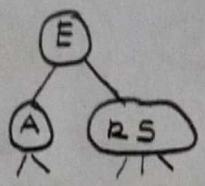
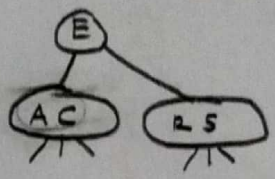
Homework-2

Student name: Göktuğ Gülsoy
Student number: 270201072

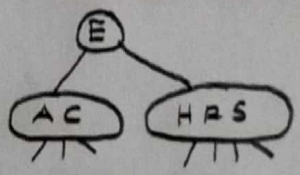
Q1-)



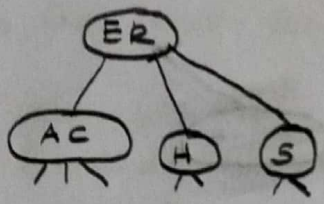
① Insertion of C



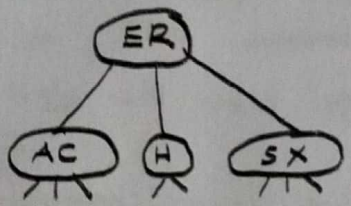
② Insertion of H



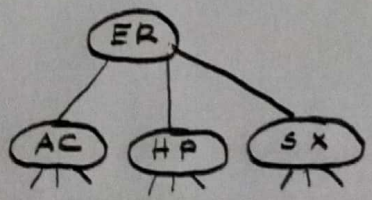
H,R,S Forms a 4-node so we will shift R to parent and split H and S into seperate childs.



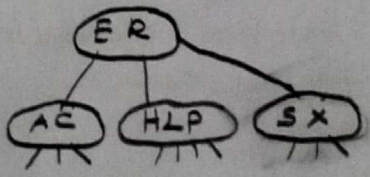
③ Insertion of X



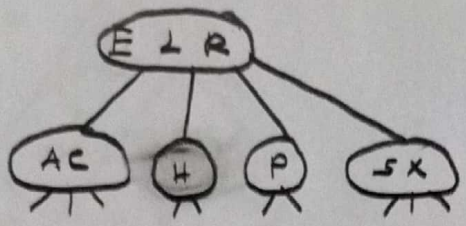
④ Insertion of P



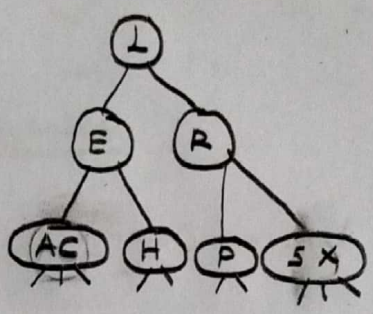
⑤ Insertion of L



H,L,P Forms a 4-node so we shift middle element L to parent and split H and P into seperate childs.



how E, L, R forms a 4-node in that case, we will shift it's middle element as parent and split E and R into seperate childs.



Final 2-3 Tree after inserting C, H, X, P and L

Q2-) a-) Complexity of brute-force algorithm is $\Omega(2^n)$. This means that, if we are given an chain of n matrices then we should try out all 2^n possibilities to find out which paranthesization has the least scalar multiplication number.

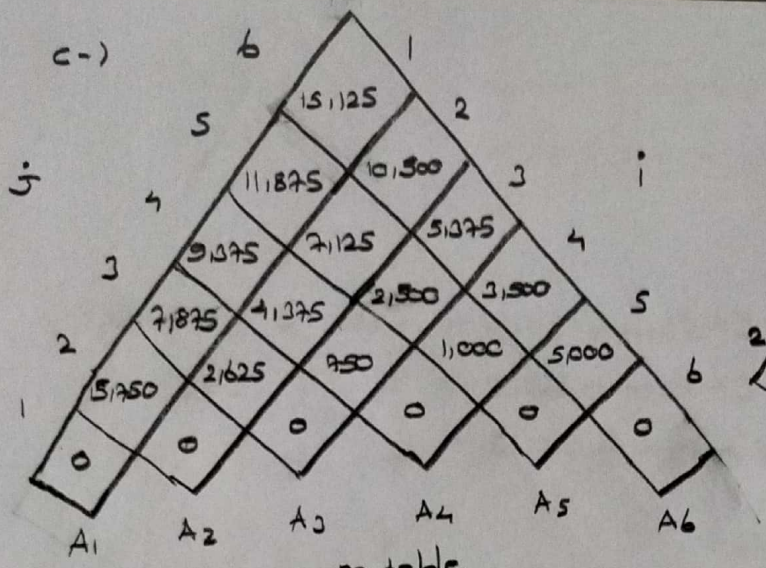
b-) Problem of Finding optimal solution to chain matrix multiplication $A_iA_{i+1} \dots A_j$ can be constructed if we can find an optimal solution to $A_i \dots A_k$ and $A_{k+1} \dots A_j$ such that $i \leq k < j$. In short if we find optimal solutions to these subproblems, they will yield optimal solution to actual problem.

Formulation:

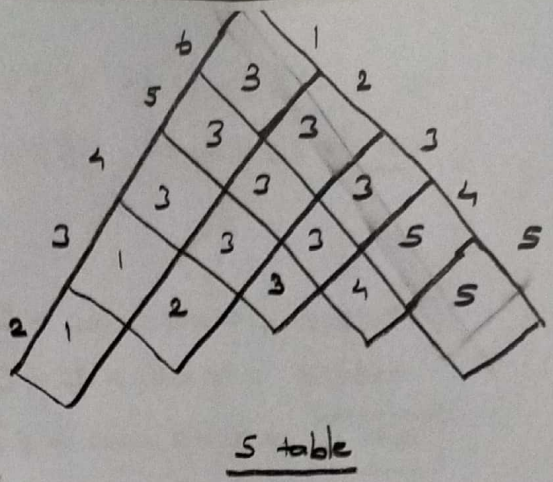
$$m[i,j] = \begin{cases} 0 & \text{if } i = j \\ \min\limits_{i \leq k < j} \{ m[i,k] + m[k+1,j] + p_{i-1} p_k p_j \} & \text{if } i < j \end{cases}$$

where $m[i,j]$ denotes the minimum number of scalar multiplications needed to compute $A_i \dots A_j$ and $p_{i-1} p_k p_j$ denotes the number of scalar multiplications needed to compute matrix product $A_i \dots A_k A_{k+1} \dots A_j$

c-)



m table



S table

$$m[1,2] = \min \begin{cases} m[1,1] + m[2,2] + 30 \times 35 \times 15 = \boxed{15,750} \\ (k=1) \end{cases}$$

$$m[2,3] = \min \begin{cases} m[2,2] + m[3,3] + 35 \times 15 \times 5 = \boxed{2,625} \\ (k=2) \end{cases}$$

$$m[3,4] = \min \begin{cases} m[3,3] + m[4,4] + 15 \times 5 \times 10 = \boxed{750} \\ (k=3) \end{cases}$$

$$m[4,5] = \min \begin{cases} m[4,4] + m[5,5] + 5 \times 10 \times 20 = \boxed{1,000} \\ (k=4) \end{cases}$$

$$m[5,6] = \min \begin{cases} m[5,5] + m[6,6] + 10 \times 20 \times 25 = \boxed{5,000} \\ (k=5) \end{cases}$$

$$m[1,3] = \min \begin{cases} m[1,1] + m[2,3] + 30 \times 35 \times 5 = \boxed{7,875} \\ m[1,2] + m[3,3] + 30 \times 5 \times 5 = 16,500 \\ (k=1) \end{cases}$$

$$m[2,4] = \min \begin{cases} m[2,2] + m[3,4] + 35 \times 15 \times 10 = 6,000 \\ m[2,3] + m[4,4] + 35 \times 5 \times 10 = \boxed{4,375} \\ (k=3) \end{cases}$$

$$m[3,5] = \min \begin{cases} m[3,3] + m[4,5] + 15 \times 5 \times 20 = \boxed{2,500} \\ m[3,4] + m[5,5] + 15 \times 10 \times 20 = 2,750 \\ (k=3) \end{cases}$$

$m[4,6] = \min$ ($k=5$)

- $m[4,4] + m[5,6] + 5 \times 10 \times 25 = 6,250$
- $m[4,5] + m[6,6] + 5 \times 20 \times 25 = \boxed{3,500}$

$m[1,4] = \min$ ($k=3$)

- $m[1,1] + m[2,4] + 30 \times 35 \times 10 = 14,875$
- $m[1,2] + m[3,4] + 30 \times 15 \times 10 = 21,000$
- $m[1,3] + m[4,4] + 30 \times 5 \times 10 = \boxed{9,375}$

$m[2,5] = \min$ ($k=3$)

- $m[2,2] + m[3,5] + 35 \times 15 \times 20 = 13,000$
- $m[2,3] + m[4,5] + 35 \times 15 \times 10 = \boxed{7,125}$
- $m[2,4] + m[5,5] + 35 \times 10 \times 20 = 11,075$

$m[3,6] = \min$ ($k=3$)

- $m[3,3] + m[4,6] + 15 \times 5 \times 25 = \boxed{5,375}$
- $m[3,4] + m[5,6] + 15 \times 10 \times 25 = 9,500$
- $m[3,5] + m[6,6] + 15 \times 20 \times 25 = 10,000$

$m[1,6] = \min$ ($k=3$)

- $m[1,1] + m[2,5] + 30 \times 35 \times 20 = 28,125$
- $m[1,2] + m[3,5] + 30 \times 15 \times 20 = 23,250$
- $m[1,3] + m[4,5] + 30 \times 5 \times 20 = \boxed{11,875}$
- $m[1,4] + m[5,5] + 30 \times 10 \times 20 = 15,375$

$m[2,6] = \min$ ($k=3$)

- $m[2,2] + m[3,6] + 35 \times 15 \times 25 = 18,500$
- $m[2,3] + m[4,6] + 35 \times 5 \times 25 = \boxed{10,500}$
- $m[2,4] + m[5,6] + 35 \times 10 \times 25 = 18,125$
- $m[2,5] + m[6,6] + 35 \times 20 \times 25 = 24,625$

$m[1,6] = \min$ ($k=3$)

- $m[1,1] + m[2,6] + 30 \times 35 \times 25 = 36,750$
- $m[1,2] + m[3,6] + 30 \times 15 \times 25 = 34,250$
- $m[1,3] + m[4,6] + 30 \times 10 \times 25 = \boxed{15,125}$
- $m[1,4] + m[5,6] + 30 \times 10 \times 25 = 12,875$
- $m[1,5] + m[6,6] + 30 \times 20 \times 25 = 26,875$

d-) Complexity of bottom-up dynamic programming algorithm is $O(n^3)$.

Q3-) In Merge Sort each time array is divided into two subarrays recursively, but division into subarrays does not involve overlapping subproblems. For a dynamic programming algorithm to speed up using memoization, there must be overlapping subproblems.

Q4-)

i	1	2	3	4	5	6	7	8
si	3	1	4	3	5	6	7	8
fi	5	4	7	9	9	10	11	12

maximum-size subsets = {a2, a3, a7}
{a2, a3, a8}

a-)

i	1	2	3	4	5	6	7	8
si	3	1	4	3	5	6	7	8
fi	5	4	7	9	9	10	11	12
durations:	2	3	3	6	4	4	4	4

This greedy approach does not provide an optimal solution with {a1, a5} because maximum-sized subsets are {a2, a3, a7} or {a2, a3, a8}.

b-)

i	1	2	3	4	5	6	7	8
si	3	1	4	3	5	6	7	8
fi	5	4	7	9	9	10	11	12

This greedy approach provides one of the optimal solutions which is {a2, a3, a8}.