Dynamic programming

→ find sub problem
→ Create DP and define meaning
→ travel and fill (direction)

Q1) Fibonaci Better.

0	1	1	2	...	n.	

$f(n) = \underbrace{f(n-1) + f(n-2)}_{\text{recurring sub problem}}$ {function}

$f(n)$

$f(n-1) \quad + \quad f(n-2)$

$f(n-2) + f(n-3) \qquad f(n-3) + f(n-4)$

recurring.

So we stop this recalculation with array of prev values.

Q2) Climb down stairs (step 1, 2, 3 at a time)

$f(n) = f(n-1) + f(n-2) + f(n-3)$ {→ function}

$f(n-2) + f(n-3) + f(n-4)$ recurring subproblem

we pass an array.

Q3) Gold mine problem

mine can start from any row and 1st column and can move → ↗ ↘ ways. Collect max Gold.

~~f(n)~~

$f(n,m) = \max[f(n,m+1), f(n+1,m+1), f(n-1,m+1)]$
$+ \text{value}(n,m)$

start from last column of every row and find max and return max of 1st column.

Date
Page

Q4) minimum jumps → Value array.

given jump array [1, 4, 3, 2, 3, 2]

indicate maximum jump from here

find minimum jump to reach last index.

$f(n) :- Min(f(n-j)+1, f(n))$ if ~~Value (f(n-j)) >=~~ n-j)

if (Value (n-j) >= (n-j))

for → i (0, n)

for j (0, i)

min

Q5) Longest Increasing subsequence.

1	2	4	3	8	5	10	12	7	9

longest IS → 1, 2, 3, 5, 7, 9

1, 2, 3, 8, 10, 12 and many more

1	2	4	3	8	5	10	12	7	9
1	12	124	13	18	15	1 10	1235 10 12	12 4 5 7	1 9
	2	24	123	128	125	1210	1238 10 12	12357	4
		4	3	1248	1245	12410	1248 10 12		
				1238	1235	12310	other		
						124810	not		
						123810	LIS		
						123510			

for →

for →

find the ~~not~~ LIS to prev and add

ex for 10 → ~~Longe~~ LIS till prev

is for 8 and 5 choose

any and add 10 behind.

Q6) Rod Cutting problem :- given rope of len n and to cut in peices of 2, 3, 5 find max cut

fun :- if n = 2 || 3 || 5
    f(n) = Max(f(n), 1)
    ~~else f(n) = f(~~
    ~~if n >~~
  else
    if (f(n-2) > 0) {
      f(n) = Max(f(n-2) + 1, f(n))
    if (f(n-3) > 0){
      f(n) = Max(f(n), f(n-3) + 1)
    if (f(n-5) > 0){
      f(n) = Max(f(n), f(n-5) + 1)

Q7) Rod Cutting 2

given array of len n indicate price of rod of length i and rode of len n. find max price to which rod can be sell in parts or whole.

prices :- [1, 4, 5, 6, 12, 13, 15]
n = 7.

ans will be → 12 + 4 = 16

DP → | 1 | | i | | n | → travel.

Meaning :- maximum profit of rod of len (i)

Date
Page

Q8) ☆ **Target Sum ~~Targ~~ Subset / test run (pepcoding)**

given an array of values find if there exist a subset with target sum. (yes / no)

arr ⇒ [1, 4, 9, 12, 8, 7, ~~5~~, 6]
target ⇒ 19 → (1, 12, 6)
(12, 7)
(8, 7, 4)

	0	1	2	3	4	5	6	7	8	9	10 ....
0	1	0	0	0	0	0	0	0	0	0	0
1	1										
4	1										
9	1										
12	1										
8	1										
⋮											

meaning → if possible to achieve ~~target~~ 4 with array element 0, 1, 4. with 4 playing or not playing.

if possible to score 2 run with players who can score 0, 1, 4, 9, 12, 8

Method :- ~~f(n)~~ = ~~f(n - value(n))~~ || ~~f(n-~~

$f(n, m) = f(n-1, m - value(n)) + value(n) == tar$
$\quad || \; f(n-1, m) == tar)$

if the player don't play.

if the player plays

Q9) Coin Changer Permutation :-

given ~~a~~ infinite supply of let say 2, 3, 5 ₹
find Combination to pay 15 ₹.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

0
2
3
22
32
23
5
222
33

2+f(5)
3+f(4)
5+f(2)

2+f(6)
3+f(5)
5+(f(3)

Formula :- ~~for coins : if i-coins[j]~~

f(n) = for coins { if (f(n - coins(j))) > 0 ; f(n) += f(n - coins(j)) }

remember → ~~coins → upper for loop~~
n → upper for loop
coins → lower for loop.

Q10) Coin change ~~permutation~~ combination

reverse the loop for combination

	1	2	3	4	5	6	7	8	9
for 2	0	2	0	22	0	222	0	2222	0
for 3	0	2	3	22	23	222 33	223		2223
for 5	0	2	3	22	23 5	222 33	223 ~~25~~	2222 35	2223 225

3 will come after 2 and 5 will come after 3 so only Combination.

Q11) Max sum with no adjacent element.

	1	4	8	5	6	9	12	14
inclu	1	4	1+8	4+5	15	18	27	
ex	0	1	4	9	9	15	18	

$f(n) = \text{Max}(f(n-1), f(n-2) + \text{Value}(n))$

meaning:- Max if include 14
Max if exclude 15

~~new include =>~~
new_include = exclude + Value(new)
new_exclude = Max(exclude, include)
include = new_include
exclude = new_exclude.

Q12) Longest Common Subsequence.

a b a c b c a d a
b c a b d b a d c

	a	b	a	c	b	c	a	d	a
b	0	1	1	1	1	1	1	1	1
c	0	1	1	2	2	2	2	2	2
a	1	1	2	2	2	2	3	3	3
b	1	2	2	2	3	3	3	3	3
d	1	2	2	2	3	3	3	4	4
b	1	2	2	2	3	3	3	4	4
a	1	2	3	3	3	3	4	4	4
d	1	2	3	3	3	3	4	5	5
c	1	2	3	4	4	4	4	5	5

Meaning here :- if charAt(i) == charAt(j)
then Value = Max((i-1)(j-1)+1, (i-1)j, (i)(j-1))
or → Max((i-1)j, i(j-1)).

13.) Bounded knapsack problem :-

Same as target pair problem

14) unBounded knapsack problem :-

Bag size :- 15

items :- 2, 3, 5

Values :- 4, 7, 9

	0	1	2	3	4	5	6	7	8	9	10...
0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	4	0	8	0	12	0	16	0	20
3	0	0	4	7	8	11	14	0	18	21	20
5	0	0	4	7	8	11	14	13	18	21	20

algorithm :- if n-item == 0 :- Max (val, f(n))

(n-item) > 0 :- if f(n-item) > 0

:- Max((f(n-item) + value(item)), f(n))

15.) Buy and sell stocks with fee

fee :- 3 2.

Array	Buy state profit	Sell state profit
3	-3	0
5	-3 || -5 = -3	0 || 5-3-3 = 0
10	-3 || -10 = -3	0 || 10-3-3 = 4
6	-3 || -6+4 = -2	4 || 6-3-3 = 4
9	-2 || -9+4 = -2	4 || 9-2-3 = 4
13	-2 || -13 + 4 = -2	4 || 13-2-3 = 8
2		
8		
10		
12		

new buy state profit = Max (old buy state profit, - arr[i] + old sell state profit)

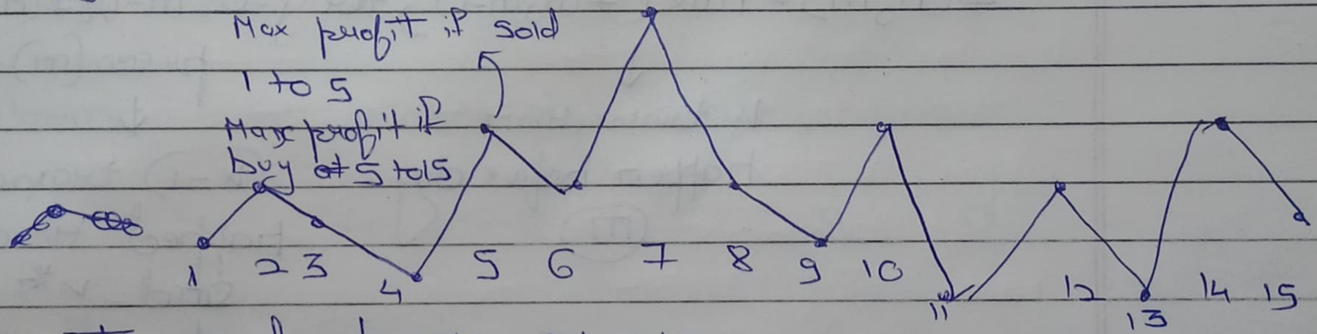
new sell state profit = Max (old sell state profit, arr[i] + old buy state prob - fee)

16) Buy sell stocks with Cooldown.

Array	BSP	SSP	CDSP
	-3	0	0
3			
5	-5 ‖ -3 = -3	5-3 ‖ 0 = 2	0
10	-3 ‖ -10+0 = -3	2 ‖ 10-3 = 7	2
4	-3 ‖ -4+0 = -3	7 ‖ 4-3 = 7	7
9	-3 ‖ -9+7 = -2	7 ‖ 9-3 = 7	7
13	-2 ‖ -13+7 = -2	7 ‖ 13-2 = 11	7

11 ↓ ans

(17) Buy and sell stocks 2 transaction allowed.



at each point Calculate
1. Profit maximum if sold at that point or earlier
2. Maximum profit if buy at that point or further.

return maximum of sum of both.

(18) Buy and sell stocks atmost k transaction allowed.

Subproble → to include / sell at k element
or to not include at k element.

	1	2	3	4	5	6	7	8	9	10
prices →	3	5	10	6	9	13	2	8	10	12
0	0	0	0	0	0	0	0	0	0	0
1										
2										
3										
4										

$k = 4,$

meaning → max profit if max 2 transaction allowed with prices till day 5

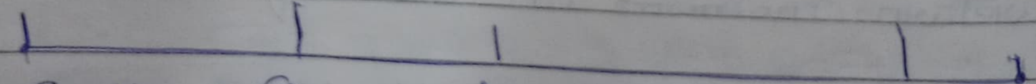
∴ can be solved

$f(n, m) = \text{Max}(f(n, m-1),$ for $(j = 0 : m-1)\{f(n, j) +$ prices(m) − prices(j)

$f(n, m-1)$ → k transaction happen before day (m)

prices(m) − prices(j) → (k−1) transaction happed till day j and k^{th} transaction happening b/w day m and j.

Q19.) Highway bill board problem



2 Condition:- ① array of profit from billboard i
② no bill board can be closer than distance k

2 method:- ① L.I.S method
② maximum profit till distance a

1 2 3 4 5 6

2 6 4 5 7 9

$x = 2$

who can join after 5 to make maximum profit

0 1 2 3 (4) 5

2||0 6||2 4+2||6 5+6||5+2 7+6||7+6||7+2

2 6 6 11 13 6

11+6|| 6||6

17

$O(N^2)$

LIS

Maximum profit till 4

Method 2

0 1 2 3 4 5 6

0 2 6

O(N)

Q2) Maximum sum increasing subsequence

2 6 4 5 7 9 8

2 6 4+2 = 6 5+2||5+6 ⇒11 7+8||7+2||7+6||7+11 ⇒18

6+2 = 8

9 8

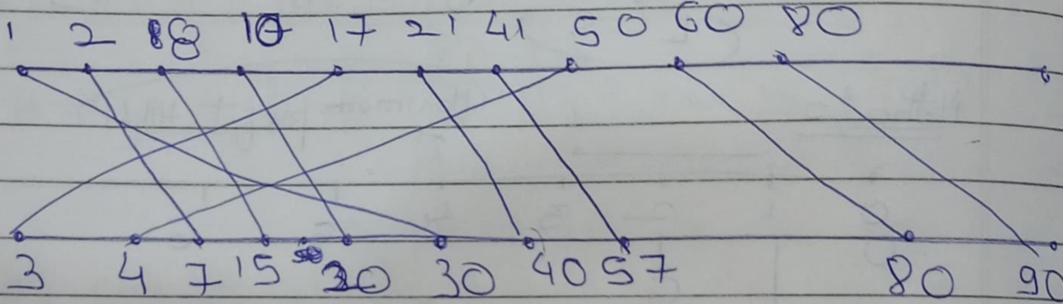
18+9 18+8

same concept as LIS

Date
Page

Q(21) Maximum non overlapping Bridges
given north and south points, find maximum non overlapping bridges

(10,20) (2,7), (8,15), (17,3), (21,40), (50,4), (41,57), (60,80), (80,90), (1,30)



Now, to solve the problem → ~~South~~ sort the array based on north / south and apply LIS on the other and return maxi example
~~for~~ sort North and apply LIS on south
i.e. for LIS of 90:
90 ⟶ ~~3, 4, 7, 15~~

final sorted array

1	2	8	10	17	21	41	50	60	80
30	7	15	20	3	40	57	4	80	90

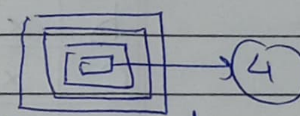
north is already sorted

apply Lis on South
ex for 90:- (7, 15, 20, 40, 57, 80, 90)

Q22) Russian doll envolope

given n dolls with ~~cod~~ width and height find maximum dolls that can fit into another.

→ height and width both should be loose than other doll.



Approach:- Sort basis on width and apply LIS on height.

Q23) Count Palindromic substring.

	a	b	c	c	b	c
a	✓	×	×	×	×	×
b	×	✓	×	×	✓	×
c	×	×	✓	✓	×	×
c	×	×	×	✓	×	✓
b	×	×	×	×	✓	×
c	×	×	×	×	×	✓

→ b = b
∴ check

c == c
but diagonal is ×

abccbc

not palindrome

Palindrome
first == last &&
middle palindrome

rule (1) travel gap way

(2) if array[i] == array[j]
check diagonally down left if ✓ then ✓ else ×.

(3) if array[i] ≠ array[j]
mark ×

ARFAT Date Page

Q24) Longest Palindromic Substring.

	a	b	c	c	b	c
a	1	0	0	0	0	0
b	x	1	0	0	4	0
c	x	x	1	2	0	0
c	x	x	x	1	0	3
b	x	x	x	x	1	0
c	x	x	x	x	x	1

Rule
for size 1 → fill 1
for size 2 Check if a[i] == a[j]
on score

E if
arr[i] == arr[j]
dP[i][j] = if dP[i+1][j-1]+2 if
dP[i+1][j-1] > 0 and

Q25) Print all minimum cost path DP

$$\begin{bmatrix} 2 & 6 & 4 & 1 & 3 \\ 9 & 1 & 1 & 0 & 5 \\ 0 & 7 & 5 & 2 & 0 \\ 4 & 3 & 0 & 4 & 7 \\ 2 & 0 & 1 & 4 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 20 & 18 & 12 & 11 & 16 \\ 21 & 12 & 11 & 10 & 13 \\ 12 & 16 & 11 & 10 & 8 \\ 12 & 9 & 6 & 9 & 8 \\ 8 & 6 & 6 & 5 & 1 \end{bmatrix}$$

to print all path of minimum cost we use BFS and increase the count the number of time we reached destination

Q26) Count distinct subsequence

	1x2	2x2=4	4x2=8	8x2=16 ⇒14	14x2⇒28 -1=27	27x2⇒54
•	a	b	c	b	a	c
•	•	• b	• c	• ~~b~~		
	a	a ab	a ac	a ~~ab~~		
			b bc	b bb		
			ab abc	ab abb		
				c cb		
				ac acb		
				bc bcb		
				abc abcb		
				16-2 =14		

54-4 = 50

we have to reduce the number of subsequence of previous b -1 ⇒ 16-2=14.

if the element hasn't comes first directly double
Or reduce the count before it (previous)

Q27) Count Palindromic Subsequence

		a	b	c	b	a	c
a	x	1	2	.			
b	x	x	1	2			
c	x	x	x	1	2		
b	x	x	x	x	1	2	
a	x	x	x	x	x	1	2
c	x	x	x	x	x	x	1

if single char → then 1

if 2 char than — if same → 3 → cc → c, c, cc

not same → 2 → ab → a, b

~~String a --- b~~

~~if a & b are equal~~ ~~not equal~~

~~if palindrome~~ ~~if not palindrome~~

~~Same as~~ C_1

String a --- b

if a == b

∴ palindromic subsequence of a---b =

$a---b \Rightarrow 2C_1$

else:-

str ⇒ $C_1 + m + C_2$

$Sq(str) =$ ___ $+ Sq(m) +$ ___ — ① C^1

$C_1 + Sq(m) +$ ___ — ② C^2

___ $+ Sq(m) + C_2$ — ③ C^3

$C_1 + Sq(m) + C_2$ — ④ C^4

$C_1 = C_2$

if ~~eq① × 2 = eq~~ ④

$C^4 = C^1 \times 2 + 1$

↓

C_1 blank C_2

total ~~C~~ $= C^1 + C^2 + C^3 + C^1 + 1$

$\Rightarrow 1 + (C^1 + C^2) + (C^3 + C^1)$

$\Rightarrow 1 + Sq(c/m) + Sq(m/c)$

$C_2 \neq C_2$

~~$C_4 = 0$~~ eq^n ④ $= 0$

∴ ~~$C = C_1 + C_2 + C_3$~~

$C = C^1 + C^2 + C^3$

$C = C^1 + C^2 + C^3 + C^1 - C^1$

$C = Sq(c/m) + Sq(m/c) - C^1$

$C = Sq(c/m) + Sq(m/c) - Sq(m)$

ARFAT
Date
Page

rules :- if char length = 1 → Ans = 1
if char length ⇒ 2 → if same → 3
if not same → 2
else :- if same → 1 + sq(c/m) + sq(m/c)
else → sq(c/m) + sq(m/c) − sq(m)

(Q28) Print subsets with target sum.
make the matrix and print the path with
BFS

(Q29) Longest Palindromic Subsequence
★
l(abcd) → s(abcd) depends on
↓
set

s(abcd) → s(bc)
→ a + s(bc)
→ s(bc) d
→ a + s(bc) + d

l(str) → l(c1 m c2) → s(c1 m c2) → s(m) — (1)
c1 s(m) — (2)
s(m) c2 — (3)
c1 s(m) c2 — (4)

if $c_1 = c_2$ then answer will be only in C^4 (4) = 2 + (1) l(str) = 2 + l(m)	if $c_1 \neq c_2$ then answer is in (1) (2) (3) not in (4) l(str) = max(l(c/m), l(m/c))

	a	b	k	c	c	b	c
a							
b						.	
k							
c				.			
c							
b							
c							

answer for (k, c)

ARFAT Date Page

Q 30) Longest Common subsequence

$l(s_1, s_2) = l(c^1 r^1, c^2 r^2) \rightarrow$ ① $\_s(r_1) \times \_s(r_2)$

② $\_s(r_1) \times c_2 s(r^2)$

③ $c^1 s(r^1) \times \_s(r^2)$

④ $c^1 s(r^1) \times c_2 s(r^2)$

if $c_1 = c_2$

④

④ = 1 + ①

$c_1 \neq c_2$

① ② ③

$l(s_1, s_2) \Rightarrow \text{Max}(l(s_1, r_1), l(r_1, s_2))$

DP filling

	a	b	c	d	-
a	3	2	1	1	0
e	2	2	1	1	0
b	2	2	1	1	0
d	1	1	1	1	0
-	0	0	0	0	0

1+0 = 1

cd & d

d & d — cd & -

① vs ⓪

Q31) Wild card pattern matching

baaabab
ba*a?

	b	a	a	a	b	a	b	_
b	✓	✗	✗	✗	✗	✗	✗	✗
a	✗	✓	✓	✓	✗	✗	✗	✗
*	✓	✓	✓	✓ *	✓	✓	✗	✗
a	✗	✗	✗	✗	✗	✓	✗	✗
?	✗	✗	✗	✗	✗	✗	✓	✗
_	✗	✗	✗	✗	✗	✗	✗	✓

* → we look down if * behave as blank

→ blank match blank

Case → ? → we look diagonally down right ~~column~~

~~* → if star stand for blank look down else whole~~

Char :- $ch_1 = ch_2$ go diagonal down right
else no

* :- if * act as blank the match a? with abab

if * act as 1 char match a? with bab
if any of below col is (+)ve fill (+)ve
we can modify it

* → (look right)
↓
look down

*a? | abab
blank / \ all rest

*a? | abab   *a? | bab (b get removed than to * survived)

if * → (look down || look right)

Q32) Regular expression matching

ch* → can become ch, blank, ch ch, ...

	_	m	i	s	s	i	s	s	i	p	p	i
_	✓	x	x	x	x	x	x	x	x	x	x	x
m	x	✓	x	x	x	x	x	x	x	x	x	x
i	x	x	✓	x	x	x	x	x	x	x	x	x
s	x	x	x	✓	x	x	x	x	x	x	x	x
*	x	x	✓	✓	✓	x	x	x	x	x	x	x
i	x											
.	x											
*	x											
p	x											
*	x											
i	x											

if s* become blank we loop 2 above

if $ch_1 = ch_2$ → ↖ diagonal up left

$ch \neq ch$ x

$ch_1 = .$ → ↖ diagonal up left

$ch_1 = *$ → 2 above for 1st row

→ 2 above or ~~left~~ left if $ch_1 = (ch_2 - 1)$

else 2 above

mis*

mi

miss*

mis*/mis

mi/mis

x

mi~~s~~s*/mi~~s~~

mis*/mi

left problem

Q33) Count of distinct palindromic sequence

Str = $C_1 + m + C_2$

$C_1 \neq C_2$

↓

$dps(C_1 m) + dps(m C_2) - dps(m)$

$C_1 = C_2$

a m a	a $\frac{a}{m}$ a	a $\frac{a - a}{m}$ a
2dps(m) + 2	2dps(m)+1	2dps(m) − dps(m')

m' more than 1 a.

~~start~~ end →

	a	b	a	b	d	a
a	1 a	2 ab	4 aba	6 abab	7 ababd	11 ababda
b	x	1 b	2 ba	4 bab	5 babd	8 babda
a	x	x	1 a	2 ab	3 abd	6 abda
b	x	x	x	1 b	2 bd	3 bda
d	x	x	x	x	1 d	2 da
a	x	x	x	x	x	1 a

~~if we use a hash map to store occurance of prev and next occurance of the element~~

we make 2 array next and prev store the next and prev index of element.

(34) Catalan number

$$C_N = \sum_{i=0}^{N-1} C_i \times C_{(N-1)-i}$$

$C_0 \to 1$

$C_1 \to 1$

$C_2 = C_0 C_1 + C_1 C_0$

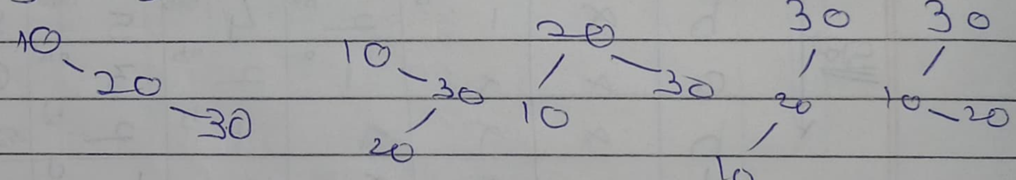
$C_3 = C_0 C_2 + C_1 C_1 + C_2 C_0$

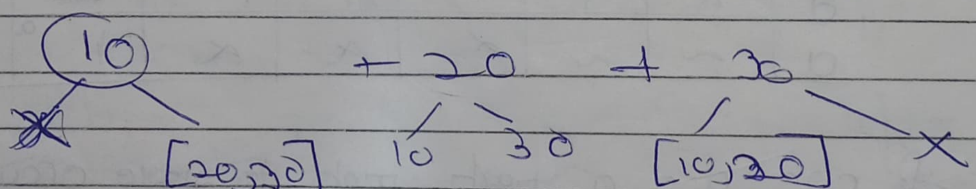
increasing decreasing

1	1	2	5	14	-	—	

Q.35) Count number of BST (catalan application)

10, 20, 30



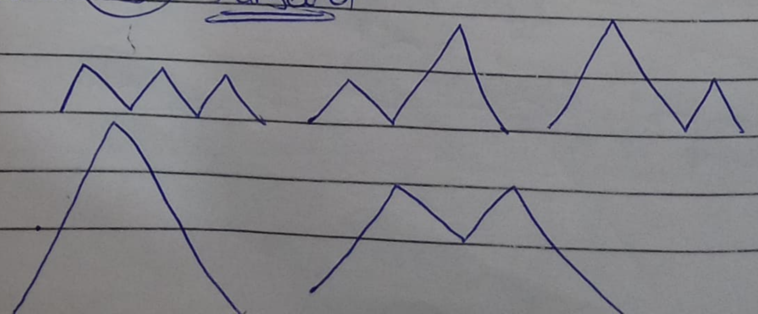


Catalan number

Q36) Count number of Vallys and mountains

Catalan number ✓

for 3 → 5 answer



Q37) Combination of balanced paranthesis (closed brackets <= open bracket)
→ Catalan number

Q38) Minimum Palindromic cut to make make it substring palindromic
⇒ left right strategy

1

ab ccbc — 2 — ab ccb|c (1) (0)
ab ccbc — 3 — abc|cbc
ab ccbc — 4 — ab cc|bc

0/ bccbc (2): 0, 1
ab| ccbc (3): 1, 1
abc|cbc: 2, 0
ab cc|bc: 2, 1

- Gap strategy.

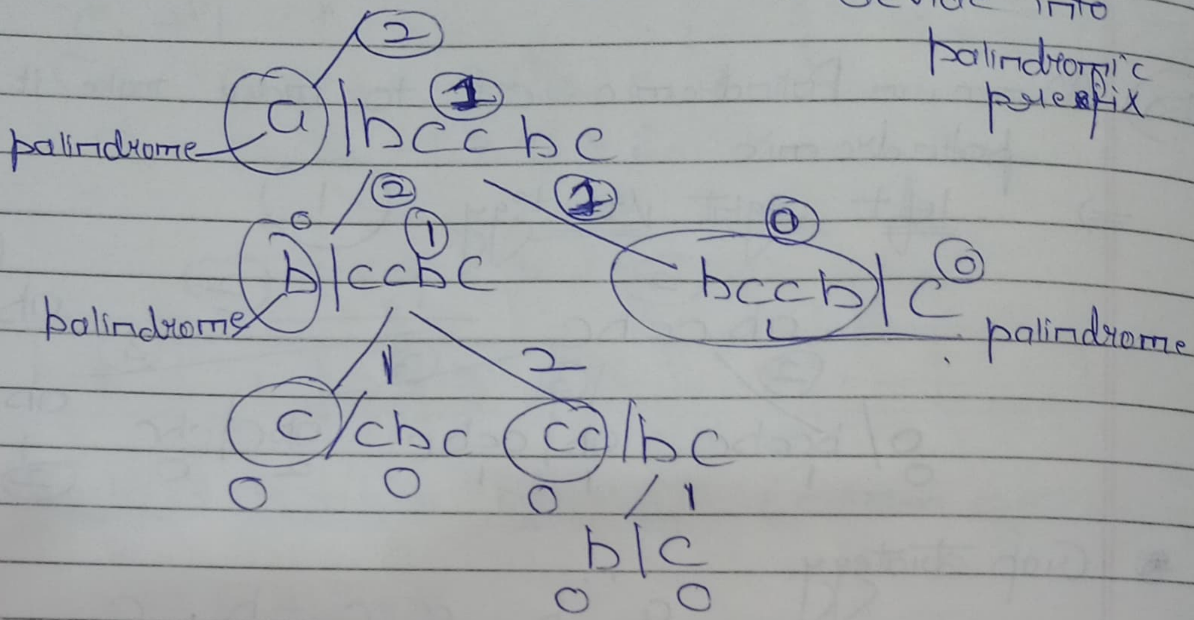
Boolean array

	a	b	c	c	b	c
a	✓	×	×	×	×	×
b	×	✓	×	×	✓	×
c	×	×	✓	✓	×	×
c	×	×	×	✓	×	✓
b	×	×	×	×	✓	×
c	×	×	×	×	×	✓
	~	~	×	×	×	×

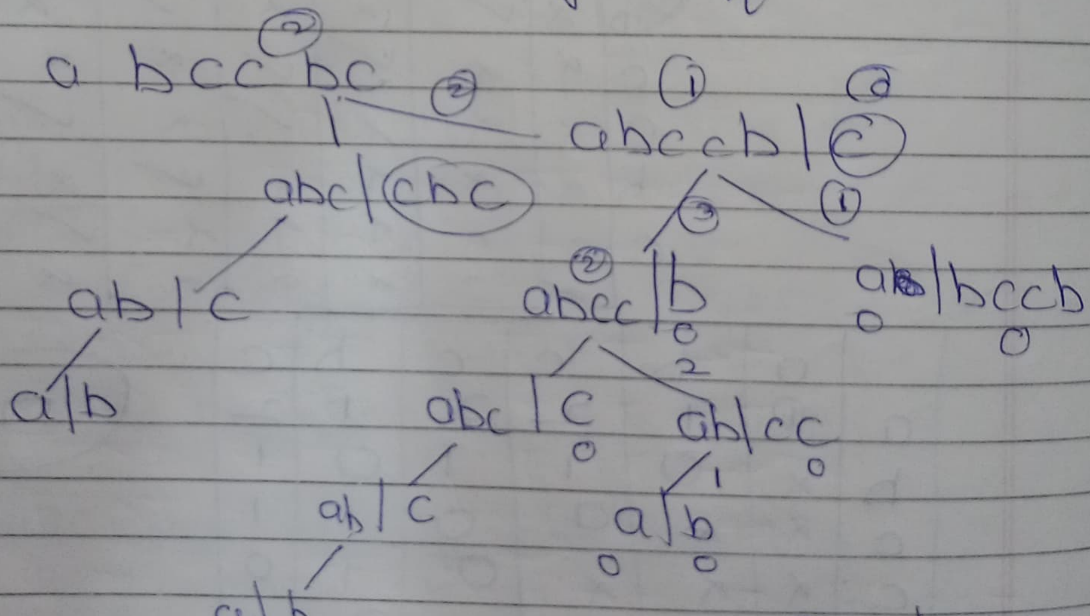
	a	b	c	c	b	c
a	0	1	2	2	1	2
b	×	0	1	1	0	1
c	×	×	0	0	1	1
c	×	×	×	0	1	0
b	×	×	×	×	0	1
c	×	×	×	×	×	0

n^3

⇒ n^2 method



we can do it also by suffix



a	b	c	c	b	c
0	1	2	2	1	2

Make boolean DP 2D array and for suffix and check if suffix is palindrome or not

(39) Matrix Chain multiplication :-

A × B = C

(2×3) [3×4] [2×4] ↑middle.

total cost of operation → 2×3×4

ABC →

A → 2×3
B → 3×4
C → 4×5

2×5

(AB) × C A×(BC)

2×4 4×5

A	B	C	D	E	
10	20	30	40	50	60

input

$l_1+r_1+m_1$ ABCDE $l_3+r_3+m_3$ $l_4+r_4+m_4$

↓ $l_2+r_2+m_2$

A|BCDE AB|CDE ABC|DE ABCD|E

l_1 r_1 l_2 r_2 l_3 r_3 l_4 r_4

Can be done in same way as palindromic cut 1st part.

(40) Boolean paranthazization

TFT → string (1) &| → string (2)

Combing string 1 & (2) to make Combine Value True by joining by brackets.

T & T → T	T | T ⇒ T	T ∧ F = T
T & F → F	T | F → T	F ∧ T → T
F & T → F	F | T → T	F ∧ F → F
F & F → F	F | F → F	T ∧ T → F

& | ^ & |

		T	F	T	F
&	T	1	0		
|	F	α	0	1	
^	T	α	α	1	1
	F	α	α	α	0

true

	T	F	T	F	
T	0	1			&
F	α	1	0		|
T	α	α	0	0	^
F	α	α	α	1	

false

[T F T T F T] → operand

[^ & | | &] → operator

T^(F&T|T|F&T)

+(T^F)&( )

if & → True × True } → true

True × F
F × T
F × F
} → False

Make 2 DP 1
one for False Count and one for true Count and remember

T^T = F	T|T → T
F^F = F	T|F → T
T^F ⇒ T	F|T → T
F^T = T	F|F → F

Same as matrix Chain multiplication

(41) **Optimal BST :-**

given :-

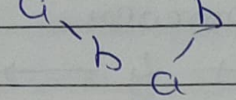
nodes = [1, 2, 7, 9]

search freq = [3, 8, 3, 5]

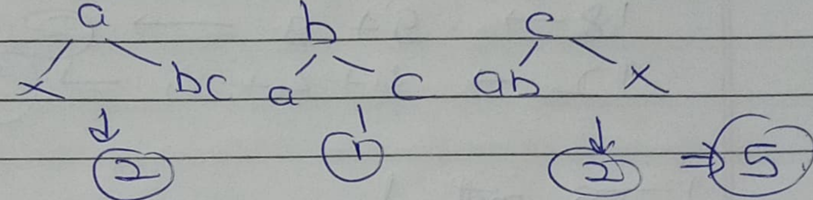
for n node, Catalan(n) number of BST.

a ⟶ 1

a, b ⟶



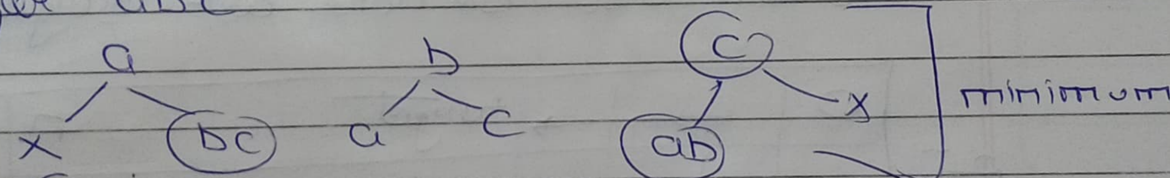
a, b, c ⟶



	a	b	c	d
a	a'	a'+2b' or 2a'+b'		
b	α	b'	b'+2c' or c'+2b'	
c	α	α	c'	c'+2d' or d'+2c'
d	α	α	α	d'

→ single node:
answer = frequency

for abc

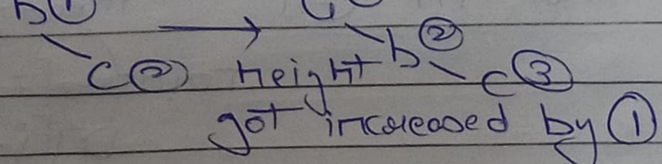


Catalan traversal →

a'+c'+ c(ab)+0
a'+b'+c' +a'+b'+c'

0 + cost(bc) + cost(a) + cost(b) + cost(c)

Here a', b', c' is added as the height got increased by 1 for all nodes.



(42:) represent the number n in minimum sum of perfect square

9 → 3^2 → (1) — answer
7 → 1 + 1 + 1 + 4 → (4)
18 → 9 + 9 → (2)
12 → 4 + 4 + 4 → (3)

LIS method

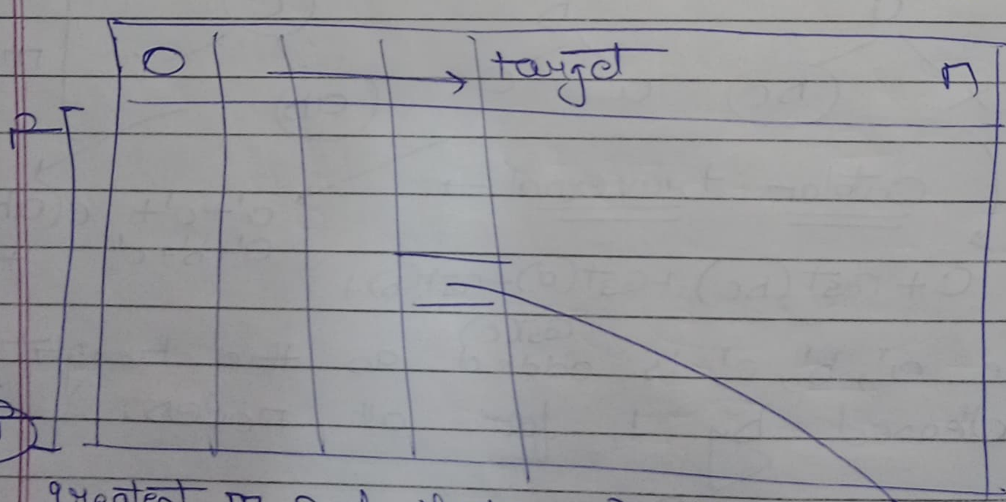
0	1	2	3	4	5	6	7	8	9	10	11
0	1	2	3	1							

no option

only 1 option of 1^2

2 option 1^2 or 2^2

This can also be done with target pair



greatest m such that $m^2 \le n$

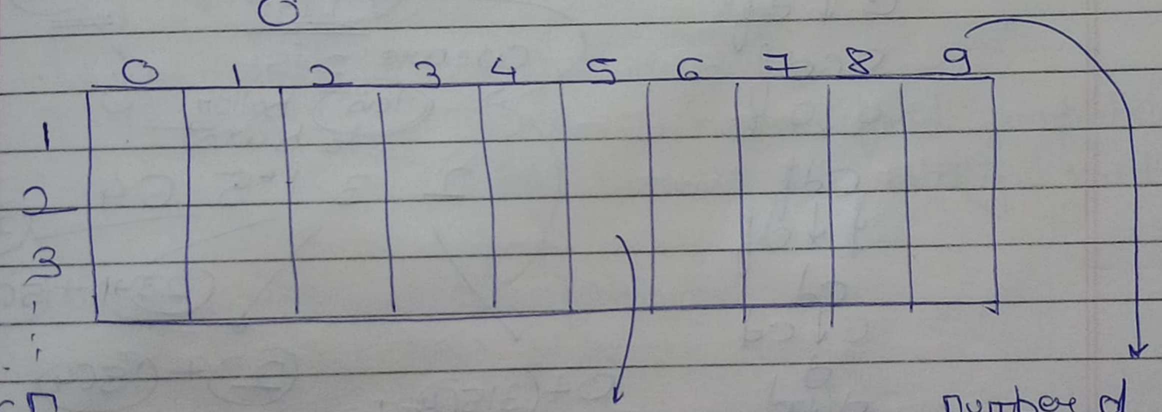
meaning
min([i-1][j],
[i][j-arr]+1)

(43) Kadane algorithm maximum sum ~~of~~ subarray.
Sum the element if the current sum $<=0$
make current sum 0 and restart from current. also keep upating the max at every point.

(Q44) Neumeric key pad dynamic programming.

1 2 3
4 5 6
7 8 9
0

→ you can press self, lef, right, up, docon.



→ number of press

number of way if the last key pressed is 5

number of way if the last key press is 9

$\therefore$ ans = [i-1][2], [i-1][4], [i-1][5]
[i-1][8], [i-1][6]

(Q45) K Concatenation :-

array is concatenated k times
abcdef _______ _______ _____ _____ ... abcdef
if $k=1 \rightarrow$ kadane of single copy
~~ke~~ else if (sum < 0) { kadane of 2 copies
else :- (k-2) sum + kadane of 2 copies.

46) **Burst Ballon :-**

a b c d e f

burst bollon to Collect maximum money.

if burst b the money Collected is abc

if burst f the collected money is ef

Collect maximum money.

a b c d e f

b ↓ abc

a c d e f

e ↓ def

a c d f

a ↓ a c

c d f

f ↓ d f

c d

c ↓ c d

d

d ↓ d

0

total ways to do is n!

IMP

size.

Cocare 2 is last ballon to burst

2 3 1 5 6 4

231 + 64

23 + 1 + 564 + 5

0 + 31564 + 2

2 + 1564 + 3

game

meaning 31564 already bursted & only 2 left in array and cost to burst 2 is 2×1×1 = 2

→ plot

	2	3	1	5	6	4
2	6	9		2315		
3	α	6	45			
1	α	α	15	105		
5	α	α	α	30	140	
6	α	α	α	α	120	140
4	α	α	α	α	α	24

Start ↓

meaning here what maximum we can earn by bursting (2315) from whole array

↓

Most Important

2 3 1 5 6, 4 for 3156

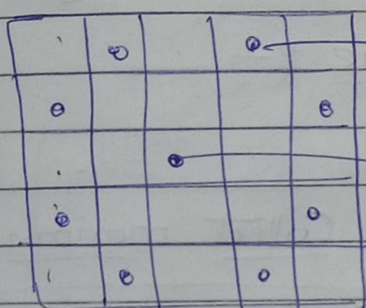
2 × 3 × 4 + 156

Or 2 × 1 × 4 + 2 + 56

2 × 5 × 4 + 31 + 6

2 × 6 × 4 + 315

Q47) Probablity of knight in chess board.



1/8

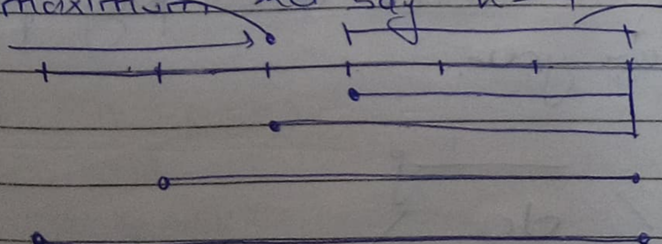
initial position of knight

after n moves find probablity of knight to stay inside the grid

Method :- make 2 planes
1 → for prev
2 → to fill the next
make 2 previous
and and fill the first

Q48) Maximum sum subarray of atleast size k.

Kadane maximum at say k = 4 sum



either evaluate all 4 and time complexity be n^2

Or sum for last k element and add the kadane of it (n − k) element

Make 2 array
(1) → Kadane sum till index i
(2) → Sum of last k element including self

then
loop and find max of array1[i-k] + array2[i]

Kadane till i-k

Sum of last k

(Q49) **Optimal Strategy of game :-**

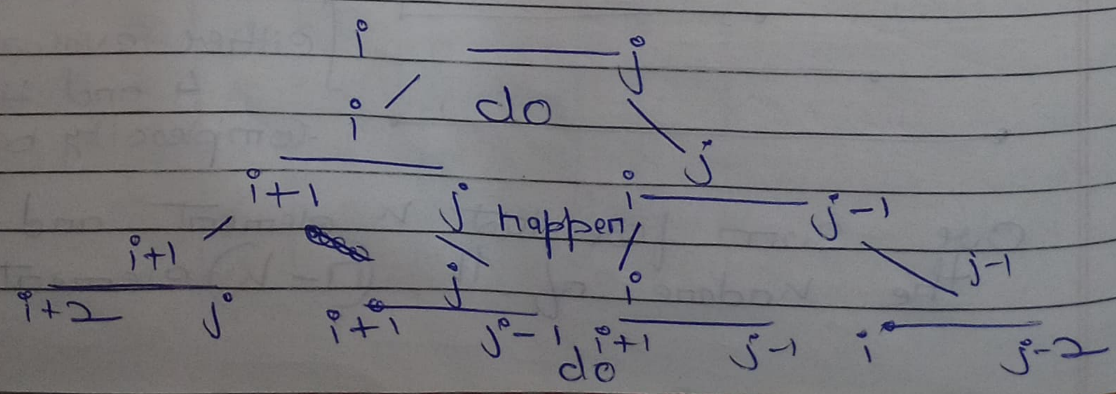
2 players
you vs opponent
Alternate turn
my 1st turn
Extrems we can pick

Q [Collect maximum]
Q → if you can win or not

———— Can be solved by greedy

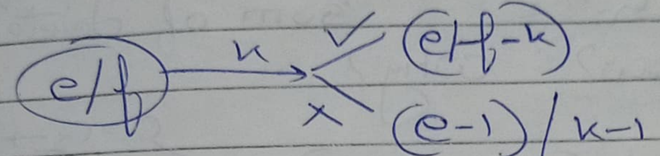
Sum all even coins and all odd coins
and alway pick the even if even > odd
or Odd if odd > even

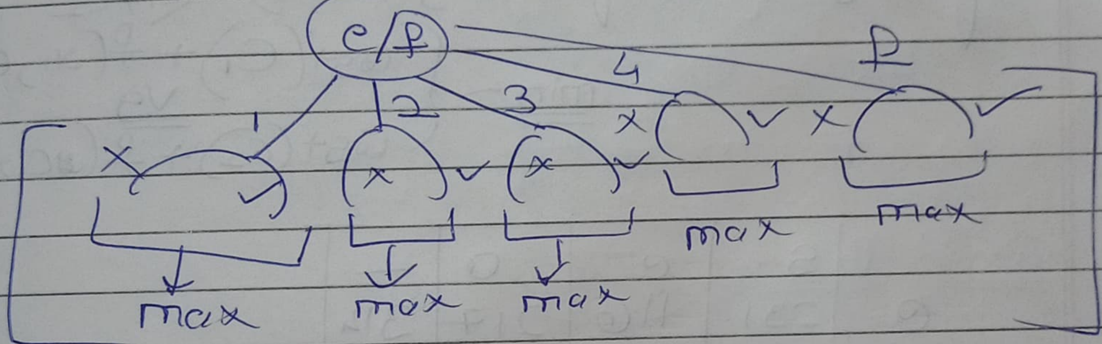
2 things
(1) Play best when your chance
(2) Assume worst when other play plays for you



ans = Math.max (arr[i] + min(f(i+2, j), f(i+1, j-1)) , arr[j] + min(f(i, j-2), f(i+1, j-1)))

Q50) Egg drop. Project estimation problem.





Q51) Longest repeating subsequence (LRS)

a b a c b c
0 1 2 3 4 5

(0 1 3) (1 4 5) → maximum length to find such that no char is shared.

same and repeated

What? How? Why?

↓

LCS (s, s)

↓

One condition just the index of both string should not be same

	a	b	a	c	b	c	-
a	3	3	3	2	2	1	0
b	3	2	2	2	2	1	0
a	3	2	1	1	1	1	0
c	2	2	1	1	1	1	0
b	2	2	1	1	0	0	0
c	1	1	1	1	0	0	0
-	0	0	0	0	0	0	0

Date
Page

Q 52. Minimum ASCII delete sum of 2 string
DP.

S_1, S_2

$S_1' = S_2'$

$S_1'' = S_2''$

$S_1''' = S_2'''$

minimise cost of ASCII sum of deleted char

What? How? Why?

$f(S_1, S_2) \Rightarrow f(c_1 r_1, c_2 r_2)$ < $c_1 = c_2 \rightarrow f(r_1, r_2)$; $c_1 \neq c_2$

$$\min - \begin{cases} Cost(c_1) + f(r_1, c_2 r_2) \\ \text{vs} \\ Cost(c_2) + f(c_1 r_1, r_2) \end{cases}$$

	s	e	a	—
e	231	116	217	314
a	332	217	116	213
t	429	314	213	116
—	313	198	97	0

Q 53 Minimum Cost to make 2 string identical.
(edit distance)

edit distance

		0	1	2	3	4	5	6	7	8
0		d	i	s	t	a	n	c	e	
0	e									
1	d									
2	i									
3	t									

ARFAT
Date
Page

→ insert

	-	d	i	s	t	a	n	c	e
-	0	1	2	3	4	5	6	7	8
e	1	1	2	3	4	5	6	7	7
d	2	1	2	3	4	5	6	7	8
i	3	2	1	2	3	4	5	6	7
t	4	3	2	2	2	3	4	5	6

delete ↓

if arr[i] == arr[j] → move diagonal
else → min of delete, insert, replace

delet → e
insert → s, a, n, c, e
} → total 6 operation.

to make edit to distance
(1) either delete character from edit
(2) either insert character from ~~insert~~ in edit
(3) either replace the character from distance distance to edit

len ≥ 3

Q54) How many subarray create an AP
arithmatic progression.

2 5 9 12 15 18 22 26 30 34
0 0 0 0 1 → 1+1 = 2 0 1 1+1=2 2+1=3
36 38 40 41
0 1 2 0

add prev + 1

total = 1+2+1+2+3+1+2 = 12.

Q55) Largest square submatrix of all 1's

0 1 0 1 0 1
1 0 1 0 1 0
0 1 1 1 1 0
0 0 1 1 1 0
1 1 1 1 1 1

Storage → meaning → direction (small to large) → travel & solve

0	1	0	1	0	1
1	0	1	0	1	0
0	1	1	1	1	0
0	0	1	2	2	0
~~0~~	~~0~~	1	2	3	1

0		0		0	1
	0		0		0
0					0
0	0				0
1	1	1	1	1	1

if 1

min( ) +1

if 1: [(min)(upper, left, diagonal) +1]

if 0: - 0. or we can travel down right to top left.

Q56) Word Break ☆

loves
pep
coding
ice
cream
ice cream
man
go
mango

p	e	p	c	o	d	i	n	g	l	o	v	e	s	m	a	n	g	o	i	c	e	c	r	e	a	m

pepcoding → pepcoding ✓
p - epcoding ✗
pe - l - ✗
pep | coding ✓

p → ✗
pe — pe ✗
p|e ✗
pep → pep ✓] 1
p|ep
pe|p

for → i
  for → j
    string to check → substring
    if dictonary.contains( )
      add →

Q57) Distinct transformation :-

aabc | abc

How many way to make aabc to make abc

aabc / abc

ax → abc / abc → 1

a✓ → abc / bc → bc / bc → 1

Q58) Word wrap

given set of integer array arrange it in lines with maximum k character in lines to reduce cost

ex :- array

4

3 2 2 5 → array

6 → k

blank space ⇒ 3

way 1		way 2
3 _ 2 _	we don't count last line blank space	3 ___
2 ___		2 _ 2 _
5 _		5 _

⇒ cost ⇒ $0 + 4^2 + 0$, $3^2 + 1^2$
16 , ⇒ 10

correct order

approach → make pair class on string and cost (~~1D array~~) → ~~DP~~
↓
int

make 1D array DP
loop through all posibility → from 1 to maximum number of words.

Q59) Number of subsequence of form $a^i + b^j + c^k$ from string str

for abcabc

	0	a	b	c	a'	b'	c'
a+	~~a~~ 0	a 1	a 1	a 1	a, aa', a' 3	3	3
a+b+	~~ab~~ 0	0	ab 1	ab 1	1	2×1+3 5	5
a+b+c+	0	0	0	abc 1	1	1	1×2+5

2a+1

2ab + a →

2abc + ab

When → a → 2a + 1 → join on prev a or start new sequence

b → 2ab + a → join on prev ab or join with prev a

c → 2abc + ab → join with prev abc or join with prev ab

60) <u>Number of ways to paint Nx3 grid</u>