Todays Content:

→ Matrix Multiplication Basics

→ Matrix Chain Multiplication

→ Longest increasing Subsequence

Matrix Multiplication:

Rule:

$A[3\ \ 4] * B[4\ \ 2] = C[3*2]$

$A[2\ \ 5] * B[5\ \ 3] = C[2*3]$

$A[3\ \ 4] * B[5\ \ 2] =$ not possible

Note: $A_{r_1 * c_1 == r_2 * c_2}\overset{* B}{} = C[r_1\ \ c_2]$

if $c_1 == r_2$ we can multiply matrices

Cost: $A[\overset{r_1}{3}\ \ \overset{c_1}{4}] * B[\overset{r_2}{4}\ \ \overset{c_2}{2}] = C[3*2]$

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 2 & 0 & 1 \\ 1 & 3 & 2 & 1 & 4 \\ 2 & -1 & 0 & 1 & 2 \end{array} \qquad \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 2 & 1 \\ 1 & 1 & 0 \\ 2 & -1 & 1 \\ 3 & 2 & -1 \end{array} \quad = \quad \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 6 & 0 \\ 1 & 15 & 0 \\ 2 & 1 & -2 \end{array}$$

$c[0][0] = 0^{th}$ row in A * 0^{th} col in B

$c[0][1] = 0^{th}$ row in A * 1^{st} col in B

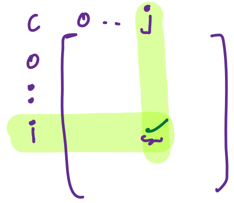
$c[1][0] = 1^{th}$ row in A * 0^{th} col in B

$c[1][1] = 1^{th}$ row in A * 1^{th} col in B

$c[2][0] = 2^{th}$ row in A * 0^{st} col in B

$c[2][1] = 2^{th}$ row in A * 1^{th} col in B

$A[r_1\ \ c_1] == [r_2\ \ c_2] * B = C_{R_1 * c_2}$



$: c[i,j] = i^{th}$ in A * j^{th} col in B

↳ Iterations: c_1 ele r_2 ele // $r_2 == c_1$

↳ // To single element: It will have $r_2 == c_1$ iterations

To get all cells : iterations =

: $c[R_1\ \ c_2]$

: matrix size $= R_1 * C_2 * (r_2 \text{ or } c_1)$

Total iterations Req To mul to $\underset{r_1\ c_1}{A} * \underset{r_2\ c_2}{B}$ will take $R_1 * c_1 * c_2$

Chain Basics:

$$\underset{3*5}{M_1} * \underset{5*7}{M_2} * \underset{7*4}{M_3} = R_{3*4}$$

Case-I : $[M_1 \; M_2] \; M_3$

$$\left[\underset{3*5}{M_1} * \underset{5*7}{M_2}\right] = \left[\underset{3*7}{C} * \underset{7*4}{M_3}\right] = \underset{3*4}{R}$$

iterations = $3*7*5 + 3*7*4 = 189$

Resultant matrix

// no: of iterations are different.

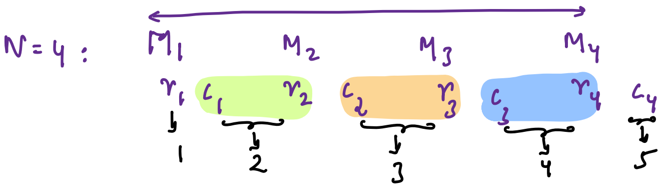
Case-II : $M_1 \; [M_2 \; M_3]$

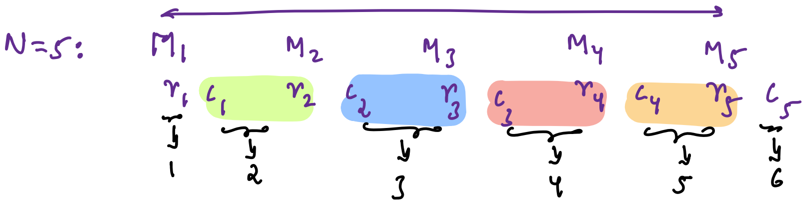
$$\left[\underset{3*5}{M_1} * \underset{5*4}{C}\right] = \left[\underset{5*7}{M_2} \quad \underset{7*4}{M_3}\right]] = \underset{3*4}{R}$$

iterations = $3*5*4 + 5*7*4 = 200$

Q) Given N matrices find min iterations to multiply all matrices?

Input :





$\boxed{\text{For N matrix : Input size = N+1}}$

Extract Inf Input

Input: N=4 : $d[5] = \left\{ \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \\ 3 & 2 & 6 & 4 & 8 \end{array} \right\}$

$\quad\quad\quad\quad\quad r \quad c$

1^{st} mat = d[0] d[1] = 3×2

2^{nd} mat = d[1] d[2] = 2×6

3^{rd} mat = d[2] d[3] = 6×4

4^{th} mat = d[3] d[4] = 4×8

Generalize: N : dim[n+1] = $\{ d_0 \; d_1 \; d_2 \; \dots \; d_n \}$

$\quad\quad\quad\quad\quad r \quad c$

1^{st} mat = d[0] * d[1]

2^{nd} mat = d[1] * d[2]

3^{rd} mat = d[2] * d[3]

i^{th} mat = d[i-1] * d[i] → Inf1

Extra Inf:

Ex: N=5 $d[6] = \left\{ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 2 & 6 & 5 \end{array} \right\}$

→ mul all mat from [1 - 3] res mat size = $\begin{array}{cccc} m_1 & m_2 & m_3 & \\ 2\times3 & 3\times4 & 4\times2 & = R_{2\times2} \end{array}$

→ mul all mat from [2 4] res mat size = $\begin{array}{cccc} m_2 & m_3 & m_4 & = R \\ 3\times4 & 4\times2 & 2\times6 & 3\times6 \end{array}$

→ mul all mat from [1 4] res mat size = $\begin{array}{ccccc} m_1 & m_2 & m_3 & m_4 & = R \\ 2\times3 & 3\times4 & 4\times2 & 2\times6 & 2\times6 \end{array}$

Generalize: N d[N+1] = $\{ d_0 \; d_1 \; d_2 \; d_3 \; .. \; d_N \}$

→ mul all mat from [1 3] res mat size = $\begin{array}{cccc} m_1 & m_2 & m_3 & = R \\ d_0 * d_1 & d_1 * d_2 & d_2 * d_3 & d_0 \; d_3 \end{array}$

Inf2 → mul all mat from [i j] res mat size = $\begin{array}{cccc} m_i & \dots\dots & m_j & = R \\ d_{i-1} * d_i & & d_{j-1} * d_j & d_{i-1} * d_j \end{array}$

: Res [i-j]: $R_{d_{i-1} * d_j}$

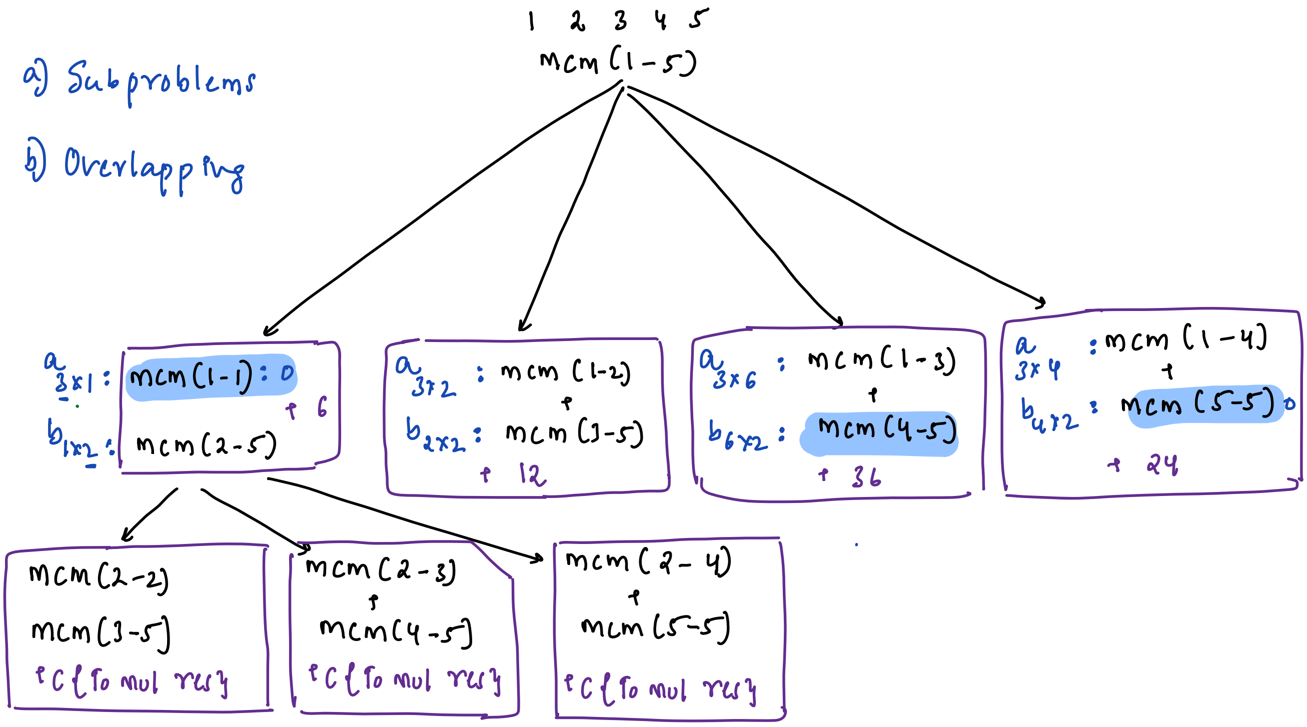
Q) Given N matrices dimensions, calculate min iterations to multiply all of them. dim[N+1] ?

N=5 : $dim[6] = \{ \overset{0}{3}\ \overset{1}{1}\ \overset{2}{2}\ \overset{3}{6}\ \overset{4}{4}\ \overset{5}{2} \} = R_{3\times2}$

min iterations to multiply all matrices from 1-5

a) Subproblems

b) Overlapping


1 2 3 4 5
mcm (1-5)
$a_{3\times1}$: mcm(1-1) : 0
+ 6
$b_{1\times2}$: mcm(2-5)
$a_{3\times2}$: mcm (1-2)
+
$b_{2\times2}$: mcm (3-5)
+ 12
$a_{3\times6}$: mcm (1-3)
+
$b_{6\times2}$: mcm (4-5)
+ 36
$a_{3\times4}$: mcm (1-4)
+
$b_{4\times2}$: mcm (5-5) 0
+ 24
mcm (2-2)
mcm (3-5)
+ C{To mul res}
mcm (2-3)
+
mcm (4-5)
+ C{To mul res}
mcm (2-4)
+
mcm (5-5)
+ C{To mul res}

dp Steps:

dp State : dp(i, j) : Min iterations to mul all matrices from i...j

dp expr : dp(i, j) : i i+1 i+2 .. j-1 j

k = i → j-1

$$dp(i,j) = \min \begin{cases} dp(i, i) + dp(i+1, j) + \text{Cost to mul res mat} \\ dp(i, i+1) + dp(i+2\ j) + \text{Cost to mul res mat} \\ dp(i, i+2) + dp(i+3\ j) + \text{Cost to mul res mat} \\ \vdots \\ dp(i, j-1) + dp(j, j) + \text{Cost to mul res mat} \end{cases}$$

$$\boxed{dp(i,j) = \min\left(\overset{j-1}{\underset{k=i}{\forall}} \underset{d[i-1]*d[k]}{dp(i, k)} + \underset{d[k]*d[j]}{dp(k+1, j)} + d_{i-1} * d_k * d_j \right)}$$

Final ans: // min cost to mul all mat from 1..n :

dp [1][n] : min cost to mul all mat 1..n

10:45 → 10:55pm

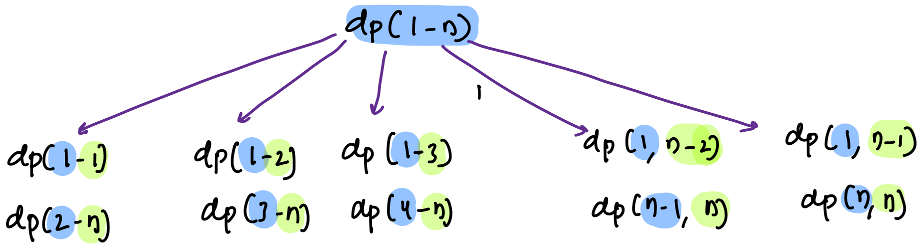


Table:

dp[n+1, n+1]

States × TC for each States

TC: O(N²) * O(N) = O(N3)

```
int dp[n+1][n+1] = INVALID/-1/...
int mcm ( int d[N+1], int i, int j) { // min cost to mul all [i..j]

    if ( i==j) { // Cost to mul to 1 single matrix return 0}

    if ( dp[i][j] ==-1) {
        int c = INT_MAX
        [k = i; k < j ; k++) {
        |  c = min( c,  mcm ( d[], i, k) + mcm ( d[], k+1, j) + d[i-1] * d[k] * d[j])
        }
        dp [i][j] = c
    }
    return dp[i][j]
}
```

2Q) Given ar[N] find length of longest Strictly increasing subsequence

$a_1 < a_2 < a_3 \ldots$

Ex1: $ar[5] = \left\{ \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \\ 9 & 2 & 4 & 3 & 10 \end{array} \right\}$ ans = 3

sub : { 9 4 10 } not inc

sub : { 2 4 10 } inc len = 3

sub : { 2 3 10 } inc len = 3

Ex2: $ar[6] = \left\{ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ 2 & -1 & 6 & 3 & 7 & 9 \end{array} \right\}$ ans = 4

sub : { 2 3 7 9 } len = 4

sub : { -1 3 7 9 } len = 4

sub : { 2 6 3 7 9 } not inc

Idea1: Generate all subseq: TC: $2^n * n$

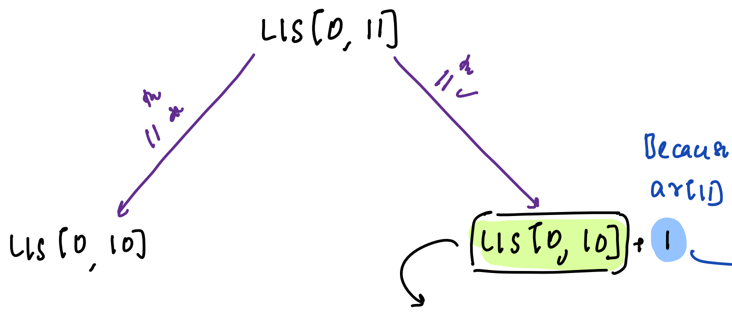
a) Check if subseq is Strictly inc

b) Update it's len & pick max

ar[12] =

0	1	2	3	4	5	6	7	8	9	10	11 ✓
10	3	12	7	2	9	11	20	11	13	6	8

#length of longest inc subseq from [0-11]



Because we picked ar[11]

ar[11]

Say $a_1 .. a_5$: $a_1 < a_2 < a_3 < a_4 < a_5$ 8

are long inc sub ele from [0, 10]

Can we 100% Say $a_5 < 8$: Cannot Say

dp[i] = length of longest inc Subseq from [0, i] ending at i
Subseq last element should be ar[i]

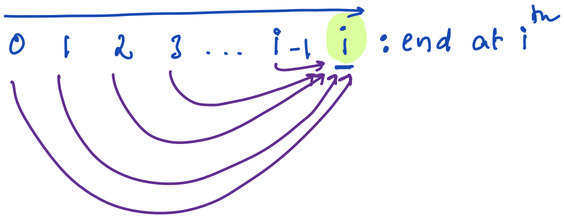
	0	1	2	3	4	5	6	7	8	9	10	11
ar[12] =	10	3	12	7	2	9	11	20	11	13	16	8
dp[] =	1	1	2	2	1	3	4	5	4	5	6	3
Sub =	10	3	10 12	3 7	2	3 7 9	3 7 9 11	3 7 9 11 20	3 7 9 11	3 7 9 11 13	3 7 9 11 13 16	3 7 8

Note: longest sub can end any where
iterate & get max

dp States:

dp[i] = length of longest inc Subseq from [0, i] ending at i

dp(i) =



```
int val=0
j = i-1; j>=0; j--){
    // when can we go from ar[j] ⟶ ar[i]
    if ( ar[j] < ar[i] ){
        val = max( dp[j], val)
    }
}
dp[i] = val + 1 // 1, becaus picking i th number.
```

Final ans : max of dp[]

Table Size : dp[n]

States * TC for each State

TC : O(N) * O(N) = O(N²)

```
int lis(int ar[n])
{
    int dp[n] = -1;
    i=0; i< n; i++) { // Calulate dp[i]   // check case for i=0 it works
        int val=0
        j = i-1; j>=0; j--) {
            // When can we go from ar[j] ⟶ ar[i]
            if ( ar[j] < ar[i] ){
                val = max( dp[j], val)
            }
        }
        dp[i] = val + 1
    }
    return max of dp[]
}
```