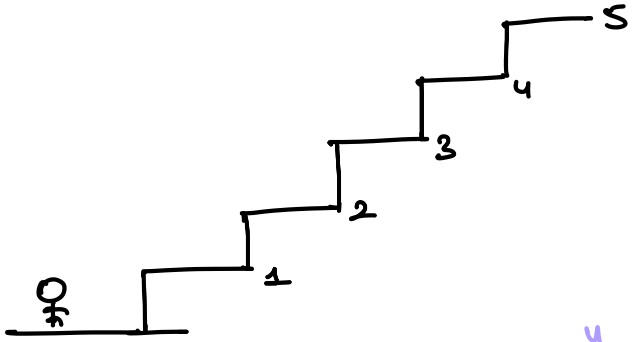
... contd

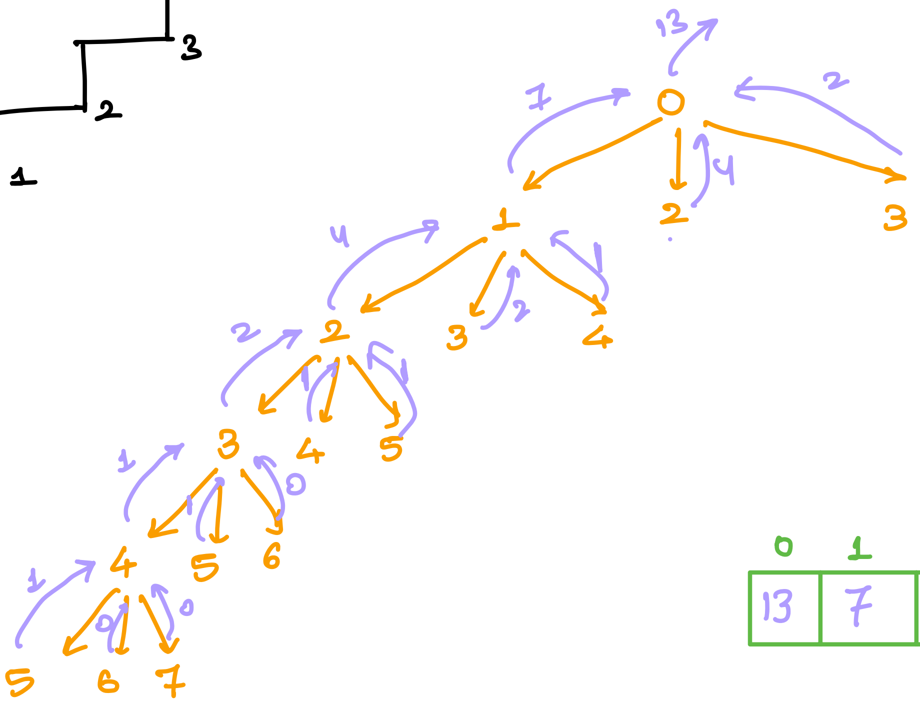
Climbing Stairs

TD DP:



no. of ways Gf → 5th floor ?

1,2,3 jump



0	1	2	3	4
13	7	4	2	1

n=5   index: 7 (n+2)   size: n+3

BU DP(Iteration)

0	1	2	3	4	5	6	7
13	7	4	2	1	1	0	0

- Size ?
- TD BC → BU fill work start
- Cell meaning
- filling dr^n ? ⇄
- fill
- final answer.

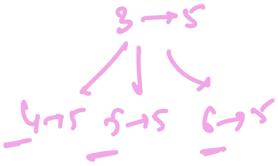
n=5

+ve bc: 5

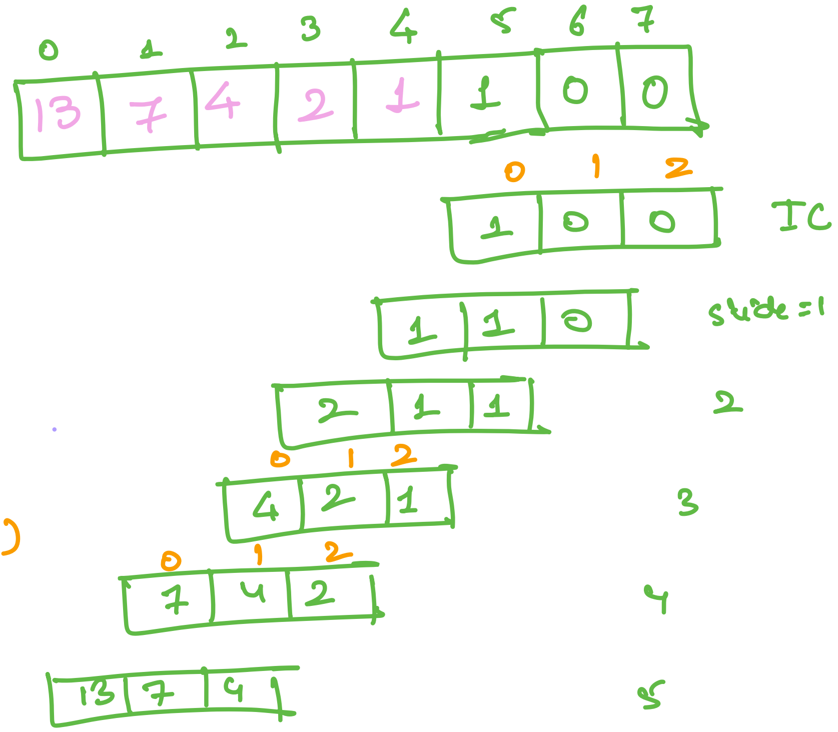
-ve bc: 6, 7

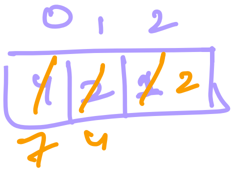
i index → ith stair → nth stair ways ?

4



n=5





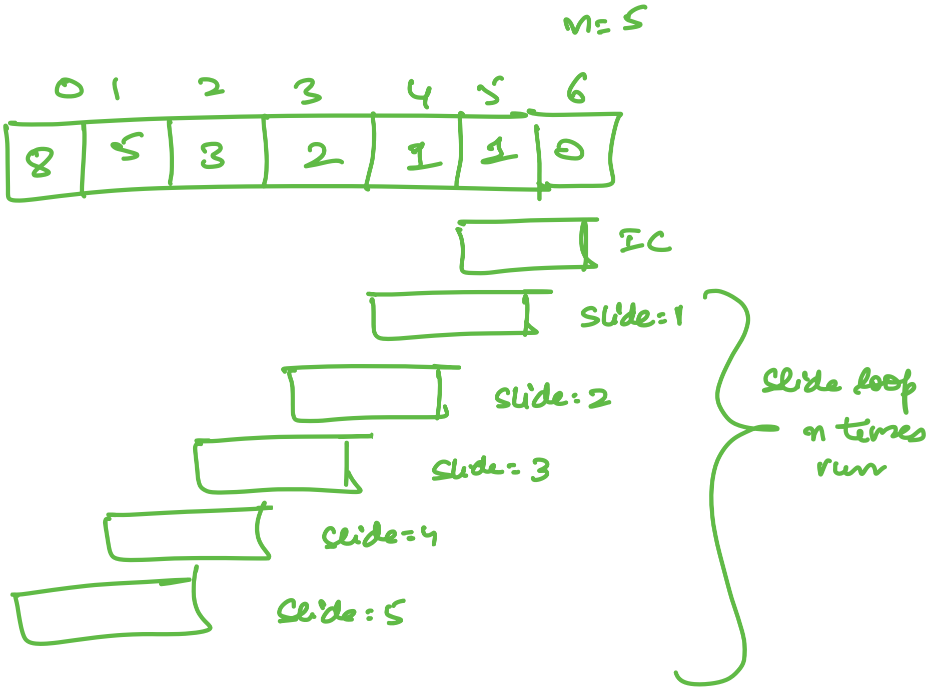
sum = s[0]+s[1]+s[2]
= 7

strg[2]: strg[1]

strg[1]=strg[0]

strg[0]=sum

Leetcode: Climbing stairs



Climbing Stairs → count based

Longest Common Subsequence (LCS)

Given 2 strings find the length of longest subsequence which is present in both the strings.

Subsequence →

Sequence that appears in the same relative order but not necessarily contiguous.

Eg: abcdefg

subsequence: abc, abg, bdg, aeg, accfg

dbg ×

Eg:

S1	S2	length of LCS	LCS
abcd	agcfd	3	acd
abc	adcb	2	ab, ac
abc	acd	2	ac

Brute force Approach:

length m abc s1 acd s2 length n

s1 subsequences: [–, a, b, c, ab, ac, bc, abc] $\rightarrow 2^m$

s2 subsequences: [–, a, c, d, ac, ad, cd, acd] $\rightarrow 2^n$

Common subsequences: [–, a, c, ac]

Time Complexity: $2^m + 2^n + 2^m \cdot 2^n = 2^m + 2^n + 2^{m+n} = \underbrace{O(2^{m+n})}_{\text{Exponential}}$

RECURSION:

Recursive call return: LCS(s1,s2) = s1, S2 LCS length

LCS (abcd, agcfd) = 3

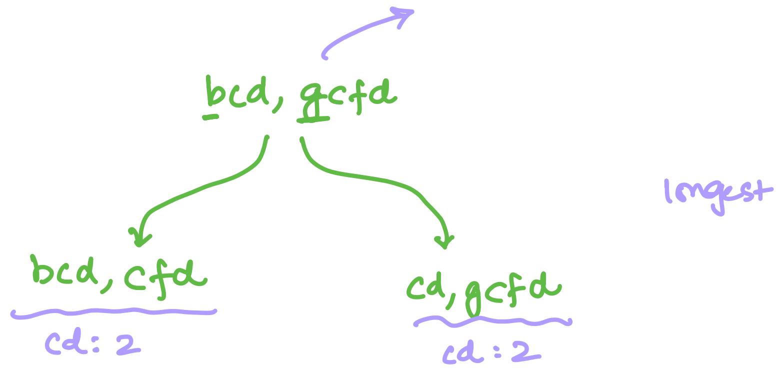
Case 1:

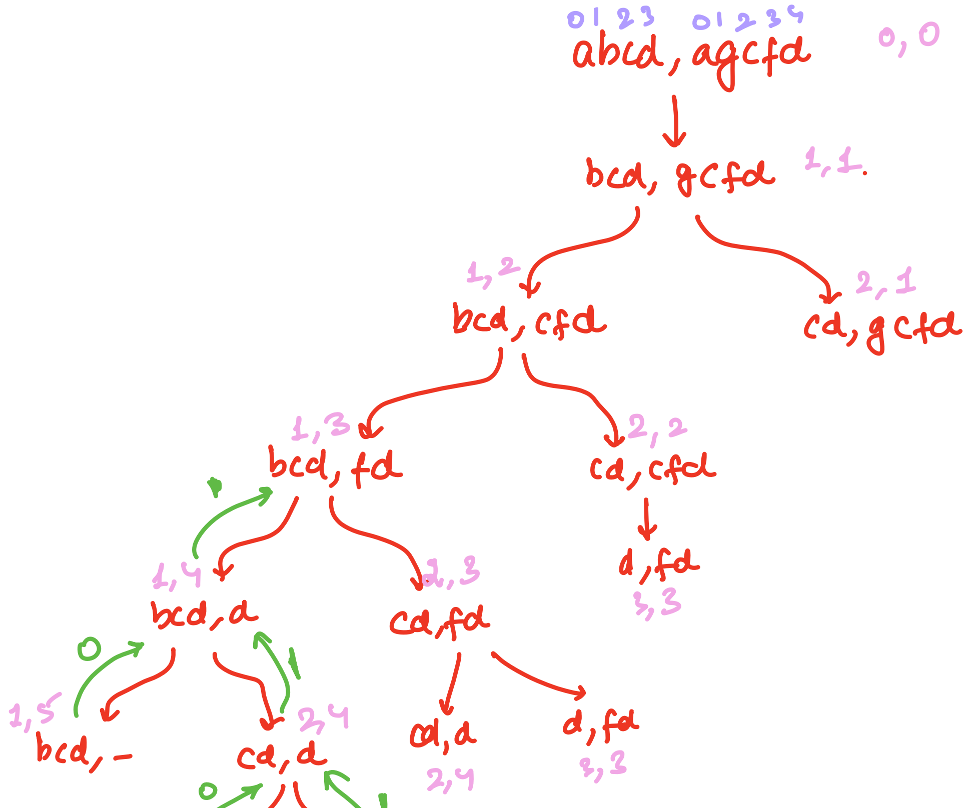
s1 abcd, s2 agcfd   ss a $\boxed{cd}$ 2+1=3

↓

bcd, gcfd $\boxed{cd}$ → 2

Case 2:

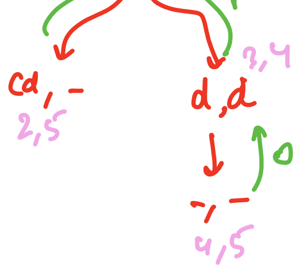


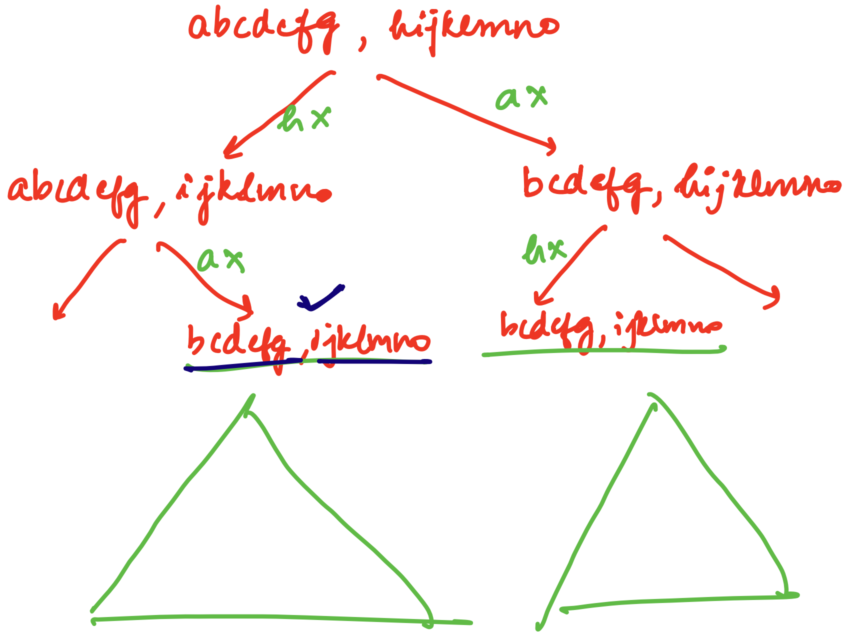


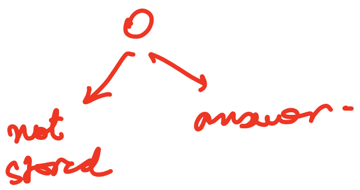
0 1 2 3
abcd

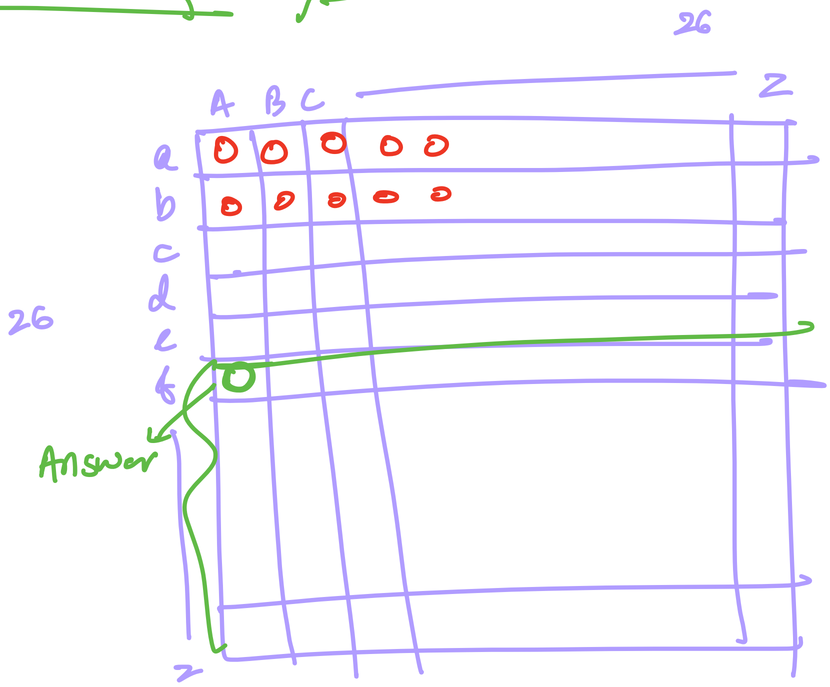
abcd

idx	
0	: abcd
1	: bcd
2	: cd
3 :	d
4 :	-









if 0 is answer

then keep default value in array as -1