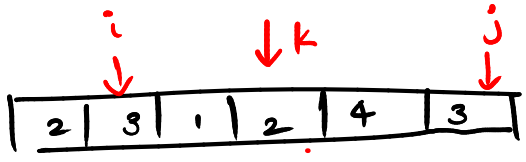
Recap : Matrix Chain Multiplication

ABCD

((AB).C).D OR (A(BC))D

by changing the 2 matrices that are multiplied first, # multiplications required to find same final matrix differs. → min cost matrix multiplication.

i/p : 

← 6 elements | n = 6

2x3 3x1 1x2 2x4 4x3 — 5 matrices.

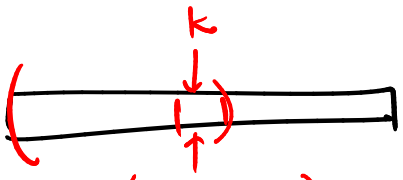
a x b b x c → a.b.c

2x1 2x2 | 2x3 → 2.2.3

2x3 3x1 1x2 | 2x4 4x3

2.3.1 + 2.1.2 = 10

2x4x3 = 24



mcm(arr, i, j)

↓

10 ← mcm(arr, i, k) +

24 ← mcm(arr, k+1, j) +

arr[i-1] x arr[k] x arr[j]

single matrix → cost 0

Code: int mcm(arr, i, j):

if (i >= j) : return 0

if memo[i][j] != -1: return memo[i][j]

int ans = 10e8

for int k = i , k<j. k++

int cost = mcm(arr, i, k) + mcm(arr, k+1, j) +

arr[i-1] x arr[k] x arr[j]

ans = min (cost, ans)

memo[i][j] = ans.

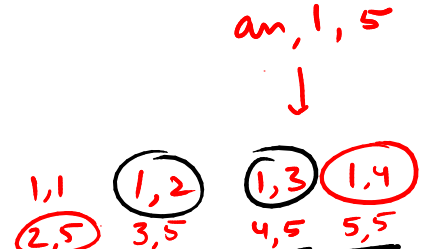
return memo[i][j]

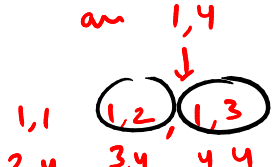
1st call. mcm(arr, 1, n-1)

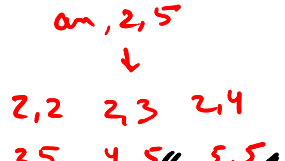
DP

memo[n][n] = {-1}

Overlaps. Memoization → must be done.



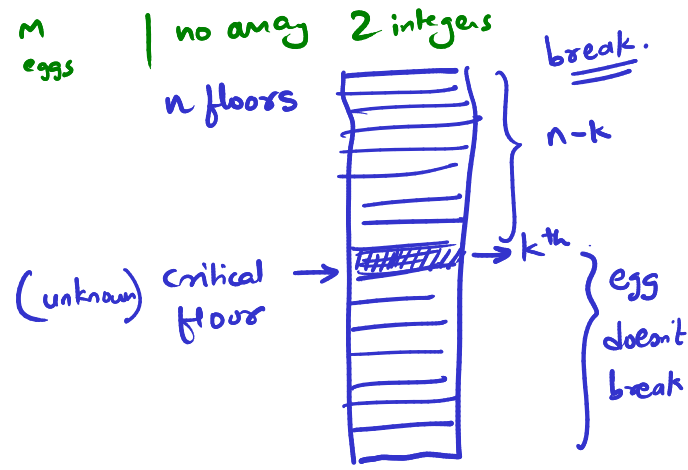




Variations : mcm → you must break the array/string and you don't know the breakpoints. → multiple breaks.

① Egg drop problem :

i/p: n floors, m eggs | no array 2 integers



critical floor → last floor from which if you drop the egg, it won't break.

You are given m eggs.

min drops required to be surely able to find the critical floor.

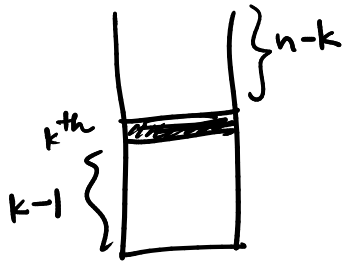
Case1 m=1. m=2 m=3

#drops = n #drops = n/2 #drops = . . .

```
int egg_drop (int m, int n) ⟶ m : eggs  n → floors
    if m==1 : return n
    if n== 0 || n==1 : return n
    int ans = 10e8
    for k=1  k<=n  k++ :
        int drops = max { egg_drop(m-1, k-1) , egg_drop(m, n-k) } + 1
        ans = min (ans, drops)
    return ans.
```

Stop for egg break → if n==0 || n==1 : return n

stop for egg doesn't break → for k=1 k<=n k++ :



accounts for egg break

egg is not breaking

② Palindrome Partitioning

s = "sagar" → 5

a single char is always a palindrome!

ans = 2

s | a g a | r

xyzwuv → n chars

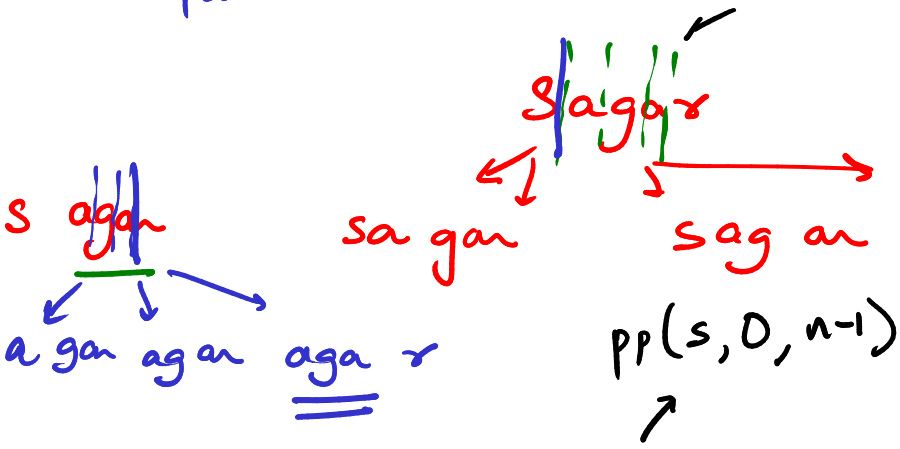
ans = 5

x | y | z | w | u | v

n−1 partitions. Can you do better?

ans = 0 ← malayalam

Q. min no of partitions of a string → each component is a palindrome.



pp(s, 0, n-1)

is_palindrome(str): returns yes if palindrome. O(n)

```
int palindrome_partition(s, i, j):
    if i >= j: return 0
    if is_palindrome(s.substring(i,j)): return 0
    int ans = 10e8
    for (int k = i , k < j , k++)
        int cost = pp(s,i,k) + pp(s,k+1,j) + 1
        ans = min(cost, ans)
    return ans;
```

T.C. $O(n^2)$

O(n) O(n) → O(n) O(n) → $O(n^3)$ or $O(n^2)$

int a = memo[i][k] == -1 ? pp(s,i,k): memo[i][k]

int b ↑

cost = a + b + 1

③ Word break problem → Google!

ip

str = "i|am|a|data|scientist"

ip

list_of_words = ["i", "am", "an", "data", "science", "scientist", "dev", "ist"]

"a"

Q. Can you break this string such that each word is available in the list provided? → True / false.

MCM: Why not greedy?

str = "~~carpet~~|isblack"

list = ["con", "carpet", "is", "black"]

```
bool is_valid (str word):
    list_of_words: set
    return list_of_words.find(word)
```

O(1)

Tc.

carpet|isblack → n

is|T black|T

a

```
bool word_break (str):
    int n = str.size()
    if n==0 : return True
    for int k=1, k<=n, k++
        if is_valid (str.substring(0,k-1)) and word_break(str.substring(k,n-1))
            return True
    return false
```

T

memo[n][n] = {-1}

```
bool a = memo[k][n-1] != -1 ? memo[k][n-1] : word_break(str.substring(k, n-1))
memo[k][n-1] = a ;
```

Thinking process :

① Implement login/signin feature. → No.

① Database where I will put users data. — 1 table — test — test "Hello"

② Write code in Py to connect to the DB. | Test it by fetching Hello.

③ How to define a table | models (FastAPI)

④ I want to run query → table in db | migrations

⑤ How to put values in db via ORM.

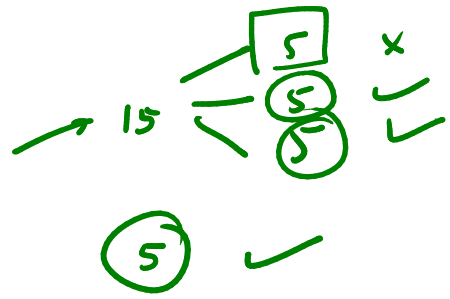
⑥ Can you fetch ?

⑦ Business logic → if else.

⑧ Redirection.

1 table — id username email pass.

* Word break. → 20 lines.



MCM → Theory → Class.

Code : 99/100

Solve as many easy & few liner codes.

CSES problem set. >>> leetcode. — 100/100 AC — 0/100 WA

no BS. | To the point questions

During leetcode.

4th variation.

→ learn as much as Catalan Numbers. — independent.

How to make variations ?

Try it to relate with MCM

① DFS BFS 6.006 MIT → failed. 1st attempt. | 2nd attempt | 3rd attempt.

② Data Structures & Algo by PPC of IIT Khanagpur NPTEL

③ Competive Programmer's Handbook by Antti Laaksonen → visual. breadth. ← 2 pointer | Kadane

Recursion: ↑

④ cp-algorithms.com | e-maxx.ru
↳ Templates. ↳ russian

Amortization

⑤ GfG ⑥ my classmates | seniors → template.

⑥ Codechef certification video lectures → Unacademy.
deleted from YT.

⑦ YT : Understanding the thought process : Q → find the closest similar question they have solved in past.
↓ Errichto ↘ William Lin.

⑧ MIT 6.0483