NAME:- KRUNAL RANK

ADM. NO:- U18CO081

B TECH 3RD YEAR

Scanned with CamScanner

DAA Tutorial 7

Ans 1: The minimum number of operations required will be given by the following chain:-

~~(((AB) C) D) E~~

(A (B (C (DE))))

The number of operations are :-

$3 \times 2 \times 1 + 4 \times 3 \times 1 + 5 \times 4 \times 1 + 1 \times 5 \times 1$

$= 6 + 12 + 20 + 5$

$= \underline{\underline{45}}$

Ans 2: Given weights = 1, 2, 5, 6, 7 Max-weight = 10.

Values = 1, 6, 18, 22, 28

Then, constructing DP table,

	0	1	2	3	4	5	6	7	8	9	10
1	0	1	1	1	1	1	1	1	1	1	1
2	0	1	6	7	7	7	7	7	7	7	7
3	0	1	6	7	7	18	18	19	24	25	25
4	0	1	6	7	7	18	22	23	28	29	29
5	0	1	6	7	7	18	22	28	29	34	35

Hence, the value that can be accomodated in knapsack is 35.

Ans 3: Given,

$S1 = abbacdcba$

$S2 = bcdbbcaac$

$$dp[i][j] = \begin{cases} dp[i-1][j-1]+1 ; & s_1[i]=s_2[j] \\ \max(dp[i-1][j], dp[i][j-1]) \end{cases}$$

Constructing a DP table,

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

As seen from the table,
the longest common subsequence is 5,
bcdba.

Ans 4: Given, P = 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1

Q = 0, 1, 1, 0

Constructing a DP table,

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

Hence, the longest common subsequence has length 4, that is, 0110.

Ans 5: Given, required change = 8
Denominations available = 1, 4, 6

Thus, using DP table, first with denomination 1,

0	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1

Now, denomination 4.

1	1	1	1	2	2	2	2	3

Now, denomination 6,

1	1	1	1	2	2	3	3	4

Thus, answer is 4, the combinations are:-
8x1, 4x1 & 1x4, 2x4, 1x6 & 2x1

Ans 6: The correct option is:-
c) expr2 = max $(l(i-1,j), l(i,j-1))$

The correct definition is:-

$l(i,j) = 0$, if $i=0$ or $j=0$
$= l(i-1, j-1)+1$ if $X[i-1] = Y[i-1]$
$= \max(l[i-1,j], l(i,j-1))$ if $X[i-1] \neq Y[i-1]$

Ans 7: FIB(N):-

1. Create dp array of size N+1
2. $dp[0] \leftarrow 0$
3. $dp[1] \leftarrow 1$
4. For I From 2 to N,
   $dp[I] = dp[I-1] + dp[I-2]$
5. Return $dp[N]$

The above algorithm has time complexity $O(N)$
space complexity $O(N)$.