DAA ASSIGNMENT - 2

Submitted by -
Bhuvan Chadha
B00815783

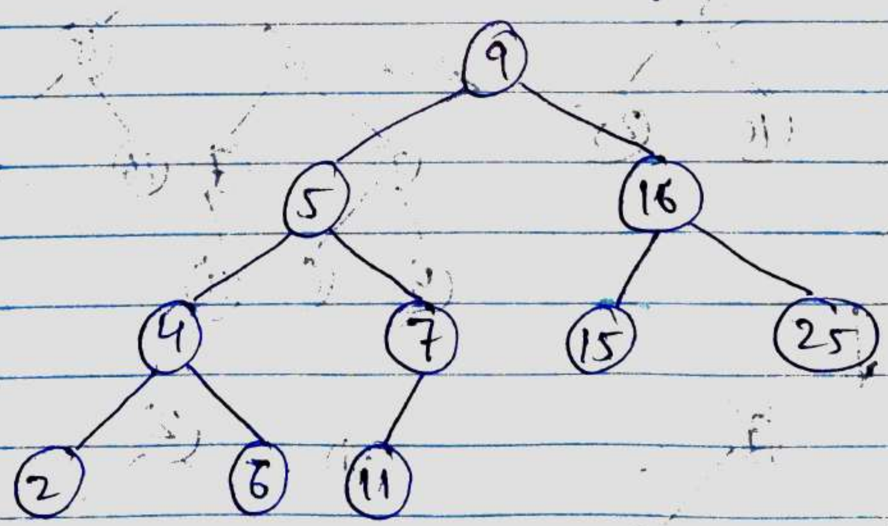
1. We are given array -

11	4	6	15	9	16	2	25	7	5

Array after Sorting in ascending order -

2	4	5	6	7	9	11	15	16	25

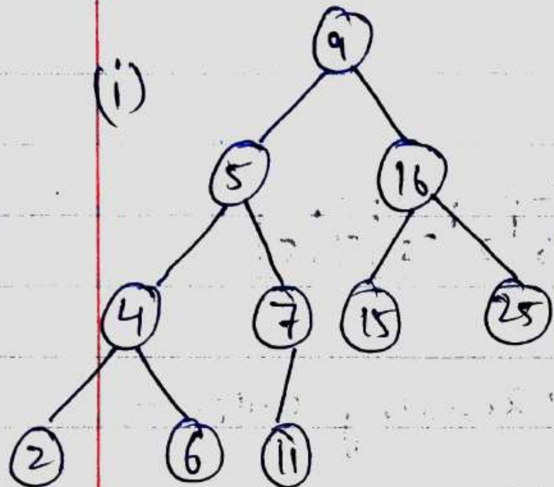
(a) Complete Binary Tree using the Sorted array -



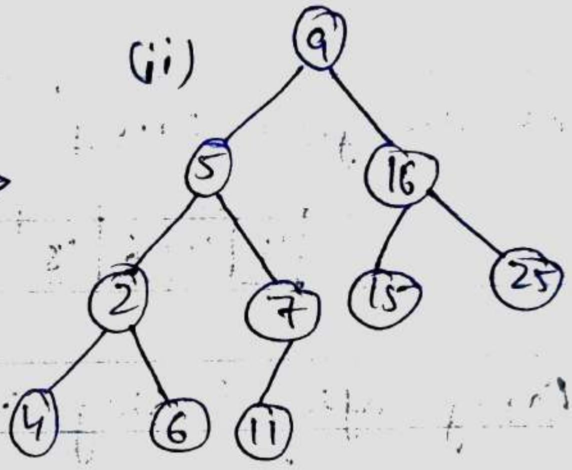
Height of this Complete Binary Tree is $\log_2 10$

(b) Min Heap can be formed using the above Binary Tree following <u>SiftDown Approach</u> :-

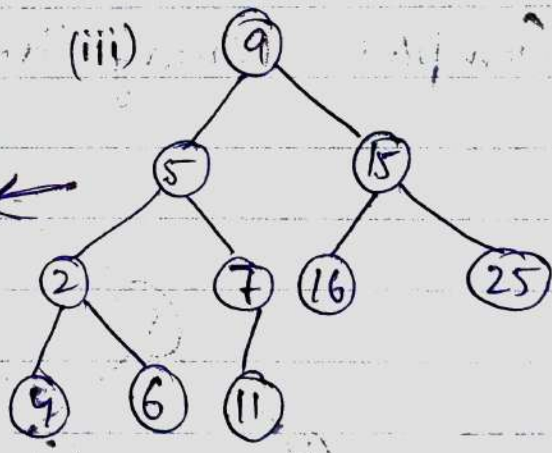
(i)



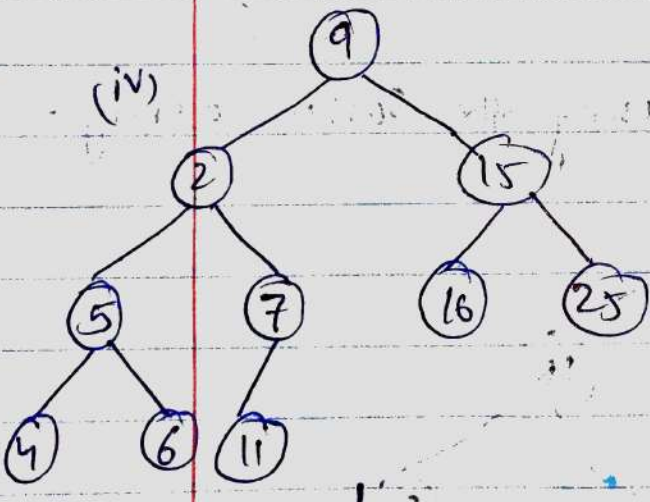
(ii)



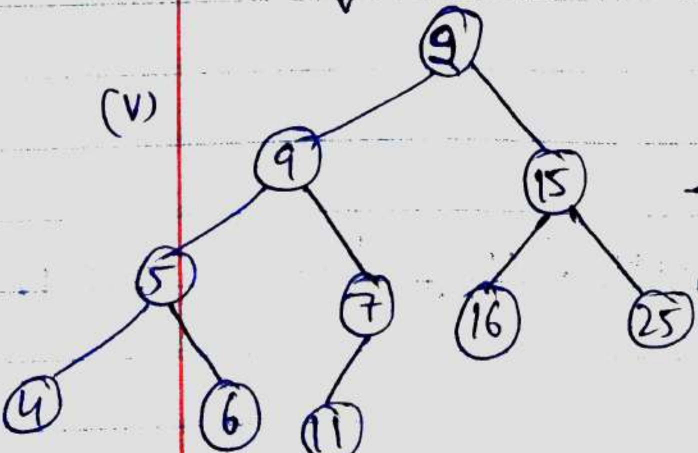
(iii)



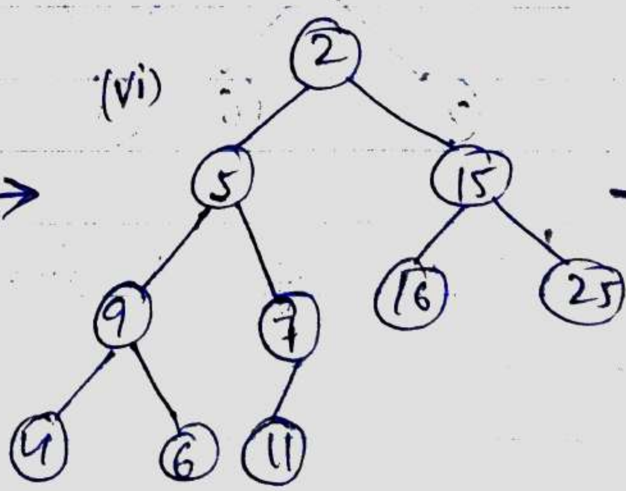
(iv)

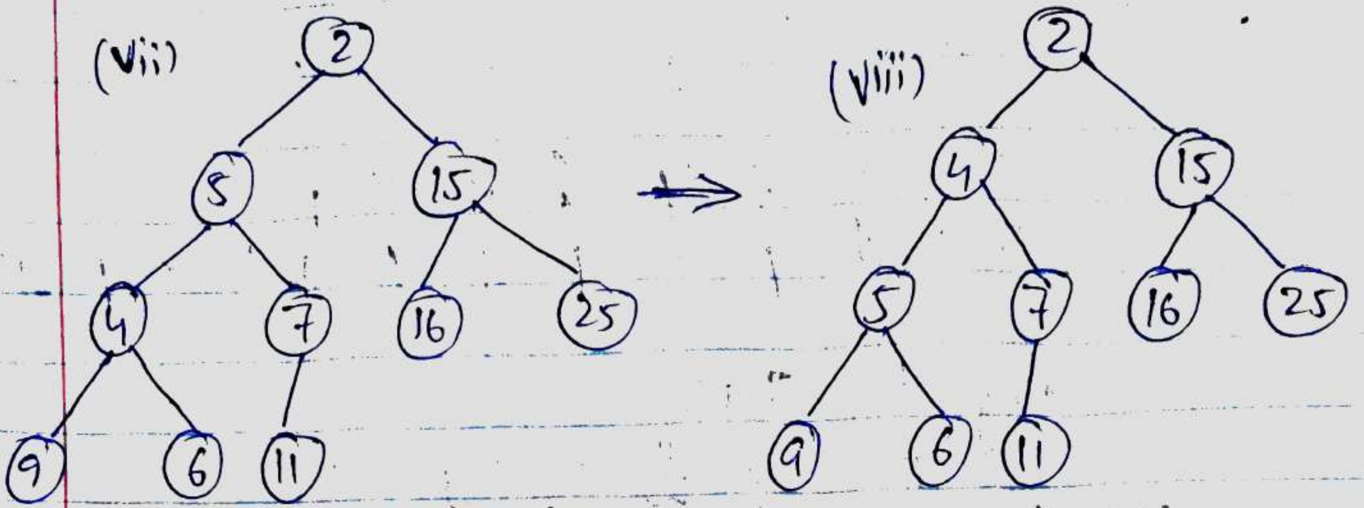


(v)



(vi)

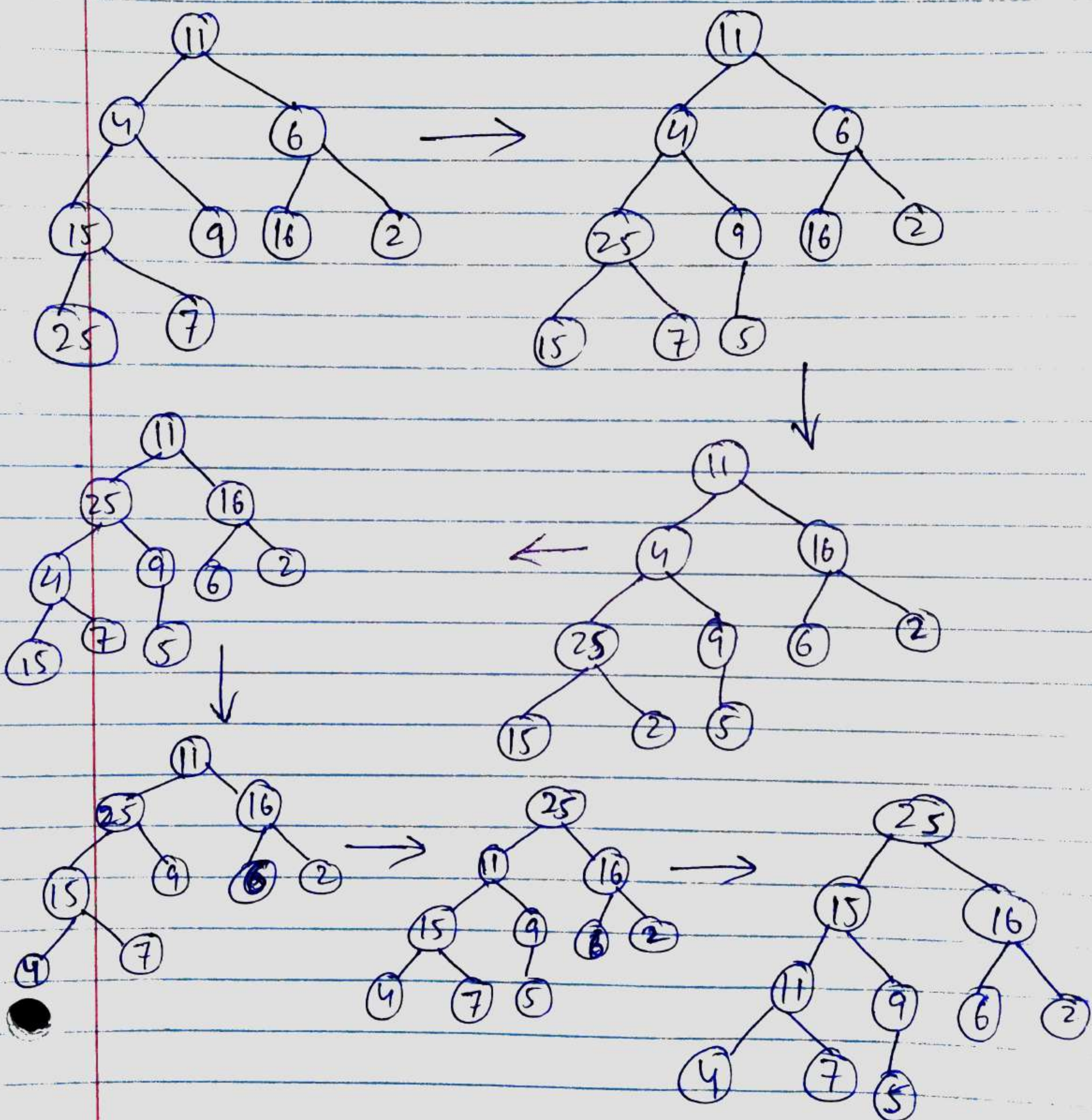




Therefore, the Array Representation of this Min Heap, H ⟹

H	2	4	15	5	7	16	25	9	6	11

(c) Using the Max heap Sorting algorithm, to sort the above Sequence in non-decreasing order, the heap is as follows-



Q. Given two strings -

$$S = \alpha\ \beta\ \alpha\ \mu\ £\ \alpha\ \alpha\ \beta$$
$$T = £\ \mu\ \alpha\ \mu\ \beta\ \mu\ £\ £\ \alpha\ \mu\ \beta$$

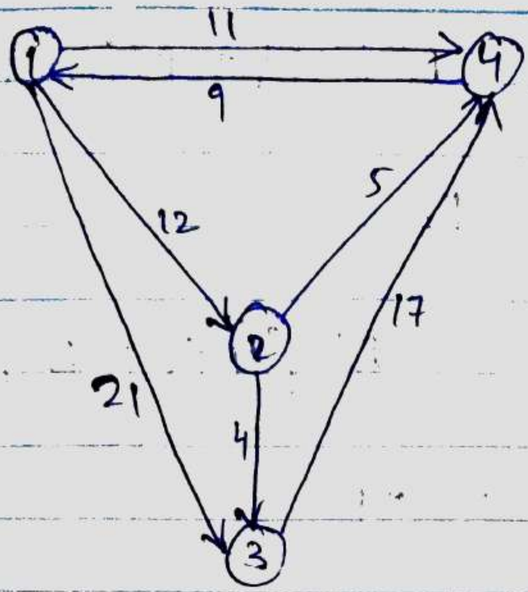
Creating the table :-

			α	β	α	μ	£	α	α	β
		-1	0	1	2	3	4	5	6	7
	-1	0	0	0	0	0	0	0	0	0
£	0	0	0	0	0	0	1	1	1	1
μ	1	0	0	0	0	1	1	1	1	1
α	2	0	1	1	1	1	1	2	2	2
μ	3	0	1	1	1	2	2	2	2	2
β	4	0	1	2	2	2	2	2	2	3
μ	5	0	1	2	2	3	3	3	3	3
£	6	0	1	2	2	3	4	4	4	4
£	7	0	1	2	2	3	4	4	4	4
α	8	0	1	2	3	3	4	5	5	5
μ	9	0	1	2	3	4	4	5	5	5
β	10	0	1	2	3	4	4	5	5	6
			α	β		μ	£	α		β

∴ Following the diagonal arrows, the longest common Subsequence of the given two strings is -

$\underline{\alpha\ \beta\ \mu\ £\ \alpha\ \beta}$

3.



$$D^0 = \begin{array}{c|cccc|} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 12 & 21 & 11 \\ 2 & \infty & 0 & 4 & 5 \\ 3 & \infty & \infty & 0 & 17 \\ 4 & 9 & \infty & \infty & 0 \end{array} \quad , \quad P^0 = \begin{array}{c|cccc|} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \end{array}$$

Now, $D^0[2,3] \qquad D^0[2,1] + D^0[1,3]$

$4 < \infty + 21$

$\therefore \underline{D^1[2,3] = 4}$

$D^0[2,4] \qquad D^0[2,1] + D^0[1,4]$

$5 < \infty + 11$

$\therefore \underline{D^1[2,4] = 5}$

$D^0[3,2] \qquad D^0[3,1] + D^0[1,2]$

$\infty = \infty + 12$

$\therefore \underline{D^1[3,2] = \infty}$

$D^0[3,4] \quad D^0[3,1] + D^0[1,4]$

$17 < \infty + 12$

$\therefore \underline{D^1[3,4] = 17}$

$D^0[4,2] \quad D^0[4,1] + D^0[1,2]$

$\infty > 9 + 12$

$\therefore \underline{D^1[4,2] = 21}$

$D^0[4,3] \quad D^0[4,1] + D^0[1,3]$

$\infty > 9 + 21$

$\therefore \underline{D^1[4,3] = 30}$

Now, keeping the first row and first column same as D^0 matrix, and filling the above calculated values in D^1 matrix, we have -

$$D^1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 12 & 21 & 11 \\ 2 & \infty & 0 & 4 & 5 \\ 3 & \infty & \infty & 0 & 17 \\ 4 & 9 & \underline{21} & \underline{30} & 0 \end{array}, \quad P^1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 10 & 0 & 0 \\ 4 & 0 & 1 & 2 & 0 \end{array}$$

Simultaneously, updating P Matrix with increamenting values where D matrix is getting updated.

Again, repeating the steps for creating D^2 matrix, keeping the 2nd row and 2nd column same as D'.

$D'[1,3]$ $\quad$ $D'[1,2] + D'[2,3]$
$21 > 12 + 4$
$\therefore \underline{D^2[1,3] = 16}$

$D'[1,4]$ $\quad$ $D'[1,2] + D'[2,4]$
$11 < 12 + 5$
$\therefore \underline{D^2[1,4] = 11}$

$D'[3,1]$ $\quad$ $D'[3,2] + D'[2,1]$
$\infty = \infty + \infty$
$\therefore \underline{D^2[3,1] = \infty}$

$D'[3,4]$ $\quad$ $D'[3,2] + D'[2,4]$
$17 < \infty + 5$
$\therefore \underline{D^2[3,4] = 17}$

$D'[4,1]$ $\quad$ $D'[4,2] + D'[2,1]$
$9 < 21 + \infty$
$\therefore \underline{D^2[4,1] = 9}$

$D'[4,3]$ $\quad$ $D'[4,2] + D'[2,3]$
$30 > 21 + 4$
$\therefore \underline{D^2[4,3] = 25}$

$$D^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 12 & 16 & 11 \\ 2 & \infty & 0 & 4 & 5 \\ 3 & \infty & \infty & 0 & 17 \\ 4 & 9 & 21 & 25 & 0 \end{array} \quad , \quad P^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 3 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 1 & 4 & 0 \end{array}$$

Repeating the steps for finding D^4 matrix
Keeping 3rd row and 3rd column same from D^3.

$D^2[1,2] \qquad D^2[1,3] + D^2[3,2]$

$12 < 16 + \infty$

$\therefore \underline{D^3[1,2] = 12}$

$D^2[1,4] \qquad D^2[1,3] + D^2[3,4]$

$11 < 16 + 17$

$\therefore \underline{D^3[1,4] = 11}$

$D^2[2,1] \qquad D^2[2,3] + D^2[3,1]$

$\infty = 4 + \infty$

$\therefore \underline{D^3[2,1] = \infty}$

$D^2[2,4] \qquad D^2[2,3] + D^2[3,4]$

$5 < 4 + 17$

$\therefore \underline{D^3[2,4] = 5}$

$D^2[4,1]$ $\quad$ $D^2[4,3] + D^2[3,1]$

$9 < 25 + \infty$

$\therefore \underline{D^3[4,1] = 9}$

$D^2[4,2]$ $\quad$ $D^2[4,3] + D^2[3,2]$

$21 < 25 + \infty$

$\therefore \underline{D^3[4,2] = 21}$

$$D^3 = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 12 & 16 & 11 \\ \infty & 0 & 4 & 5 \\ \infty & \infty & 0 & 17 \\ 9 & 21 & 25 & 0 \end{bmatrix} \end{array}, \quad P^3 = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 0 \end{bmatrix} \end{array}$$

Again, repeating the steps to find D^4 matrix, Keeping 4th row and 4th column same as D^3.

$D^3[1,2]$ $\quad$ $D^3[1,4] + D^3[4,2]$

$12 < 11 + 21$

$\therefore \underline{D^4[1,2] = 12}$

$D^3[1,3]$ $\quad$ $D^3[1,4] + D^3[4,3]$

$16 < 11 + 25$

$\therefore \underline{D^4[1,3] = 16}$

$D^3[2,1]$ $D^3[2,4] + D^3[1,4]$

$\infty > 5 + 11$

$\therefore \underline{D^4[2,1] = 16}$

$D^3[2,3]$ $D^3[2,4] + D^3[4,3]$

$4 < 5 + 25$

$\therefore \underline{D^4[2,3] = 4}$

$D^3[3,1]$ $D^3[3,4] + D^3[4,1]$

$\infty > 17 + 9$

$\therefore \underline{D^4[3,1] = 26}$

$D^3[3,2]$ $D^3[3,4] + D^3[4,2]$

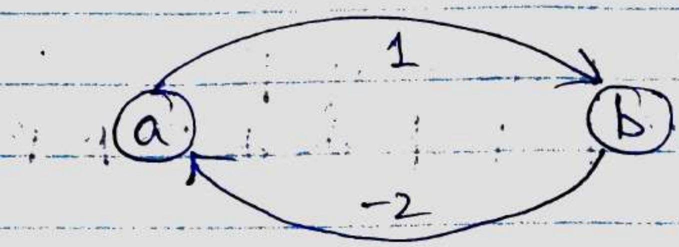
$\infty > 17 + 21$

$\therefore \underline{D^4[3,2] = 38}$

The final D^4 matrix with shortest paths is —

$$D^4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 12 & 16 & 11 \\ 16 & 0 & 4 & 5 \\ 26 & 38 & 0 & 17 \\ 9 & 21 & 25 & 0 \end{bmatrix} \end{array}, \quad P^4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 0 & 3 & 0 \\ 5 & 0 & 0 & 0 \\ 6 & 7 & 0 & 0 \\ 0 & 1 & 4 & 0 \end{bmatrix} \end{array}$$

4. Floyd's Algorithm



We are given a negative weight Cycle graph, because, the result of the weights of the edges is negative overall.

$+1 + (-2) = \underline{-1}$

There is no Shortest path between any pair of vertices 'a' and 'b' which form part of a negative cycle.

Now for Floyd's Algorithm to yeild correct results, it assumes that there are no negative Cycles in a graph. But if there are any negative cycles, it Can detect them.
Hence, to detect negative Cycles, we Can trace the diagonal path of the path matrix and check for any negative weights/values. If found, then Floyd's Algorithm will yeild negative Cycle results.

5. We are given two heaps of sizes n1 and n2. Now, to merge these heaps, we first, copy the nodes of heap 1 to heap 2. Using the first for loop and then use SiftDown approach to build the heap (or Heapify), using the 2nd for loop.

The given algorithm -

```
for (i=1; i<=n2; i++)
    bt1 [n1+i] = bt2 [i]
last = n1 + n2
for (i=n1; i>0; i=i-1)
    SiftDown (bt1, i)
```

This algorithm is correct as it follows the correct procedure of loops and recursive function call.

The time complexity of the two for loops will be $O(n1 + n2)$.

Hence, $T(n)$ for the given algorithm = $\underline{O(n)}$

6. The binomial value (or binomial Coefficient) of $\binom{n}{k}$ is given by -

$$C(n,k) = C(n-1, k-1) + C(n-1, k)$$

Now, for computing Binomial value of $\binom{8}{6}$, we can use the same relation.

Here, if $k=0$, or $k=n$, then $C=1$.

$\therefore C(8,6) = C(8-1, 6-1) + C(8-1, 6)$
$\Rightarrow C(7,5) + C(7,6)$

Using the same formula for calculating further,

$\Rightarrow C(6,4) + C(6,5) + C(6,5) + C(6,6)$
$\Rightarrow C(5,3) + C(5,4) + 2[C(5,4) + C(5,5)] + 1$
$\Rightarrow C(4,2) + C(4,3) + C(4,3) + C(4,4) + 2[C(4,3) + C(4,4)] + 3$

$\Rightarrow C(3,1) + C(3,2) + 4[C(3,2) + C(3,3)] + 6$
$\Rightarrow C(2,0) + C(2,1) + C(2,1) + C(2,2) + 4[C(3,2)] + 10$
$\Rightarrow 4[C(2,1) + C(2,2)] + 16$
$\Rightarrow 4[C(1,0) + C(1,1)] + 20$
$\Rightarrow 4(1+1) + 20$

$\therefore C(8,6) = \underline{28}$

We Create a B-Table to Store the (n, K)' Values after each step. Dynamic Programming is based on the same Strategy. Instead of recursively Calling the function to calculate C(n, K) repeatedly, we can create a B Table and Store the Values in it and using them whenever needed during the Calculation.

K →

n ↓	1	2	3	4	5	6
1	1					
2	1	1				
3	1	3	1			
4	1	4	4	1		
5	1	5	10	5	1	
6	1	6	20	15	6	1
7	1	7	35	35	21	7
8	1	8	56	70	56	28

7. Sum of Subsets Problem.

(1) Recursive formula which can be associated with Sum of Subsets Problem can be generalized as follows –

We assume a set given – S, with number of ~~At~~ elements, n, and given sum value, Sum, with recursive function – Sum_of_Subsets ( ).

$$\therefore \text{Sum\_of\_Subsets}(S, n, Sum) = \begin{cases} \text{Sum\_of\_Subsets}(S, n-1, Sum) \\ \cancel{\text{Sum\_of\_Subsets}(S, n-1, Sum} \end{cases}$$

Time Complexity can be generalized by –

$$T(n) = \begin{cases} 1, & \text{if } n = 0 \\ T(n-1) + n, & n \geq 1 \end{cases}$$

(2) Following is the Pseudocode (Algorithm) for Sum-of-Subsets ( ) problem.

```
Sum_of_Subsets(S, n, Sum)
{   n = len(S)
    if Sum == 0
        return True
    else if (Sum != 0 and n == 0)
        return False

    if (S[n-1] > Sum)
        return Sum_of_Subsets(S, n-1, Sum)

    return Sum_of_Subsets(S, n-1, Sum) or
        Sum_of_Subsets(S, n-1, Sum-S[n-1])
}
```

(3) We are given a set sample and the target sum -

$S = \{1, 2, 4\}$, $Sum = 6$

Creating a B-matrix for Dynamic Programming for the given algorithm -

Here, if $P_i + P_j = 6$, we will return True else, the result will be False.

$P_1 + P_2 = 1+2 \Rightarrow 3$
return false.

$P_2 + P_3 = 2+4 \Rightarrow 6$
return True

$P_1 + P_3 = 1+4 \Rightarrow 5$
return false.

B-Matrix –

Sum →

i ↓

	1	2	3	4	5	6
1	True	True	False	False	False	False
2.	False	False	False	True	False	False
3	false	False	false	False	False	False
4	false	False	false	False	false	False

8. Given the recurrence relation –

$$b(n) = \sum_{k=0}^{n-1} b(k)\, b(n-1-k) \quad \text{for } n>0,$$
$$b(0) = 1$$

Let the number of distinct binary trees with n nodes be $b(n)$.
K represents the number of nodes of the left subtree of a binary tree such that
$0 \le k \le n-1$

So, the right subtree should have $n-1-k$ nodes according to the given recurrence relation.
Hence, the total number of distinct binary trees that can be formed to satisfy the recurrence relation is –
$\underline{b(k) \cdot b(n-1-k)}$

9. We are given -

Key	A	B	C	D
	1	2	3	4
Probabilities	0.1	0.2	0.4	0.3

for finding the Optimal Binary Tree of the given keys with the given probabilities, we have the recurrence relation -

$$C[i, j] = \min\{C[i, k-1] + C[k+1, j]\} + \sum_{s=i}^{j} P_s\},$$

$$C[i,i] = P_i$$

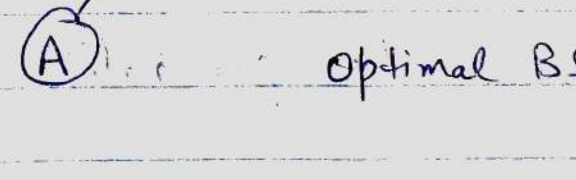
Creating a Table, and storing the values of all nodes

i \ j	0	1	2	3	4
1	0	0.1	0.4	1.1	1.7
2		0	0.2	0.8	1.4
3			0	0.4	1.0
4				0	0.3
5					0

This table stores the calculated values for $C[i, j]$.
We can establish another table for storing the values of K for minimum values of C.

i \ j	0	1	2	3	4
1		1	2	3	3
2			2	3	3
3				3	3
4					4
5					

Therefore, from the tables, Optimal Binary Search Tree can be formed as below-



Optimal BST

10. We are given j requesters to ~~tea~~ rent the house, where $j = 1, 2, 3, 4, 5, 6$.
Each requester is willing to pay $v(j)$ dollars for the period of $t(j)$.

(1) The recurrence relation can be established as -

$$R = \begin{cases} MaxProfit(j) = 0 & ; \quad j = 0. \\ Max\left[v(j) + MaxProfit(j), MaxProfit(j-1)\right], & \text{otherwise} \end{cases}$$

Here, MaxProfit(j) is the maximum profit a requester can provide if the j^{th} requester is chosen.

(2) The time complexity of the algorithm will be
$T(n) = O(n \log n)$

Using binary search in sorted array to find non-conflicting request.