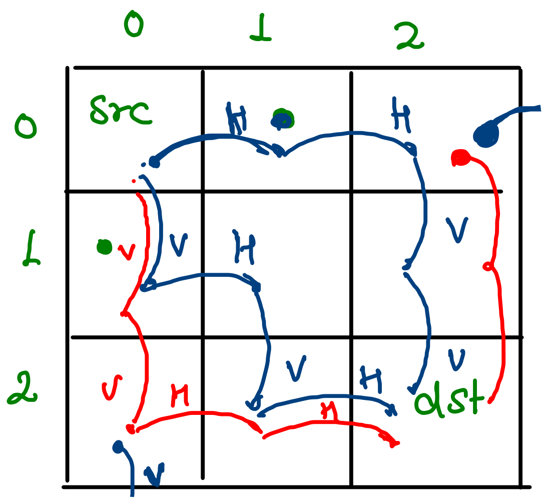
Get Maze paths

Initial point
(0, 0)
Destination point
(n-1, m-1)
Return all possible path from source to destination with allowed moves in arraylist.



Example

for some paths.

① H H V V

② V V H H

③ V H V H

⋮ all paths.

Rules → Horizontal (Right)
vertical (down)

Expectation. all paths (0,0, 2,2) → all paths from src to dst

faith. all paths (0,1, 2,2) → 0,1 to 2,2 allpath

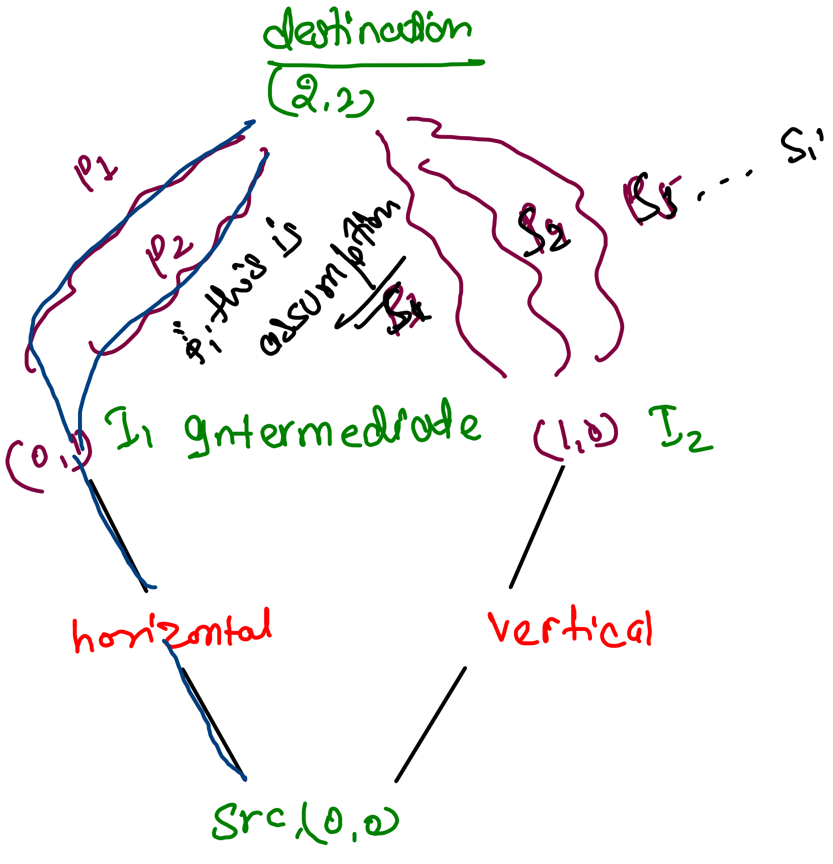
allpath (1,0,2,2) → 1,0 to 22 allpath

Merging

hres = allpath(0,1,2,2);

vres = allpath(1,0,2,2);

mres - {"H" + hres, 'V' + vres} =



jump, path.

$$\begin{tikzcd}
(r,c) \arrow[r, "\text{Horizontal}"] \arrow[d, "\text{vertical}"] & (r,c+1) \\
(r+1,c) &
\end{tikzcd}$$

$\boxed{c+1}$ i dc

No. of path from src to dst.

Sum of path from all's intermediate.

= I_1 paths + I_2 paths

= 2+3

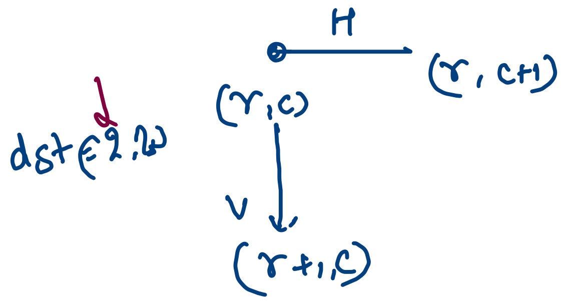
= 5

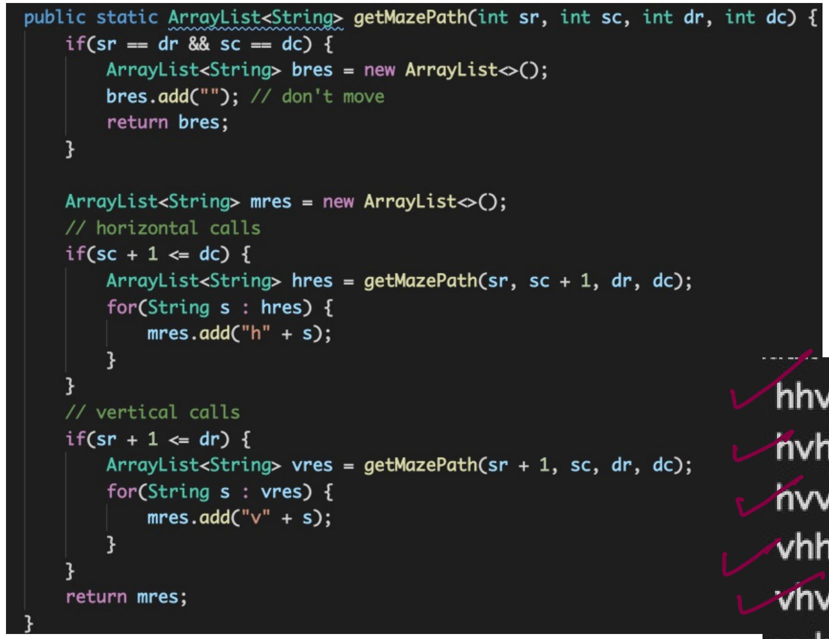
H P1
H P2

V P3
V P4
V P5

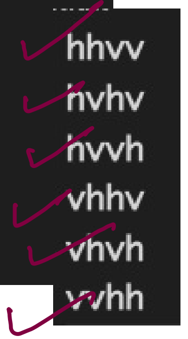
intermediate valid
→ call - smart
→ base case - fool base case.

→ call - stupid
→ base case - smart
↳ Return from invalid intermediate

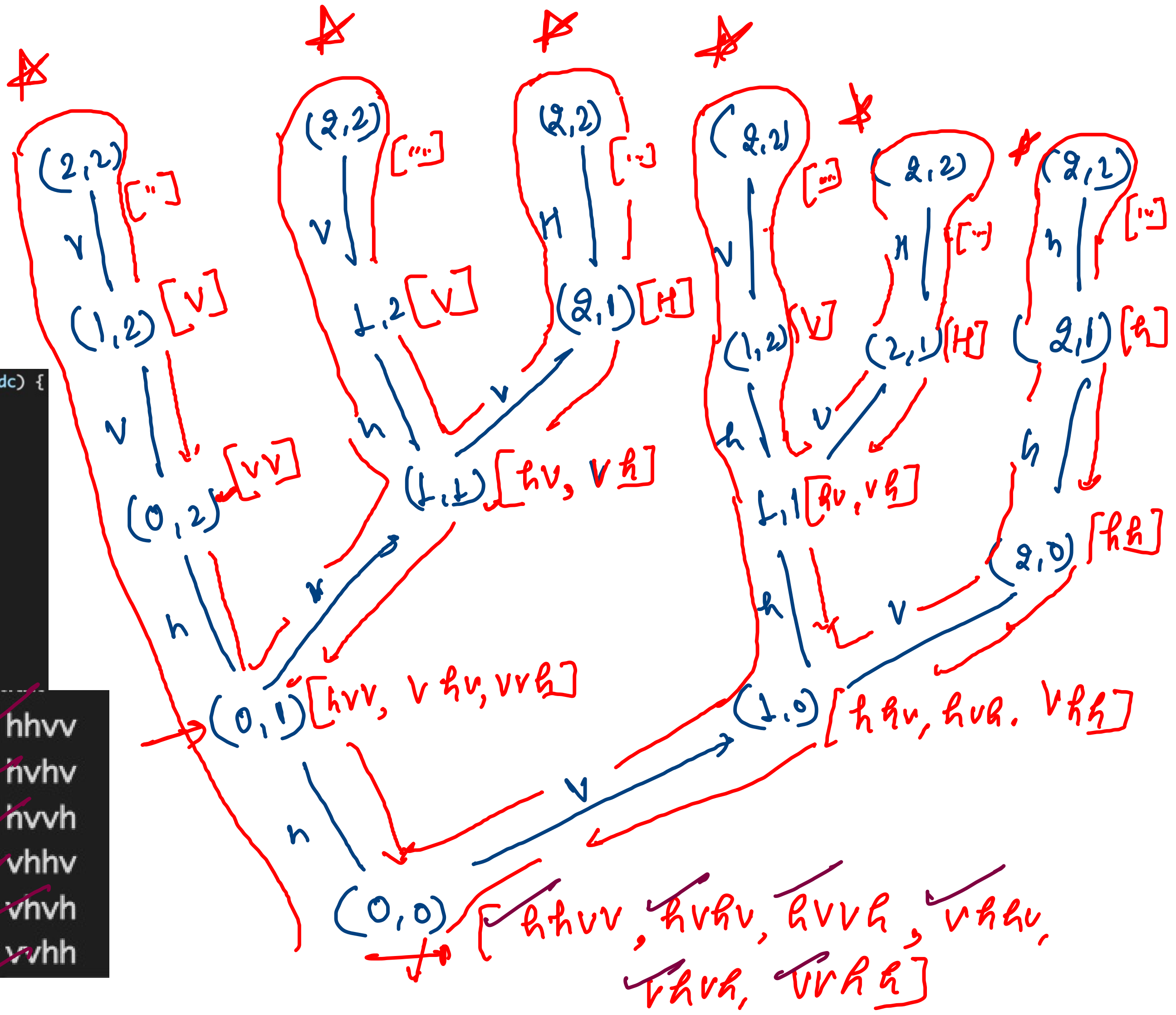




```java
public static ArrayList<String> getMazePath(int sr, int sc, int dr, int dc) {
    if(sr == dr && sc == dc) {
        ArrayList<String> bres = new ArrayList<>();
        bres.add(""); // don't move
        return bres;
    }

    ArrayList<String> mres = new ArrayList<>();
    // horizontal calls
    if(sc + 1 <= dc) {
        ArrayList<String> hres = getMazePath(sr, sc + 1, dr, dc);
        for(String s : hres) {
            mres.add("h" + s);
        }
    }
    // vertical calls
    if(sr + 1 <= dr) {
        ArrayList<String> vres = getMazePath(sr + 1, sc, dr, dc);
        for(String s : vres) {
            mres.add("v" + s);
        }
    }
    return mres;
}
```

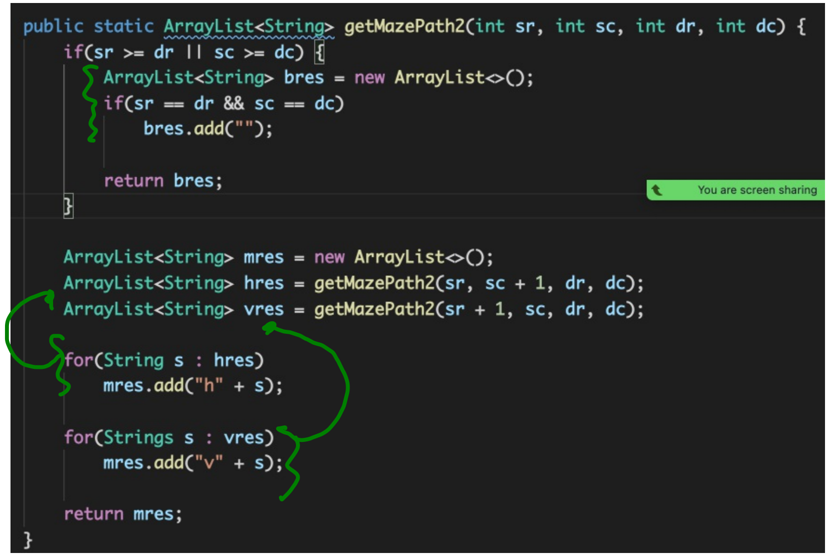


```
hhvv
hvhv
hvvh
vhhv
vhvh
vvhh
```

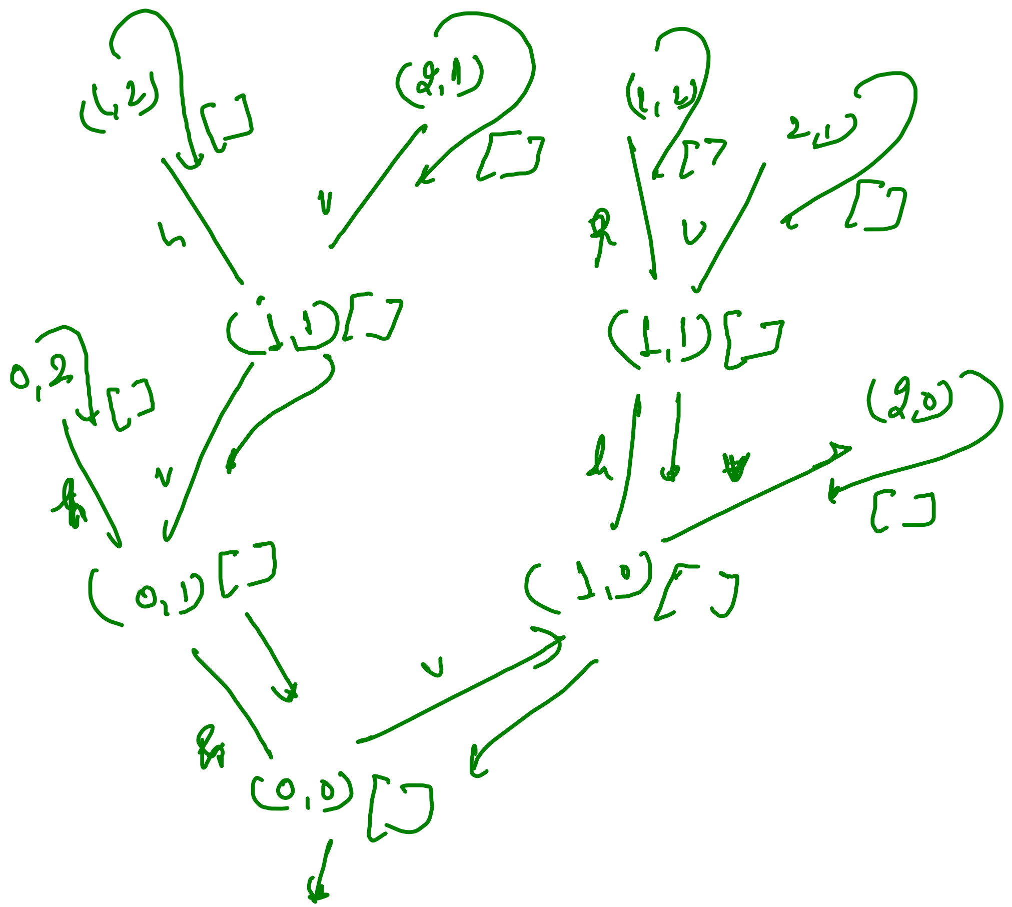


$$\begin{tikzcd} (r,c) \arrow[r, "h"] \arrow[d, "v"] & (r,c+1) \\ (r+1,c) \end{tikzcd}$$

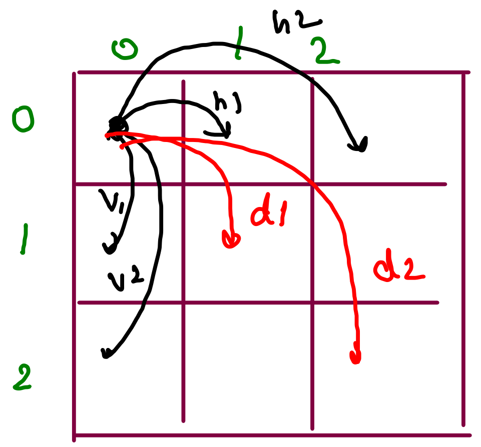
Invalid base case

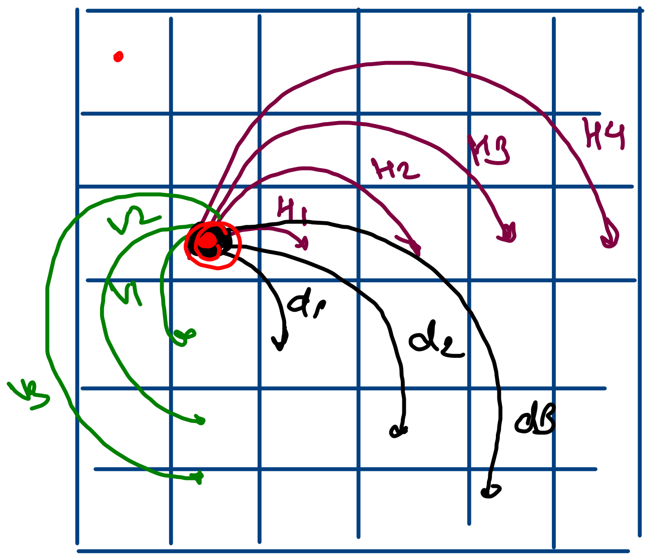


```
public static ArrayList<String> getMazePath2(int sr, int sc, int dr, int dc) {
    if(sr >= dr || sc >= dc) {
        ArrayList<String> bres = new ArrayList<>();
        if(sr == dr && sc == dc)
            bres.add("");

        return bres;
    }

    ArrayList<String> mres = new ArrayList<>();
    ArrayList<String> hres = getMazePath2(sr, sc + 1, dr, dc);
    ArrayList<String> vres = getMazePath2(sr + 1, sc, dr, dc);

    for(String s : hres)
        mres.add("h" + s);

    for(Strings s : vres)
        mres.add("v" + s);

    return mres;
}
```



Get Mazepath with jumps →

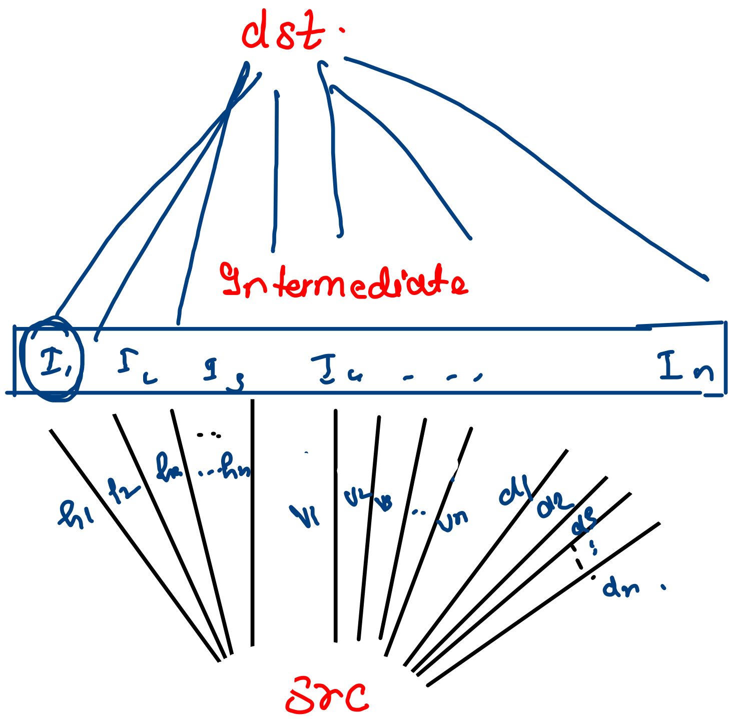


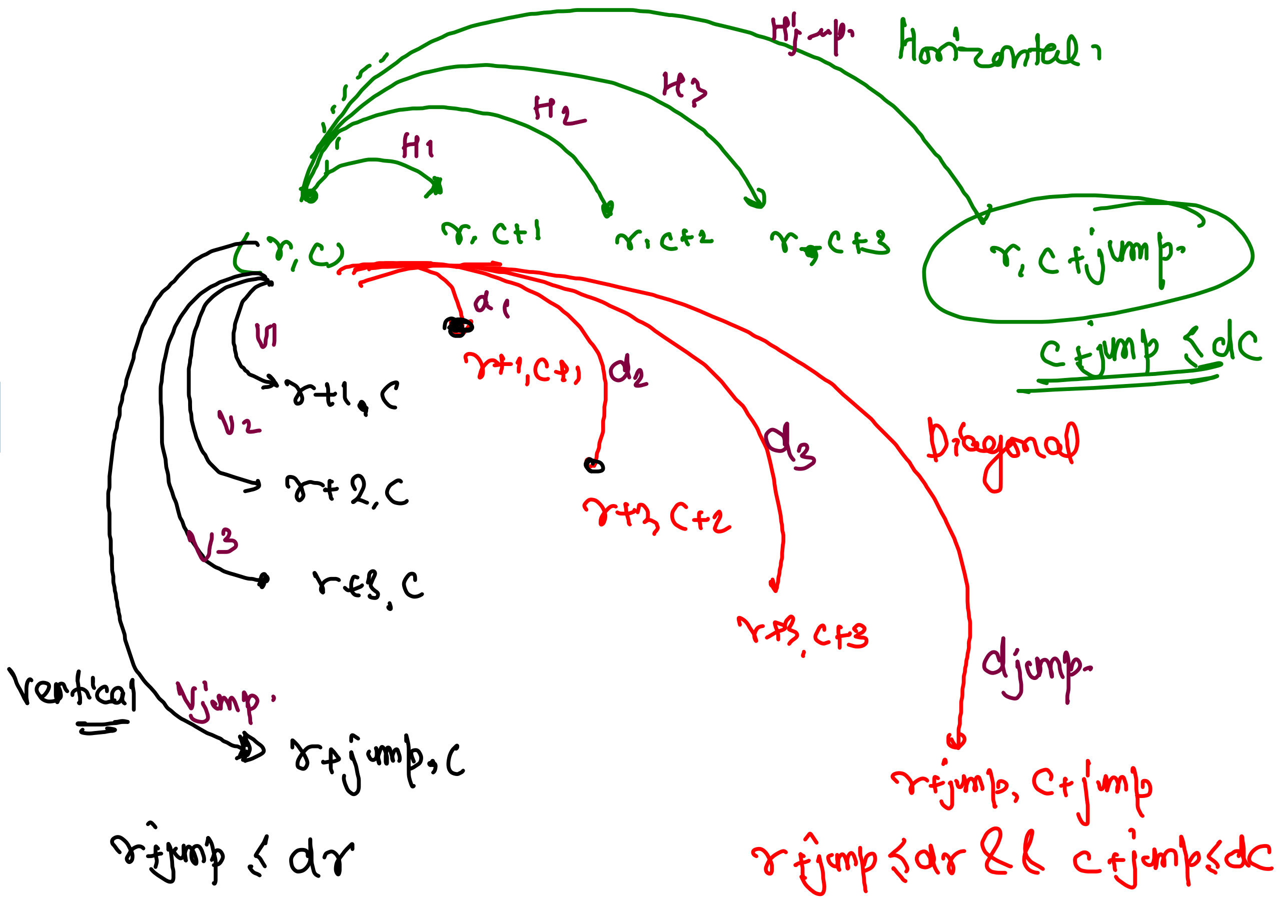


Return all path from source (0,0) to destination (2,2) with following movs.

Horizontal – Right
Vertical – Down.
Diagonal – Down Right

Order → Horizontal
Vertical
Diagonal.


dst.
Intermediate
I_1 I_2 I_3 I_4 ... I_n
h_1 h_2 h_3 ... h_n
v_1 v_2 v_3 ... v_n
d_1 d_2 d_3 ... d_n
src

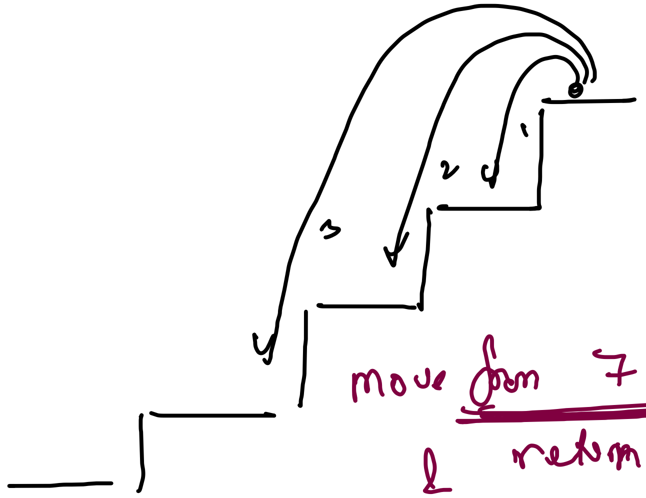


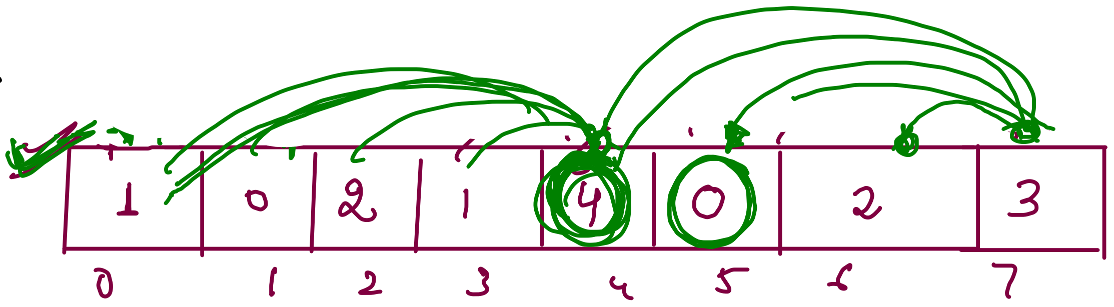
for n^{th} jump $\longrightarrow$ loop of jump.

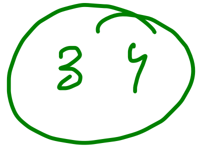
jump = 1 ← ... Condition depend on type.

$\boxed{j{+}{+}}$

Stair path with jumps →







move from 7 to 0

& return all

possible

path