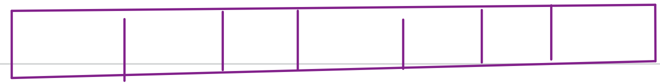
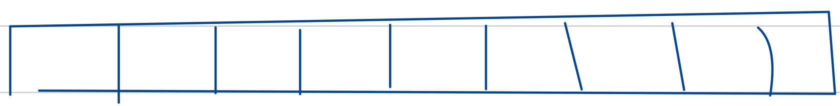
Q:- LIS



find a subsequence which is inc & longest also

longest inc subsequence

i → what is LIS ending at i



$$\boxed{\begin{matrix} a[i] > a[j] \\ i > j \end{matrix}}$$

$f(i) = 1 + \max(f(j))$ if $a[j] < a[i]$

$\forall j \in [0, i-1]$

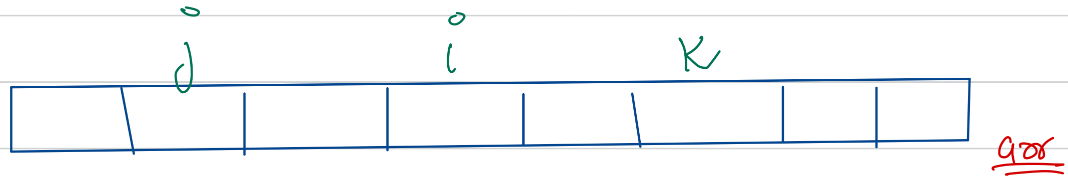
func^n denoting lis ending at i^{th} index

optimum → dp

final ans → $\max(f(i))$

$\forall i\ [0, n-1]$

Let's optimize lis $\longrightarrow$ $n \log n$ (Binary Search, Seg tree)



~~old approach $O(n^2)$~~

we say $dp(i) \rightarrow$ lis ending at i

$\downarrow$

$\forall j \in [0, i-1] \longrightarrow j < i$

$arr[j] < arr[i]$

$j < i < k$

$arr[k] > arr[j]$

$arr[k] > arr[i]$

let's consider $\longrightarrow$ $j < i \longrightarrow dp(j) < dp(i) \longrightarrow arr[j] > arr[i]$

then for any $k > i \longrightarrow dp(j)$ is irrelevant.

$arr[k] > arr[j] \longrightarrow dp(k) = 1 + dp(j) \rightarrow x$

$arr[k] > arr[i] \longrightarrow dp(k) = 1 + dp(i) \rightarrow y$

we know $dp[j] < dp[i]$

$\boxed{arr[j] > arr[i]}$

all those elements which can make lis with arr[j] can also make lis with arr[i]

always $y > x$

and preference will be arr[i] why? $dp(i) > dp(j)$

→ Can we conclude

for any index $j \in [0, i-1]$

$j < i$

$arr[j] > arr[i]$

$dp(j) < dp(i)$

we can remove the element j from further consideration

$a[j] < a[i]$
$dp(j) < dp(i)$

$a[j] > a[i]$
$dp(j) < dp(i)$

So after removals, what will be left ??

will there be any j
$arr[j] > arr[i]$ yes / No

	0	1	2	3	4
	7	8	4	3	9
	1	2	1	1	

No we delete

$a[1] > a[3]$
$dp(1) < dp(3)$

$\rightarrow$

$a[1] > a[3]$
$dp(1) > dp(3)$

$\rightarrow$ yes we need it

How about if we keep the data somehow sorted

$a[0] > a[4]$

$7 > 5$

$dp[0] < dp[4]$

$n\log n + n\log n \rightarrow O(n\log n)$

sorted based on ele value

linear search

0	1	2	3	4
7	8	4	3	5
↓	↓	↓	↓	
1	2	1	1	

rearrange

(idx, ele, dp)

4, 9, x

4, 5, x

it

(3, 3, 1) (2, 4, 1) (0, 7, 1) (1, 8, 2)

(3, 3, 1) (2, 4, 1) (4, 5, 2) (0, 7, 1) (1, 8, 2)

map <pair>

ordered map

Binary search

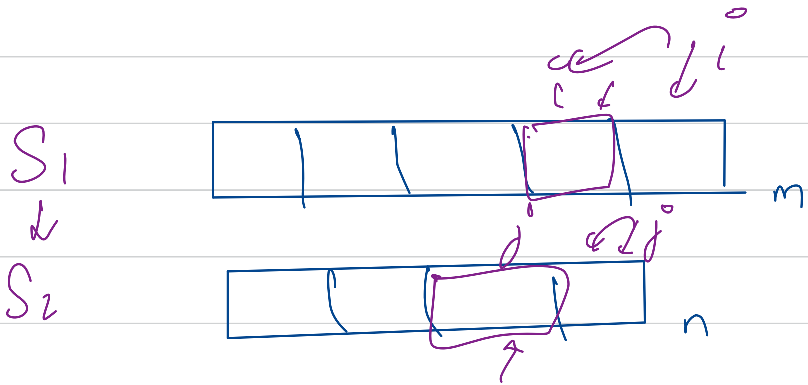
Q:

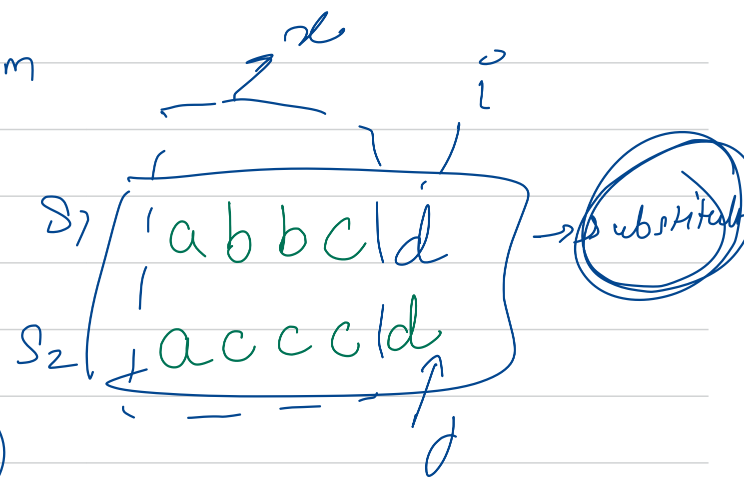
horse

ros

addition
deletion
substitution

levenshtein distance





operation can be applied only → S1

ans → $1 + x$

abbcd i

accd j

Substitute

abbcd 1+ abbc $i-1$

accd acc $j-1$

Removal

abbc $i-1$

accd j

y

1+y

i

abbcx (i, j)

accd

j

addition

abbcxd

accd

i

abbcx $(i, j-1) + 1$

acc

j

$f(i, j)$ = $\begin{cases} f(i-1, j-1) & \text{if } (s_1[i] == s_2[j]) \\ \min\big(f(i, j-1), f(i-1, j), f(i-1, j-1)\big) + 1 & \text{else} \end{cases}$

min operation to convert

$S_1[0, i] \rightarrow S_2[0, j]$

insert, delete, replace

abbc ← $i-1$
acc ← $j-1$

x

abbcd
accd

replace

abbcx ↓ i

accd ↓ j

$1+x$

addition

$1+y$

abbcxd
accd

abbcx
acc
$j-1$

$S_1[0,i]$
$S_2[0,j-1]$ → y

delete

abbc → $i-1$
accd → j

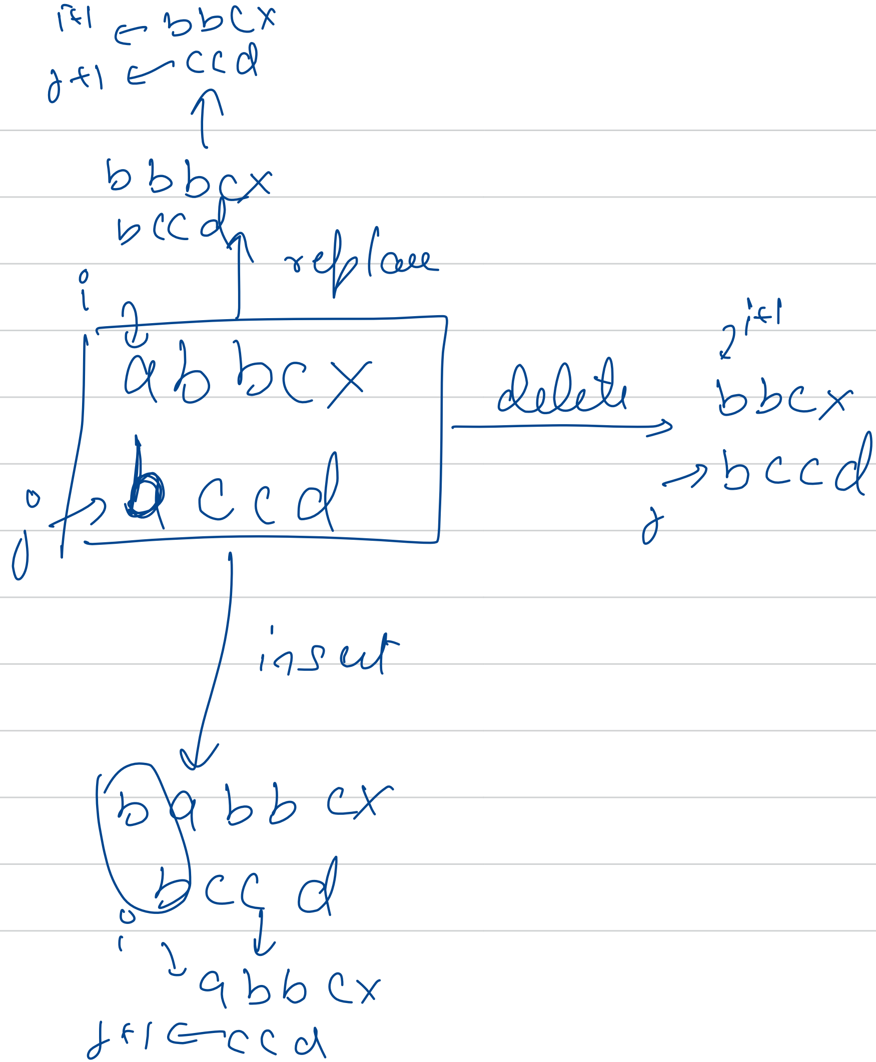
$1+z$

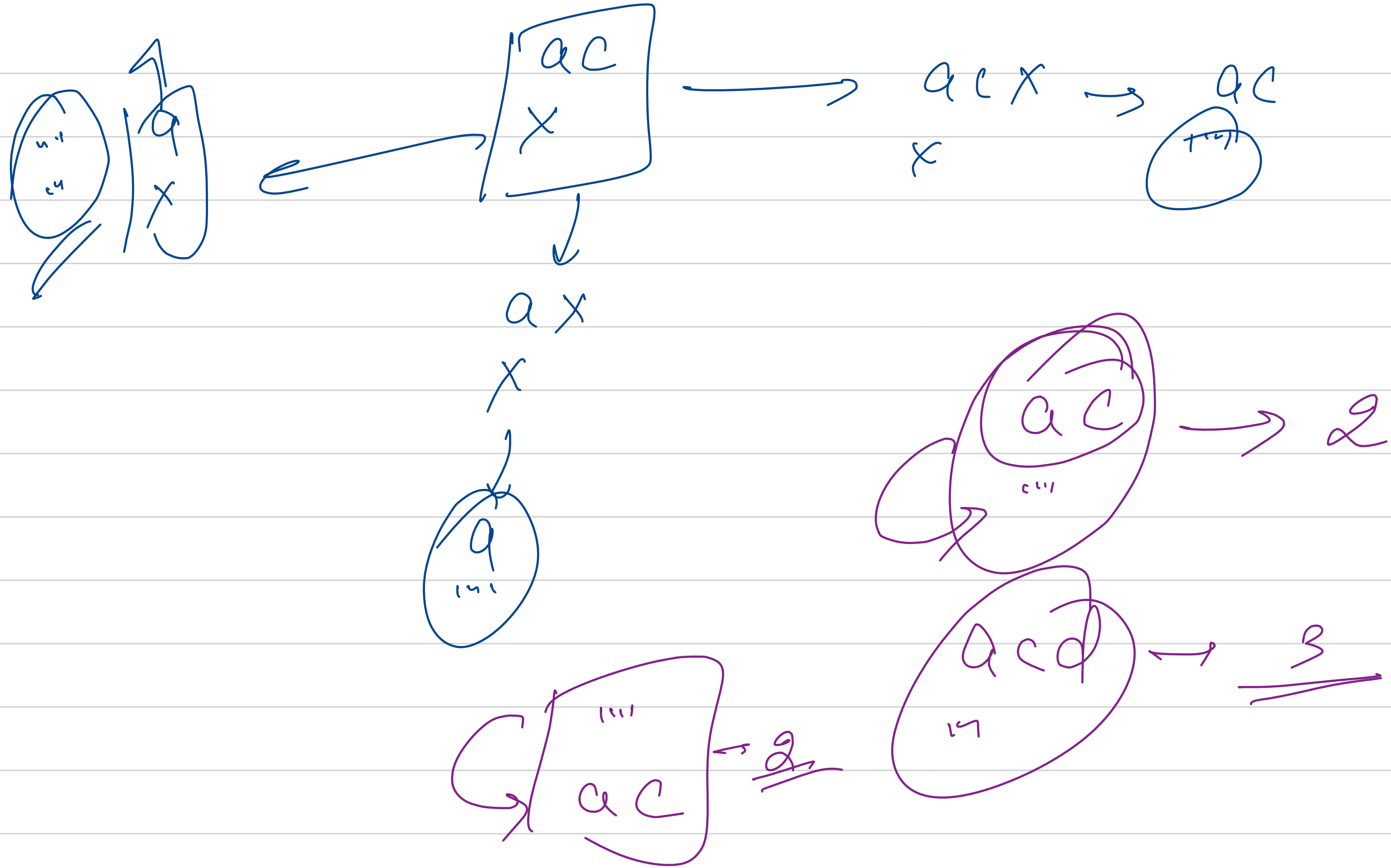
$z = \left(S_1[0,i-1], S_2[0,j] \right)$

final

$\min(1+x, 1+y, 1+y)$

$1+\min(x,y,z)$


i+1 ← bbcx
j+1 ← ccd
bbbcx
bccd
replace
i
j
abbcx
bccd
j
delete
i+1
j
bbcx
bccd
j
insert
babbcx
bccd
i
j
abbcx
j+1 ← ccd


a
x
ac
x
acx
x
ac
ax
x
a
ac
2
acd
3
ac
2

abbcx

accd

	""	a	c	c	d
""					
a					
b					
b					
c					
x					

d_i

day

$i \to a_i \to 0$

$i \to b_i \to 1$

$i \to c_i \to 2$

$a = \{0, 1, 2\}$

$i-1 \quad i \to 0$

1 2

max

2D → 3 rows, n cols

$$f(x, i) = \begin{cases} a_i + \max\big(f(1, i-1),\ f(2, i-1)\big) & \text{for } x = 0 \\ b_i + \max\big(f(0, i-1),\ f(2, i-1)\big) & \text{for } x = 1 \\ c_i + \max\big(f(0, i-1),\ f(1, i-1)\big) & \text{for } x = 2 \end{cases}$$

max happiness gained in the first i days by doing the given activity → a on i^{th} day.

$\underline{\text{finalans}} \rightarrow \max\left(f(0, n-1), f(1, n-1), f(2, n-1)\right)$

Q → Given a string, → count the no. of subsequences of the type following this regex

$$a^+ b^+ c^+$$

$(+) \rightarrow$ one or more

→ a b c a b c

→ ?

abc

abc   abcc

abbc   abc

aabc   abc

abbc

→ 3

aaa

↓

a^+

abb

a^+b^+

abc

abc

abbc

$$\boxed{a^+ b^+ c^+}$$

dis

subsequences of the regex ending at i^th index

	a	b	c	a'	b'	c'
$\boxed{a^+}$ →	1	1	$\boxed{1}$	3	→ 3	3
a^+b^+ →	0	1	1	→ 1	5	5
$a^+b^+c^+$ →	0	0	1	→ 1	→ 1	5 + 2 → 7

3 +

5 +

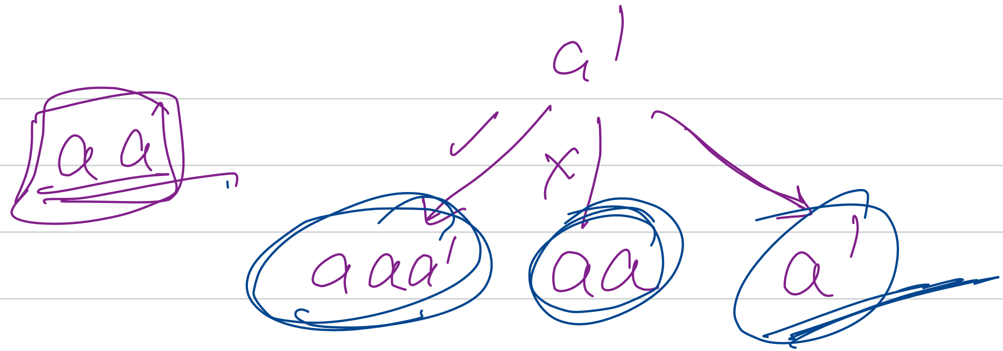
abc

1

a ← aa'

a ← ~~a~~ a'

a' ←



$1 + \underbrace{2 * acount}$

$\underline{acount} + 2 * \underline{bcount}$

$bcount + 2 * ccount$

$$f(r, i) = \begin{cases} 1 + 2^{*} f(a^+, i-1) & r = a^+ \\ f(a^+, i-1) + 2^{*} f(a^+b^+, i-1) & r = a^+b^+ \\ f(a^+b^+, i-1) + 2^{*} f(a^+b^+c^+, i-1) & r = a^+b^+c^+ \end{cases}$$

no of subsequence of regex r ending i

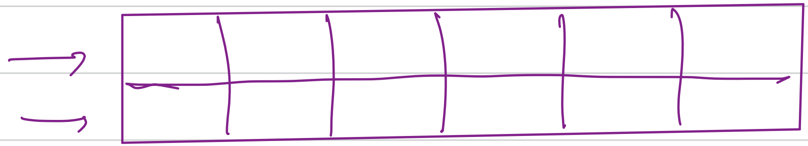
$a^+ \to 0$
$a^+b^+ \to 1$
$a^+b^+c^+ \to 2$

$r \to a^+$
$\to a^+b^+$
$a^+b^+c^+$

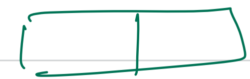
$$dp(r,i) = \begin{cases} 1 + 2 \cdot dp(0, i-1) & r == 0 \\ dp(0, i-1) + 2 \cdot dp(1, i-1) & r == 1 \\ dp(1, i-1) + 2 \cdot dp(2, i-1) & r == 2 \end{cases}$$

Ans $\rightarrow$ return $\underline{\underline{dp(2, n-1)}}$

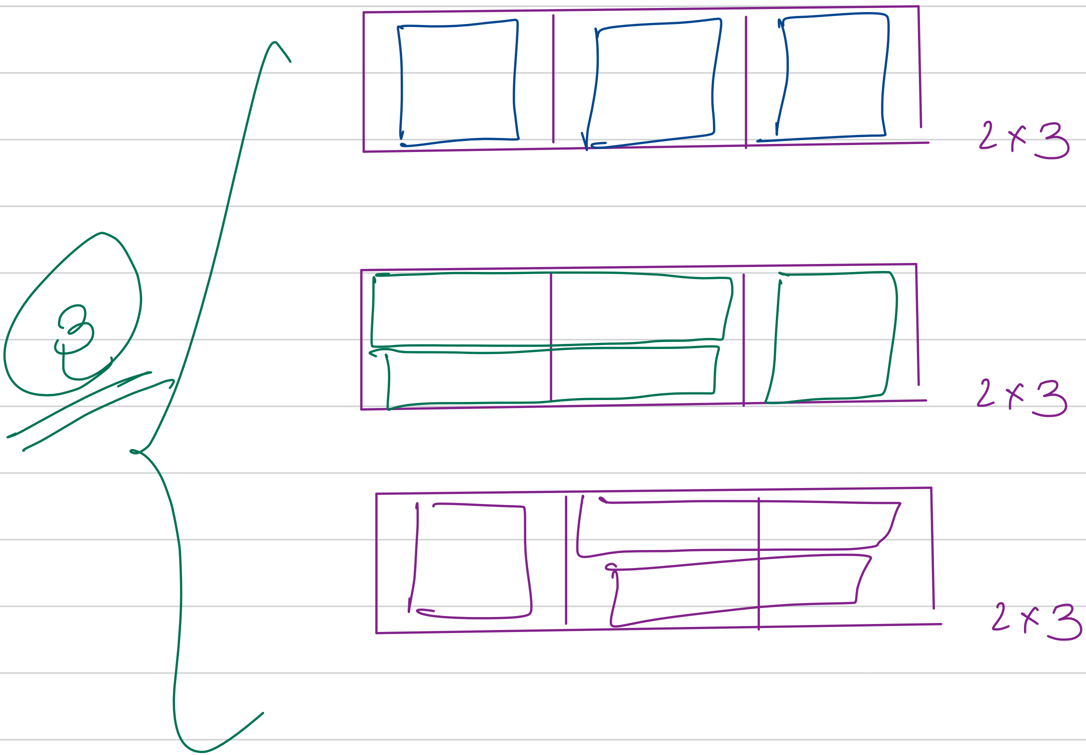
Q. grid

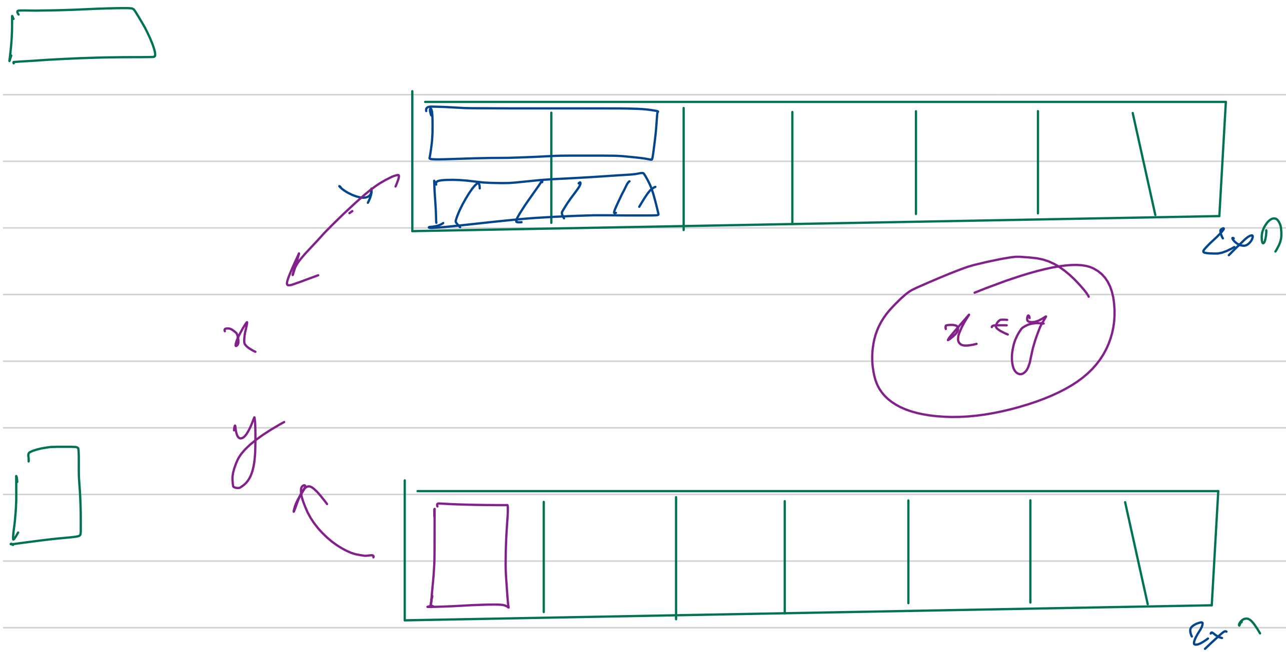


$2 \times n$

tiles → 2×1  → rotate →  1×2

How many ways you can tile the $2 \times n$ grid using the tiles


3
2 × 3
2 × 3
2 × 3


2xn
x
x ∈ y
y
2xn

$$\overbrace{f(n) = f(n-1) + f(n-2)}$$

$f(n)$ → func^n to return the no. of ways to tile $2 \times n$ grid

$f(n-1)$ → first pos, we choose vertical tile

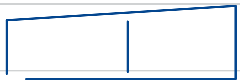
$f(n-2)$ → first 2 pos, we chose horizontal tile

$f(1) \longrightarrow$ 1 way (vertical)

2×1

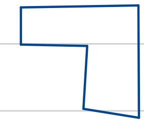
$f(2) \longrightarrow$ 2 ways (2 vertical) (2 horizontal)

2×2



1x2

2x1



L shaped

covers 3 squares

rotation allowed

2 x n grid

? ?