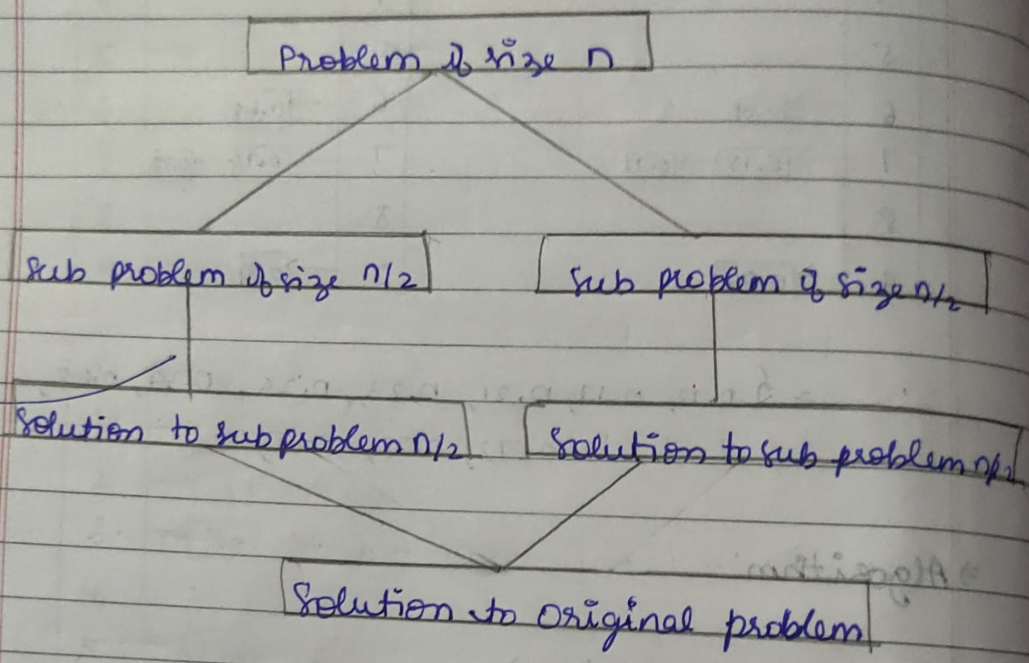
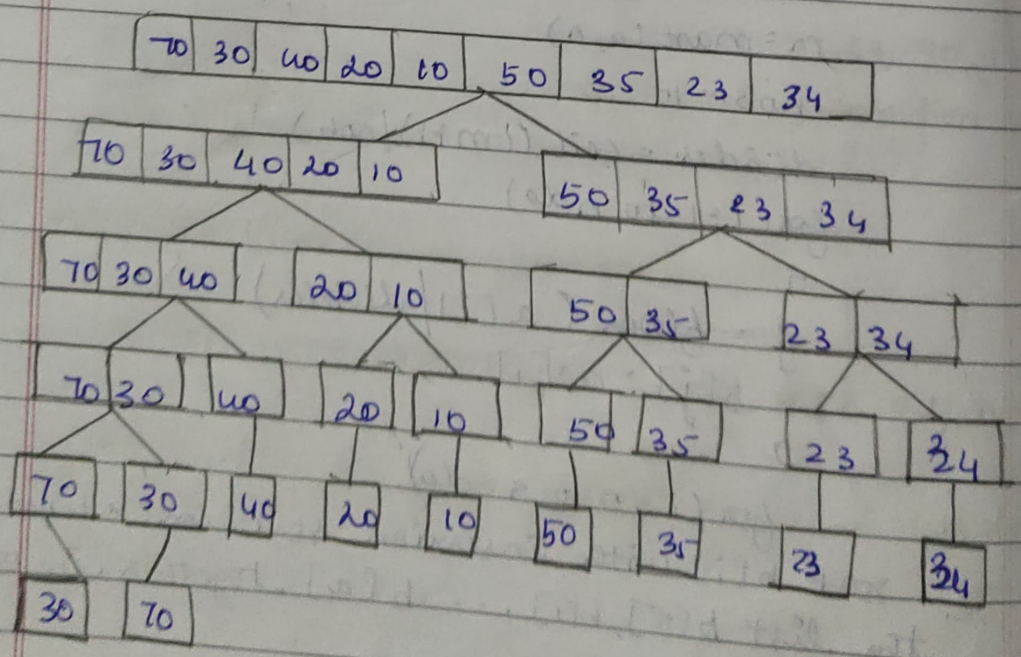
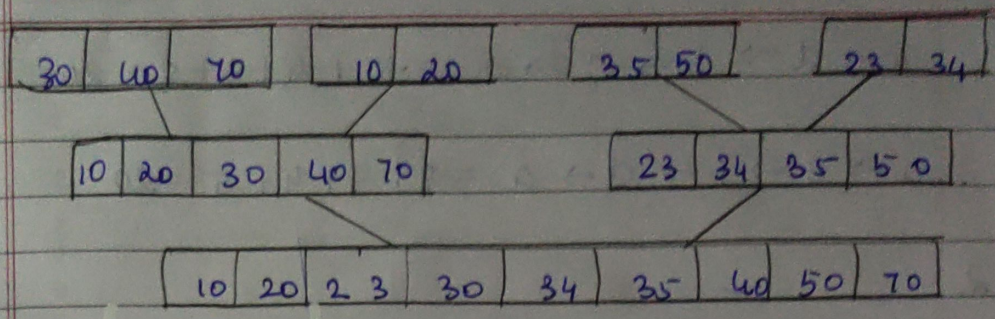
UNIT - 3

- Brute force attack
- Divide & conquer Method







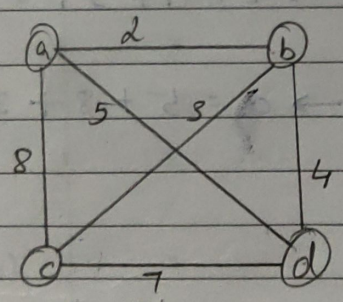
⇒ Brute force Attack:

A straight forward problem solving technique that ~~tries~~ tries every possible solution until the correct one is found.

⇒ Travelling Salesman Problem (TSP)

A problem find the minimum distance

1. Given 'n' cities with known distance b/w each pair find the shortest route that passes through all the cities exactly one before returning to the starting city



$(n-1)!$

$a \to b \to d \to c \to a = 2 + 4 + 7 + 8 = 21$

$\boxed{a \to d \to c \to b \to a = 5 + 7 + 3 + 2 = 17}$

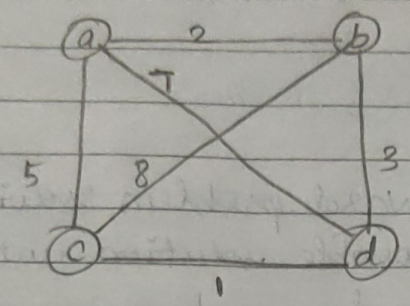
$a \to c \to d \to b \to a = 8 + 7 + 4 + 2 = 21$

$\boxed{a \to b \to c \to d \to a = 2 + 3 + 7 + 5 = 17}$

$a \to d \to b \to c \to a = 5 + 4 + 3 + 8 = 20$

$a \to c \to b \to d \to a = 8 + 3 + 4 + 5 = 20$

2.



$(n-1)! = (4-1)! = 3! = 6$

$\boxed{a \to b \to d \to c \to a = 2 + 3 + 1 + 5 = 11}$

$a \to d \to c \to b \to a = 7 + 1 + 8 + 2 = 18$

$\boxed{a \to c \to d \to b \to a = 5 + 1 + 3 + 2 = 11}$

$a \to b \to c \to d \to a = 2 + 8 + 1 + 7 = 18$

$a \to d \to b \to c \to a = 7 + 3 + 8 + 5 = 23$

$a \to c \to b \to d \to a = 5 + 8 + 3 + 7 = 23$

⇒ **Knap Sack Problem:-**

1. Given some items pack the knapsack to get the maximum total value. Each items has some weight and some value. Total item that we can carry is no more than some fixed number

∴ The time complexity is $(2^n - 1)$

Given 'n' items weights : $W_1, W_2, \ldots W_n$

Value : $V_1, V_2, \ldots V_n$

Knapsack = W

Item	weight	value
1	2	20
2	5	30
3	10	50
4	5	10

Knapsack Capacity = 16

Item	Total weight	Total Value
{1}	2	20
{2}	5	30
{3}	10	50
{4}	5	10
{1, 2}	7	50
{1, 3}	12	70
{1, 4}	7	30
{2, 3}	15	80
{2, 4}	10	40
{3, 4}	15	60
{1, 2, 3}	X	X
{1, 2, 4}	12	60
{1, 3, 4}	X	X
{2, 3, 4}	X	X
{1, 2, 3, 4}	X	X

combination

Therefore, the optimal solution is {2, 3} with total weight 15 and total Value is 80

Time complexity : $2^n - 1 = 2^n$

$2^n - 1$

2^4

2^{n+1}

$16 - 1$

15

2.

item	weight	value
0	1	1
1	3	4
2	4	5
3	5	7

$2^n - 1 = 15$

$2^4 - 1$

$16 - 1 = 15$

Capacity = 7

item	total weight	total value
{0}	1	1
{1}	3	4
{2}	4	5
{3}	5	7
{0, 1}	4	5
{0, 2}	5	6
{0, 3}	6	8
{1, 2}	7	9
{1, 3}	X	X
{2, 3}	X	X
{0, 1, 2}	X	X
{0, 1, 3}	X	X
{0, 2, 3}	X	X
{1, 2, 3}	X	X
{0, 1, 2, 3}	X	X

→ Assignment Problem:-

1. Let us consider There are n people and n jobs each person has to be assigned only one job when the jth job is assigned to p^{th} person, the cost incurred is referred by

$C = C[P, J]$, Time complexity is $= n!$

Person \ Job	J_1	J_2	J_3	J_4
P_1	9	2	7	8
P_2	6	4	3	7
P_3	5	8	1	8
P_4	7	6	9	4

P1:- P → 1 2 3 4

J → {1, 2, 3, 4} = 9+4+1+4 = 18

1 2 3 4

{1, 2, 4, 3} = 9+4+8+9 = 30

1 2 3 4

{1, 3, 2, 4} = 9+3+8+4 = 24

1 2 3 4

{1, 3, 4, 2} = 9+3+8+6 = 26

1 2 3 4

{1, 4, 2, 3} = 9+7+8+9 = 33

1 2 3 4

{1, 4, 3, 2} = 9+7+1+6 = 23

P2:- 1 2 3 4

{2, 1, 3, 4} = 2+6+1+4 = 13

1 2 3 4

{2, 1, 4, 3} = 2+6+8+9 = 25

1 2 3 4

{2, 3, 1, 4} = 2+3+5+4 = 14

1 2 3 4

{2, 3, 4, 1} = 2+3+8+7 = 20

$\{2, 4, 1, 3\} = 2 + 7 + 5 + 9 = 23$

$\{2, 4, 3, 1\} = 2 + 7 + 1 + 7 = 17$

P3:- $\{3, 1, 2, 4\} = 7 + 6 + 8 + 4 = 25$

$\{3, 1, 4, 2\} = 7 + 6 + 8 + 6 = 27$

$\{3, 2, 1, 4\} = 7 + 4 + 5 + 4 = 20$

$\{3, 2, 4, 1\} = 7 + 4 + 8 + 7 = 26$

$\{3, 4, 1, 2\} = 7 + 7 + 5 + 6 = 25$

$\{3, 4, 2, 1\} = 7 + 7 + 8 + 7 = 29$

P4:- $\{4, 1, 2, 3\} = 8 + 6 + 8 + 9 = 31$

$\{4, 1, 3, 2\} = 8 + 6 + 1 + 6 = 21$

$\{4, 2, 1, 3\} = 8 + 4 + 5 + 9 = 26$

$\{4, 2, 3, 1\} = 8 + 4 + 1 + 7 = 20$

$\{4, 3, 1, 2\} = 8 + 3 + 5 + 6 = 22$

$\{4, 3, 2, 1\} = 8 + 3 + 8 + 7 = 26$

Minimal cost is 13, by assigning

Job 2 ⇒ person 1
Job 1 ⇒ person 2
Job 3 ⇒ Person 3
Job 4 ⇒ Person 4

2.

	J_1	J_2	J_3
P_1	9	2	7
P_2	6	4	3
P_3	5	8	1

6 combination

P_1:- $\{\overset{1}{1}, \overset{2}{2}, \overset{3}{3}\} = 9+4+1 = 14$

$\{\overset{1}{1}, \overset{2}{3}, \overset{3}{2}\} = 9+3+8 = 20$

P_2: $\{\overset{1}{2}, \overset{2}{1}, \overset{3}{3}\} = 2+6+1 = 9$

$\{\overset{1}{2}, \overset{2}{3}, \overset{3}{1}\} = 2+3+5 = 10$

P_3: $\{\overset{1}{3}, \overset{2}{1}, \overset{3}{2}\} = 7+6+8 = 21$

$\{\overset{1}{3}, \overset{2}{2}, \overset{3}{1}\} = 7+4+5 = 16$

Minimal cost is 9, by assigning

Job 2: Person 1
Job 1: Person 2
Job 3: Person 3

⇒ Closest pair Algorithm:-

Given 'n' points in the plan find a pair with smallest Euclidean distance between them. When Brute force method is used it is required to check all pair of coins P and Q with $\theta(n^2)$ comparisons.

Euclidean distance $= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Page No.:
Date: / /

⇒ Algorithm brute force closests points (P):-

```
    // P is the list of points
    dmin ← ∞
for i ← 1 to n-1 do
    for j ← i+1 to n do
        d ← sqrt ((xi - xj)^2 + (yi - yj)^2)
    if d < dmin then
        dmin ← d; index 1 ← i; index 2 ← j;
return index 1, index 2.
```

⇒ Convex Hull problem:-

It is the smallest convex set containing the points A convex set is one in which if we connect any two points in the set the line segment connecting these points must also be in the set.

* Theorem:-

The convex hull of any set S of $n > 2$ points is a convex polygon with the vertices at some of the points of S

* Algorithm

```
For all points p in S
    for all points q in S
        if p != q
            draw a line from p to q
            if all points in S except p and q
lie to the left of the line
                add the directed vector pq to the
```

TNPL

Page No.:
Date: / /

→ DIVIDE AND CONQUER ALGORITHM:-

Alg. divide and conquer (P)
{
if small (P)
then return S(P)
else
{
divide 'P' into smaller instances P1, P2, ... Pk
Apply divide and conquer to each sub proble
return combine D and C(P1), Dand C(P2) ... D a
C (Pk))
}
}

→ BINARY SEARCH (Always only sorted list should be
⇒ Very fast and efficient method
↳ Requires elements in sorted order
→ Find the mid element
→ Compare key element with mid element
→ If key is lower than mid element, we h
to work in the left side and if the key
greater than mid element we have to se
in the right side of the mid element.

→ Algorithm Binary Search (K, n, x)
// K → array containing the elements
// n → total number of elements
// x → element to be searched
lower ← 0, upper ← n − 1
while lower ≤ upper
mid ← (lower + upper) / 2
if (x = k[mid]) then

~~else if~~
lower ← mid + 1
else
write ("Element found at", mid)
Quit
end if
end if
end while
write ("Element not present in the test")
end

⇒ Problem:-

0	1	2	3	4	5	6	7	8	9
1,	22,	35,	40,	43,	56,	75,	83,	90,	98.

K = 40

↓ lower ↓ upper

lower = 0, upper = n-1 = 10-1 = 9
0 ≤ 9 → T
mid ← (0+9)/2 = 4 [Take only floor value]
if (40 < 43) → T
upper ← 3
0 ≤ 3 → T
mid → (0+3)/2 = 1
if [40 < 22] → false
if [40 > 22] → True
Lower ← 2
2 ≤ 3 → T
mid ⇒ (2+3)/2 = 2

if (40 < 35) → false
else if (40 > 35) → T
lower ← 2+1 = 3

3 ≤ 3 → T

mid ← (3+3)/2 = 3
if (40 < 40) → F
else if (40 > 40) → F
else
Element found at position 3

2. 43, 53, 58, 69, 72, 75, 80, 81 K = 90
0 1 2 3 4 5 6 7
↓ lower ↓ upper

Lower = 0, upper = n-1 = 8-1 = 7

0 ≤ 7 → T
mid ← (0+7)/2 = 3
if (90 < 69) → F
else if (90 > 69) → T
lower ← 3+1 = 4

4 ≤ 7 → T
mid ← (4+7)/2 = 5
if (90 < 75) → F
else if (90 > 75) → T
lower ← 4+1 = 5

5 ≤ 7 → T
mid ← (5+7)/2 = 6
if (90 < 80) → F
else if (90 > 80) → T
lower ← 5+1 = 6

6 ≤ 7 → T
mid ← (6+7)/2 = 6
if (90 < 80) → F
else if (90 > 80) → T
lower ← 6+1 = 7

$7 \leq 7 \rightarrow T$

mid $\leftarrow (7+7)/2 = 7$

If $(90 < 81) \rightarrow F$

else if $(90 > 81) \rightarrow T$

lower $\leftarrow 7+1 = 8$

$8 \leq 7 \rightarrow F$

"Element not present in the list"

⇒ Analysis of Binary Search:-

$$C_{worst}(n) = C_{worst}(n/2) + 1$$

Dividing the array

comparison

$C_{worst}(1) = 1$

$n = 2^k, \; k = \log_2 n$

$$C_{worst}(2^k) = C_{worst}\left(\frac{2^k}{2}\right) + 1$$

$$C_{worst}(2^k) = C_{worst}(2^{k-1}) + 1 \rightarrow (1)$$

By using backward substitution,

$$C_{worst}(2^{k-1}) = C_{worst}(2^{k-1-1}) + 1$$

$$C_{worst}(2^{k-1}) = C_{worst}(2^{k-2}) + 1 \rightarrow (2)$$

Page No.:
Date: / /

sub equ ② in eq ①

$C_{worst}(2^k) = C_{worst}(2^{k-2}) + 1 + 1$

$C_{worst}(2^k) = C_{worst}(2^{k-2}) + 2 \rightarrow ③$

$C_{worst}(2^{k-2}) = C_{worst}(2^{k-2-1}) + 1$

$C_{worst}(2^{k-2}) = C_{worst}(2^{k-3}) + 1 \rightarrow ④$

sub eqn ④ in eqn ③

$C_{worst}(2^k) = C_{worst}(2^{k-3}) + 3$

$C_{worst}(2^k) = C_{worst}(2^{k-k}) + k$

$C_{worst}(2^k) = C_{worst}(2^0) + k$

$= C_{worst}(1) + k$

$= 1 + k$

$\boxed{C_{worst}(2^k) = 1 + \log_2 n}$

$C_{worst}(2^k) = \log_2 n$

→ Quick Sort:-

lb 0	1	2	3	4	5	6	7	8	9 UP
42	23	74	11	65	58	94	36	99	87

↑ start ↑ end

Pivot = 42

0	1	2	3	4	5	6	7	8	9
42	23	74	11	65	58	94	36	99	87

↑ start ↑ start ↑ start ↑ end ↑ end ↑ end

swap ~~x~~ start & end

0	1	2	3	4	5	6	7	8	9
42	23	36	11	65	58	94	74	99	87

↑ end ↑ start

swap pivot and end

0	1	2	3	4	5	6	7	8	9
11	23	36	42	65	58	94	74	99	87

P1 P2

Pivot = 65

0	1	2	3	4	5
65	58	94	74	99	87

↑ start ↑ start ↑ start ↑ end ↑ end

0	1	2	3	4	5
65	58	94	74	99	87

↑ end ↑ start

swap with pivotal element

0	1	2	3	4	5
58	65	94	74	99	87

Pivotal = 94

94	74	99	87

start, start, start, end

94	74	99	87

start, end → swapping

94	74	87	99

start, end, end, start

94	74	87	99

end, start

as end goes beyond start it shld be swapped with pivot

87	74	94	99

P_3, P_4

Pivot = 87

P_3

87	74

start, end, start

74	87

Now, combine all P_1, P_2, P_3, P_4

0	1	2	3	4	5	6	7	8	9
11	23	36	42	58	65	74	87	94	99

Need to add step by step.

P1:-

0	1	2	3
11	23	36	42

P1 & P2 :

0	1	2	3	4	5
11	23	36	42	58	65

P1 & P2 & P3 :

0	1	2	3	4	5	6	7
11	23	36	42	58	65	74	87

P1 & P2 & P3 & P4 :

0	1	2	3	4	5	6	7	8	9
11	23	36	42	58	65	74	87	94	99

→ fastest algo

⇒ Algorithm Quicksort (K, lower, upper)

```
If (lower < upper) then
begin
    i ← lower +1
    j ← upper
    flag ← 1
      Key ← K [lower]
    while (flag)
  begin
    while ( K [i] <= key)
          i ← i+1
    End while
      while (K[j] > key)
          j ← j - 1
      End while
      if (i < j) then
      K [i] ↔ K[j]
        i ← i + 1
        j ← j - 1
        else
```

Page No.:
Date: / /

flag ← 0
End if
End while
k[j] ← k[lower]
Quick sort (k, lower, j-1)
Quick sort (k, j+1, upper)
End if
End Quick sort.

⇒ Analysis of Quick sort:-

$T(n) = T(n/2) + T(n/2) + 1$

$T(n) = 2T(n/2) + 1$

$T=0$,
$T(0) = 2T(0/2) + 1$
$= 0 + 1$
$T(0) = 1$
$n = 1$
$T(1) = 2T(1/2) + 1$

using Master's Method:
$T(n)$

Best case:-
$T(n) = T(n/2) + T(n/2) + 1$
$T(n) = 2T(n/2) + 1$
$T(n/2) = 2T(n/3) + 1$

$T(n) = 2[2T(n/4) + 1] + 1$

$T(n) = 4T(n/4) + 2$
$T(n) = 8T(n/8) + 3$

$T(n) = 2^3 T(n/2^3) + 3$

$\boxed{k=3}$

$2^k = n$

$\boxed{k = \log_2 n}$

$T(n) = 2^k T(n/2^k) + k$

$= nT(n/n) + \log_2 n$

$\boxed{T(n) = n + \log_2 n}$

→ <u>Worst case</u>:-

$T(n) = T(n-1) + n$

$T(n-1) = T((n-1)-1) + (n-1)$

$T(n-1) = T(n-2) + (n-1)$

$T(n) = T(n-2) + (n-1) + n$

$T(n-2) = T(n-3) + (n-2)$

$T(n) = T(n-3) + (n-2) + (n-1) + n$

$\vdots$

$T(n) = T(n-k) + (n-(k-1)) + (n-(k-2)) + n$

$T(n) = T(n-k) + (n-k+1) + (n-k+2) + n$

$\boxed{k=n}$

$T(n) = T(n-1) + (n-n+1) + (n-n+2) + n$

$= T(0) + T(1) + T(2) \cdots + T(n)$

$= 1 + 2 + 3 \cdots$

$= \frac{n(n+1)}{2}$

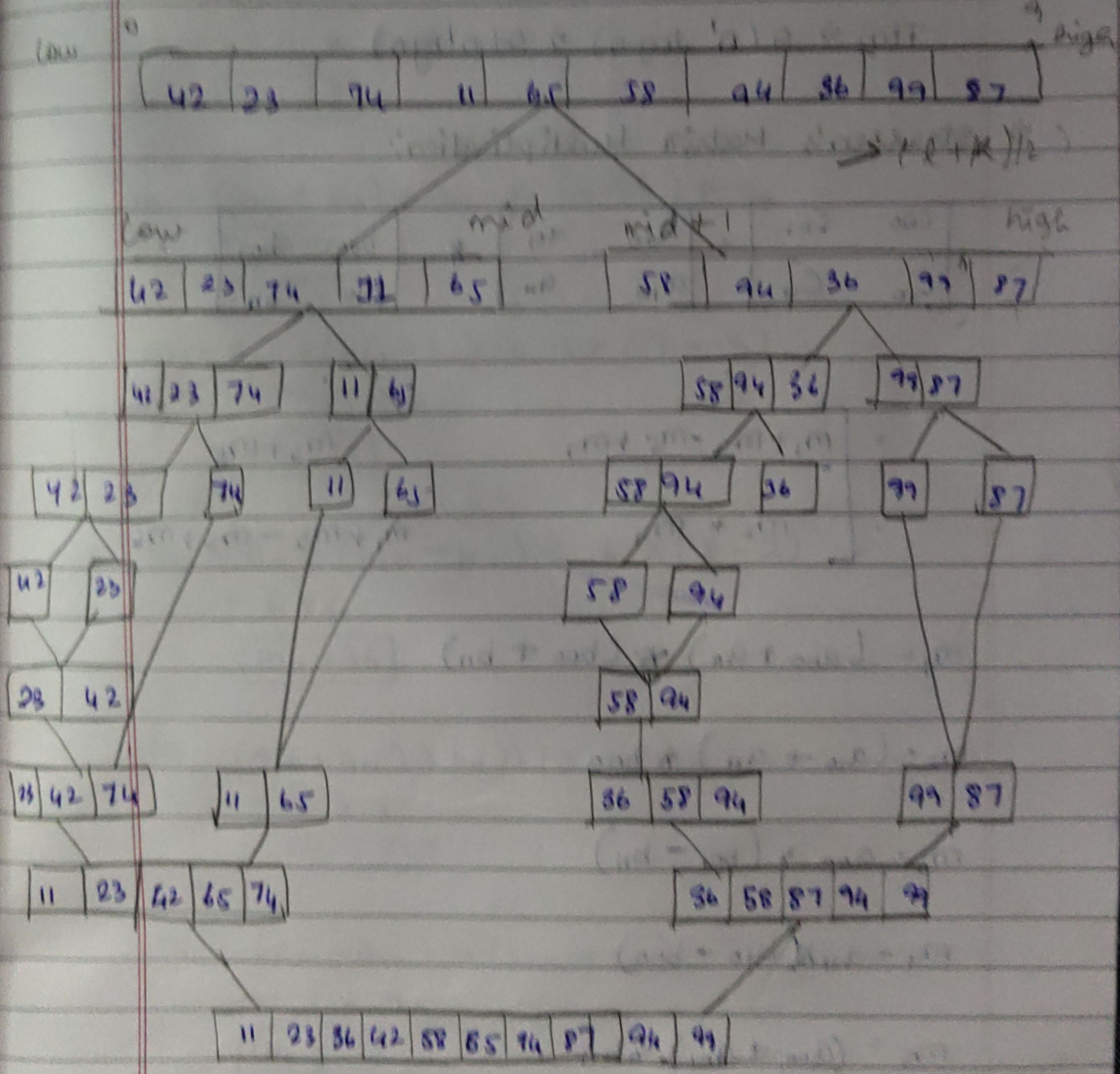
$T(n) = O(n^2)$

⇒ Merge Sort:

```
Algorithm merge (k, low, mid, high)

  i ← low , j ← mid+1 ; l ← 0
while (i ≤ mid) and (j ≤ high)
  if (k[i] < k[j]) then
      temp (l) ← k[i]
   i ← i+1
   l ← l+1
  else
   temp [l] ← k[j]
     j ← j+1
   l ← l+1
   end if
   end while
   if (i > mid) then
      while (j ≤ high)
      temp (l) ← k[j]
        j ← j+1
        l ← l+1
    end while
    else
     while (i ≤ mid)
       temp[l] ← k[i]
        i ← i+1
        l ← l+1
    end while
    end if
  Repeat for m=0 to l step 1
    k [low+m] ← temp[m]
  End Repeat
  End merge
```

```
Algorithm mergesort (A, low, high)
  if (low < high) then
      mid = (low + high)/2
  mergesort (low, mid)
  mergesort (mid+1, high)
  merge(low, mid, high)
  End if
  End mergesort.
```



procedure to write in Exam ⇒ basic tech defn of algo algo explanation example

Page No.: Date: / /

- Analysis of merge sort:-

$T(n) = T(n/2) + T(n/2) + n$

$T(n) = 2T(n/2) + n$

$a = 2, \quad b = 2, \quad d = 1$

$a = b$

$T(n) \Rightarrow \Theta(n^1 \log n) \Rightarrow \Theta(n \log n)$

⇒ Strassen's Matrix Multiplication:-

$$\begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} * \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix}$$

$$= \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6 \end{bmatrix}$$

$m_1 = (a_{00} + a_{11}) * (b_{00} + b_{11})$

$m_2 = (a_{10} + a_{11}) * b_{00}$

$m_3 = a_{00} \times (b_{01} - b_{11})$

$m_4 = a_{11} \times (b_{10} - b_{00})$

$m_5 = (a_{00} + a_{01}) \times b_{11}$

$m_6 = (a_{10} - a_{00}) \times (b_{00} + b_{01})$

$m_7 = (a_{01} - a_{11}) \times (b_{10} + b_{11})$

TNPL

Page No.:

Date: / /

1. $\begin{bmatrix} 3 & 5 \\ 4 & 6 \end{bmatrix} \times \begin{bmatrix} 2 & 7 \\ 8 & 3 \end{bmatrix}$

$a_{00} = 3$, $a_{01} = 5$, $a_{10} = 4$, $a_{11} = 6$

$b_{00} = 2$, $b_{01} = 7$, $b_{10} = 8$, $b_{11} = 3$

$m_1 = (a_{00} + a_{11}) * (b_{00} + b_{11})$

$= (3 + 6) * (2 + 3)$

$= 9 * 5 \Rightarrow 45$

$\boxed{m_1 = 45}$

$m_2 = (a_{10} + a_{11}) * b_{00} \Rightarrow (4 + 6) * 2$

$\boxed{m_2 \Rightarrow 20}$

$m_3 = a_{00} \times (b_{01} - b_{11}) \Rightarrow 3 \times (7 - 3)$

$\Rightarrow 3 \times 4$

$\boxed{m_3 = 12}$

$m_4 = a_{11} \times (b_{10} - b_{00}) \Rightarrow 6 \times (8 - 2)$

$= 6 \times 6$

$\boxed{m_4 = 36}$

$m_5 = (a_{00} + a_{01}) \times b_{11} \Rightarrow (3 + 5) \times 3$

$= 8 \times 3$

$\boxed{m_5 = 24}$

$m_6 = (a_{10} - a_{00}) \times (b_{00} + b_{01}) = (4 - 3) \times (2 + 7)$

$= 1 \times 9$

$\boxed{m_6 = 9}$

$m_7 = (a_{01} - a_{11}) * (b_{10} + b_{11}) = (5 - 6) \times (8 + 3)$

$\boxed{m_7 = -11}$

Page No.:
Date: / /

$$= \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6 \end{bmatrix}$$

$$= \begin{bmatrix} 45 + 36 - 24 + (-11) & 12 + 24 \\ 20 + 36 & 45 + 12 - 20 + 9 \end{bmatrix}$$

$$= \begin{bmatrix} 46 & 36 \\ 56 & 46 \end{bmatrix}$$

10.9.25

Unit - IV

Greedy Approach & Dynamic programming

⇒ 3D Approach:-

- Most of the problems have 'n' inputs and require us to obtain a subset that satisfies some constrains.

- Any subset the satisfies these constraints is called a feasible solution.

- We need to find a feasible solution that either maximes or minimies the objective function.

- A feasible solution that does this is called optimal solution. (2M)

Eg:- $c > 10$ $\{20, 5, 10, 20, 70, 65\}$

↓

$\{20, 70, 65\}$ → feasible solution

↓

70 → Optimal solution

⇒ Algorithm:-

```
Algorithm Greedy (a,n)
  //a(1:n) contains 'n' inputs
  {
  solution = ∅
  for i=1 to n do
  {
    x = select (a);
    if feasible (solution, x) then
      solution = Union (solution, x)
  }
```

return solution
}

→ Application:-

1. Minimum Spanning
2. Single source shortest path
3. Simple Scheduling Problems
4. Huffmann

} for optimal

• Approx soln for greedy:-

Eg:- • Travelling salesman problem
• knapsack problem

→ Optimal merge patterns:-

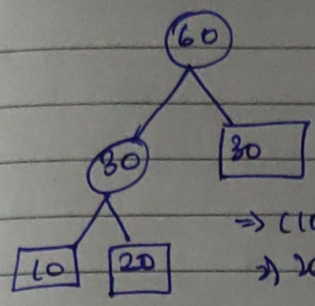
Given 'n' sorted files find an optimal way to pairwise merge them into one sorted file

l_1 l_2 l_3
(30, 20, 10)

l_1 l_2	l_1 l_3 l_2	l_2, l_1, l_3
→ 30 + 20 = 50	30 + 10 = 40	20 + 30 = 50
10 + 50 = 60	40 + 20 = 60	20 + 10 = 60
100		110

l_3 l_2 l_1	l_2 l_3 l_1	l_3 l_1 l_2
10 + 20 = 30	20 + 10 = 30	10 + 30 = 40
30 + 30 = 60	30 + 30 = 60	40 + 20 = 60
90	90	100

(30, 20, 10) ⇒ (10, 20, 30)



⇒ 60 + 30 = 90

⇒ (10×2) + (20×2) + (30×1)

⇒ 20 + 40 + 30 = 90

Ex:-

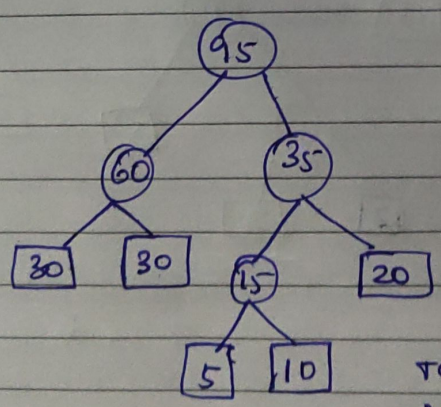
20, 30, 10, 5, 30

5, 10, 20, 30, 30
15

15, 20, 30, 30
35

30, 30, 35
60

35, 60
95



Total no. of nodes = 95 + 60 + 35 + 15

⇒ 205

2. 2, 3, 5, 7, 9, 13

2, 3, 5, 7, 9, 13
5

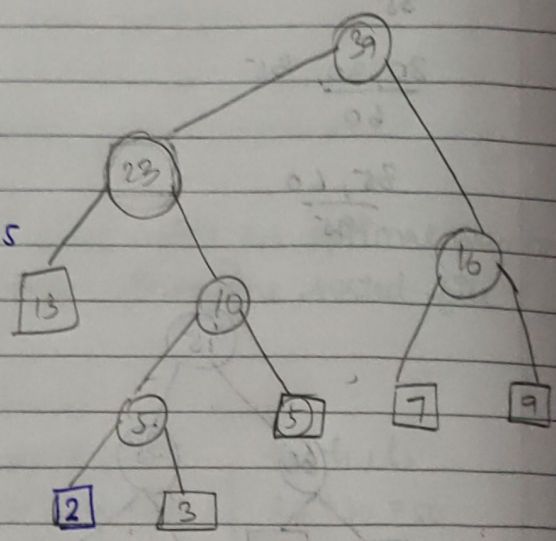
5, 5, 7, 9, 13
10

7, 9, 10, 13
16

10, 13, 16
23

16, 23
39

Total no. of nodes
⇒ 39+23+16+10+5
⇒ 93



→ algorithm:-

```
Algorithm struct treenode {
        Struct treenode *lchild, *rchild;
        int weight;
    }
for i=1 to n-1 do
    declare newnode
node *leftchild = least(list)
```

```
node.right.child = dequeue(list)
node.weight = ((node.left.child).weight) +
                ((node.right.child).weight))
enqueue(list, node);
return dequeue list;
```

Ex:-

30, 30, 10, 5, 30

5, 10, 10, 30, 30

15

15, 10, 30, 30

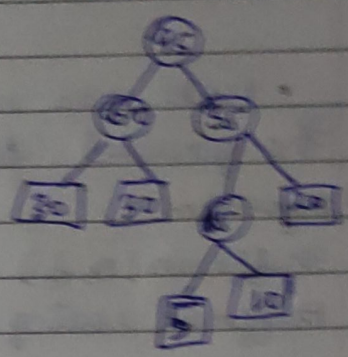
35

30, 30, 35

60

35, 60

95



⇒ Optimal storage on tapes:-

l_1 l_2 l_3
(5, 10, 3)

1 ⇒ $l_1 + l_2 + l_3 = 5 + (15) + 18 = 38$
(l_1, l_1+l_2, $l_1+l_2+l_3$)

2 ⇒ (5, 3, 10) ⇒ $l_1 + l_3 + l_2$ ⇒ 5 + 8 + 18 ⇒ 31

3 ⇒ (10, 5, 3) ⇒ $l_2 + l_1 + l_3$ ⇒ 10 + (15) + 18 = 43

4 ⇒ (10, 3, 5) ⇒ $l_2 + l_3 + l_1$ ⇒ 10 + 13 + 18 = 41

5 ⇒ (3, 5, 10) ⇒ $l_3 + l_1 + l_2$ ⇒ 3 + 8 + 18 ⇒ 29

6 ⇒ (3, 10, 5) ⇒ $l_3 + l_2 + l_1$ ⇒ 3 + 13 + 18 = 34

Page No.:
Date: / /

Huffman code:-

- It is used for lossless data compression

- Assign variable length code to input characters, length of the assigned course are based on the frequencies of the corresponding characters.

- Make sure no ambiguity when decoding the generated bit stream.

Algorithm:-

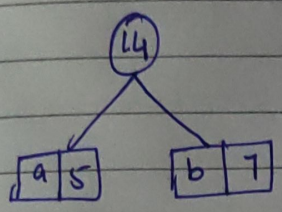
Huffman (n, f)

Huffman coding input; Array of numerical frequencies or probabilities

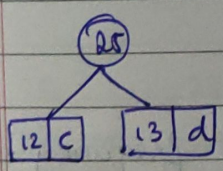
output: binary coding tree with 'n' leaves that has minimum expected code length

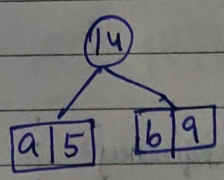
for i = 1 to n do
H[i] = i, f(i)
Create a leafnode labelled i
Build heap (H)
for i = n+1 to 2n-1 do
x = Extract Min(H);
y = Extract Min (H);
Create a node labelled i with children the nodes labelled x. label and y label
Insert H(i, x. freq + y. freq)

character	frequency
a	5
b	9
c	12
d	13
e	16
f	45

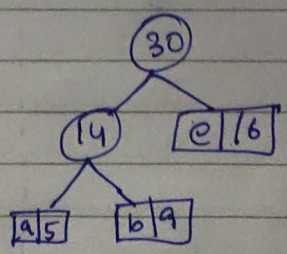


c	12
d	13
Internal Node	14
e	16
f	45

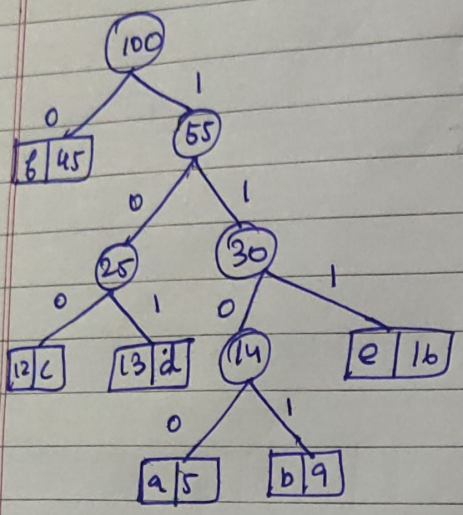




Internal node	14
e	16
Internal Node	25
f	45



Internal Node 25
Internal Node 30
f 45



f 45
Internal node 55

a → 1100
b → 1101
c → 100
d → 101
e → 111
f → 0

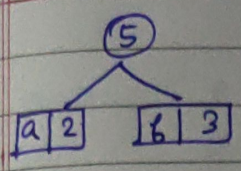
Character	frequency
a	2
b	7
c	23
d	45
e	17
f	3

Character	frequency
a	2
f	3
b	7
e	17
c	23
d	45

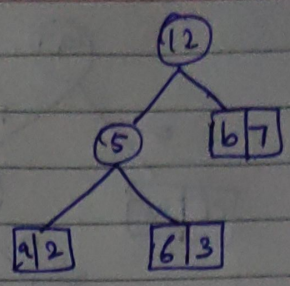
TNPL

Page No.

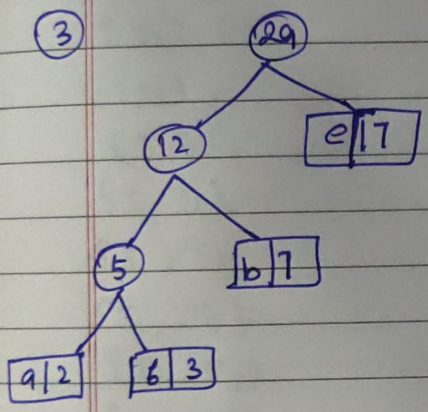
Date / /

① 

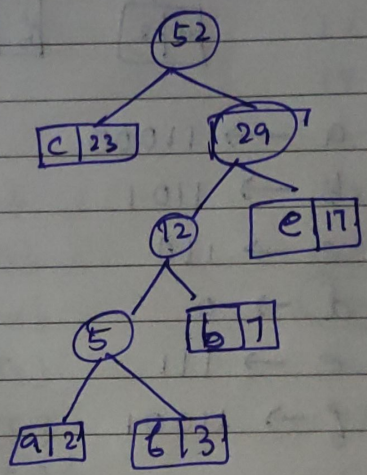
Internal Node	5
b	7
e	17
c	23
d	45

② 

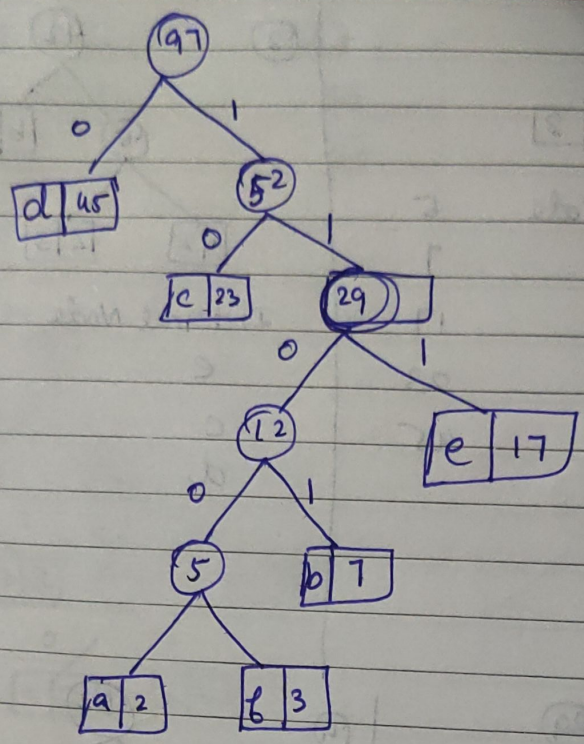
Internal Node	12
e	17
c	23
d	45

③ 

c	23
Internal node	29
d	45

④ 

d	45
Internal node	52



a → 11000
b → 1101
c → 110
d → 0
e → 111
f → 11001

⇒ <u>Job sequencing problem</u> :-

~~There~~ There are 'n' jobs to be processed on machine each job 'i' has a dead line $d_i \geq 0$ and profit $P_i \geq 0$. P_i is earned if and only if the job is completed by its dead line.

J_1	J_2	J_3	J_4
20	30	10	5
2	1	3	2

1. $n=4$, $P = (70, 12, 18, 35)$
$d = (2, 1, 2, 1)$

	J_1	J_2	J_3	J_4
$P =$	70	12	18	35
$d =$	2	1	2	1

Jibs	Feasible solution	profit
$\{J_1\}$	J_1	70
$\{J_2\}$	J_2	12
$\{J_3\}$	J_3	18
$\{J_4\}$	J_4	35
$\{J_1, J_2\}$	$\{J_1, J_2\}$	82
$\{J_1, J_3\}$	$\{J_1, J_3\}$	88
$\{J_1, J_4\}$	$\{J_1, J_4\}$	105
$\{J_2, J_3\}$	$\{J_2, J_3\}$	30
$\{J_2, J_4\}$	$\{J_2, J_4\}$	47
$\{J_3, J_4\}$	$\{J_3, J_4\}$	53

Optimal soln = 105 = $\{J_1, J_4\}$

To cross verify

105

$0 \frac{35}{J_4} 1 \frac{70}{J_1} 2$

d. $P = (20, 15, 10, 5, 1)$

$d = (2, 2, 1, 3, 3)$

	J_1	J_2	J_3	J_4	J_5
$P =$	20	15	10	5	1
$d =$	2	2	1	3	3

Jobs	Feasible soln	Profit
$\{J_1\}$	J_1	20
$\{J_2\}$	J_2	15
$\{J_3\}$	J_3	10
$\{J_4\}$	J_4	5
$\{J_5\}$	J_5	1
$\{J_1, J_2\}$	$\{J_1, J_2\}$	35
$\{J_1, J_3\}$	$\{J_1, J_3\}$	30
$\{J_1, J_4\}$	$\{J_1, J_4\}$	25
$\{J_1, J_5\}$	$\{J_1, J_5\}$	21
$\{J_2, J_3\}$	$\{J_2, J_3\}$	25
$\{J_2, J_4\}$	$\{J_2, J_4\}$	20
$\{J_2, J_5\}$	$\{J_2, J_5\}$	16
$\{J_3, J_4\}$	$\{J_3, J_4\}$	15
$\{J_3, J_5\}$	$\{J_3, J_5\}$	11
$\{J_4, J_5\}$	$\{J_4, J_5\}$	6
$\{J_1, J_2, J_3\}$	$\{J_1, J_2, J_3\}$	45
$\{J_1, J_2, J_4\}$	$\{J_1, J_2, J_4\}$	40
~~$\{J_1, J_2, J_4\}$~~	~~$\{J_1, J_2, J_4\}$~~	
$\{J_1, J_2, J_5\}$	$\{J_1, J_2, J_5\}$	36
$\{J_1, J_3, J_4\}$	$\{J_1, J_3, J_4\}$	35
$\{J_1, J_3, J_5\}$	$\{J_1, J_3, J_5\}$	31
$\{J_1, J_4, J_5\}$	$\{J_1, J_4, J_5\}$	26
$\{J_2, J_3, J_4\}$	$\{J_2, J_3, J_4\}$	30
$\{J_2, J_3, J_5\}$	$\{J_2, J_3, J_5\}$	26

$\{J_3, J_4, J_5\}$ $\{J_3, J_4, J_5\}$ 16

Optimal solution = 45 = $\{J_1, J_2, J_3\}$

⇒ Optimal binary search Tree (OBST):-

$w[i, j] = q[i]$

$c[i, i] = 0$, $r[i, i] = 0$,

$w[i, j] = p(j) + q(j) + w[i, j-1]$

$$c[i, j] = \min_{i < k \le j} \{c[i, k-1] + c[k, j]\} + w[i, j]$$

$r[i, j]$ = min. Value of k

1. keys = {10, 20, 30, 40}

$p_i = \{\underset{1}{3}, \underset{2}{3}, \underset{3}{1}, \underset{4}{1}\}$

$q_i = \{\underset{0}{2}, \underset{1}{3}, \underset{2}{1}, \underset{3}{1}, \underset{4}{1}\}$

W = weight , c = cost , r = root

$W_{00} = 2$ $C_{00} = 0$ $r_{00} = 0$	$W_{11} = 3$ $C_{11} = 0$ r_{11} 0	$W_{22} = 1$ $C_{22} = 0$ $r_{22} = 0$	$W_{33} = 1$ $C_{33} = 0$ $r_{33} = 0$	$W_{44} = 1$ $C_{44} = 0$ $r_{44} = 0$
$W_{01} = 8$ $C_{01} = 8$ $r_{01} = 1$	$W_{12} = 7$ $C_{12} = 7$ $r_{12} = 2$	$W_{23} = 3$ $C_{23} = 3$ $r_{23} = 3$	$W_{34} = 3$ $C_{34} = 3$ $r_{34} = 4$	
$W_{02} = 12$ $C_{02} = 19$ $r_{02} = 1$	$W_{13} = 9$ $C_{13} = 12$ $r_{13} = 2$	$W_{24} = 5$ $C_{24} = 8$ $r_{24} = 3$		
$W_{03} = 14$ $C_{03} = 25$ $r_{03} = 2$	$W_{14} = 11$ $C_{14} = 19$ $r_{14} = 2$			
$W_{04} = 16$ $C_{04} = 32$ $r_{04} = 2$				

keys = {10, 20, 30, 40}

$P_i = \{\underset{1}{3}, \underset{2}{3}, \underset{3}{1}, \underset{4}{1}\}$

$q_i = \{\underset{0}{2}, \underset{1}{3}, \underset{2}{1}, \underset{3}{1}, \underset{4}{1}\}$

2nd row ⇒

~~$W_{01} = q_1$~~

$W[0,1] = P(1) + q(1) + W[0,0]$
i j $= 3 + 3 + 2 \Rightarrow 8$

$W[1,2] = P(2) + q(2) + W[1\ 1]$
i j $= 3 + 1 + 3 \Rightarrow 7$

$W[2,3] = P(3) + q(3) + W[2\ 2]$
i j $= 1 + 1 + 1 \Rightarrow 3$

$W[3,4] = P(4) + q(4) + W[3,3]$
i j $= 1 + 1 + 1 \Rightarrow 3$

~~$C[0,1]$~~ $C[0,1] = \min_{0 < k \le 1} \{C[0,0] + C[1,1] + W[0,1]\}$
i j

$\boxed{k=1}$

$k = 1 = r$

$\boxed{r=1}$

$\boxed{r=k}$

$= 0 + 0 + 8 \Rightarrow 8$

$$c[1,2] = \min_{0 < k \le 2} \{c[1,1] + c[2,2] + w[1,2]\}$$
$i \quad j$

$\boxed{k=2}$

$= 0 + 0 + 7 \Rightarrow 7$

$$c[2,3] = \min_{2 < k \le 3} \{c[2,2] + c[3,3] + w[2,3]\}$$
$i \quad j$

$\boxed{k=3}$

$= 0 + 0 + 3 \Rightarrow 3$

$$c[3,4] = \min_{3 < k \le 4} \{c[3,3] + c[4,4] + w[3,4]\}$$
$i \quad j$

$\boxed{k=4}$

$= 0 + 0 + 3 \Rightarrow 3$

3rd row $\Rightarrow$

$w[0,2] = p(2) + q(2) + w[0,1]$
$i \quad j$
$= 3 + 1 + 8 \Rightarrow 12$

$w[1,3] = p(3) + q(3) + w[1,2]$
$i \quad j$
$= 1 + 1 + 7 \Rightarrow 9$

$w[2,4] = p(4) + q(4) + w[2,3]$
$i \quad j$
$= 1 + 1 + 3 \Rightarrow 5$

$$c[0,2] = \min_{0 < k \le 2} \{\underbrace{c[0,0] + c[1,2]}_{k=1} + \underbrace{c[0,1] + c[2,2]}_{k=2}\} + w[0,2]$$
$i \quad j$

$\boxed{k=1,2}$

$= \min\{0 + 7 + 8 + 0\} + 12 = 19$ $\boxed{k=1}$

$$c[1,3] = \min_{1 < k \le 3} \{\underbrace{c[1,1] + c[2,3]}_{k=2}, \underbrace{c[1,2] + c[3,3]}_{k=3}\} + w[1,3]$$
i, j

$\boxed{k=2,3}$ $\Rightarrow \min\{0 + 3, 7 + 0\} + 9$

$\Rightarrow 3 + 9 \Rightarrow 12$ $\boxed{k=2}$

$$\underset{i}{c}[\underset{j}{2,4}] = \min_{2<k\le 4} \left\{ \underbrace{c[2,2]+c[3,4]}_{k=3}, \underbrace{c[2,3]+c[4,4]}_{k=4} \right\} + w[2,4]$$

$\boxed{k=3,4}$

$$= \min\{0+3, 3+0\} + 5$$

$\boxed{k=3}$

$$\Rightarrow 8$$

4th row:

$$\underset{i}{w}[\underset{j}{0,3}] = P(3) + q(3) + w[0,2]$$
$$= 1+1+12$$
$$\Rightarrow 14$$

$$\underset{i}{w}[\underset{j}{1,4}] = P(4) + q(4) + w[1,3]$$
$$= 1+1+9$$
$$= 11$$

$$\underset{i}{c}[\underset{j}{0,3}] = \min_{0<k\le 3} \left\{ \underbrace{c[0,0]+c[1,3]}_{k=1}, \underbrace{c[0,1]+c[2,3]}_{k=2}, \underbrace{c[0,2]+c[3,3]}_{k=3} \right\} + w[0,3]$$

$\boxed{k=1,2,3}$

$$\min\{0+12, 8+3, 19+0\} + 14$$

$$= \{12, 11, 19\} + 14$$
$$= 11 + 14$$
$$= 25$$

$\boxed{k=2}$

$$c[1,4] = \min_{1 < k \le 4} \left\{ \underbrace{c[1,1]+c[2,4]}_{k=2}, \underbrace{c[1,2]+c[3,4]}_{k=3}, \underbrace{c[1,3]+c[4,4]}_{k=4} \right\} + w[1,4]$$

$\boxed{k=2,3,4}$

$$= \min\{0+8, 7+3, 12+0\} + 11$$

$$= \min\{8, 10, 12\} + 11 \qquad \boxed{k=2}$$

$$= 19$$

5th row:-

$$w[0,4] = p(4) + q(4) + w[0,3]$$

$$= 1 + 1 + 14$$

$$= 16$$

$$c[0,4] = \min_{0 < k \le 4} \left\{ \underbrace{c[0,0]+c[1,4]}_{k=1}, \underbrace{c[0,1]+c[2,4]}_{k=2}, \underbrace{c[0,2]+c[3,4]}_{k=3}, \underbrace{c[0,3]+c[4,4]}_{k=4} \right\} + w[0,4]$$

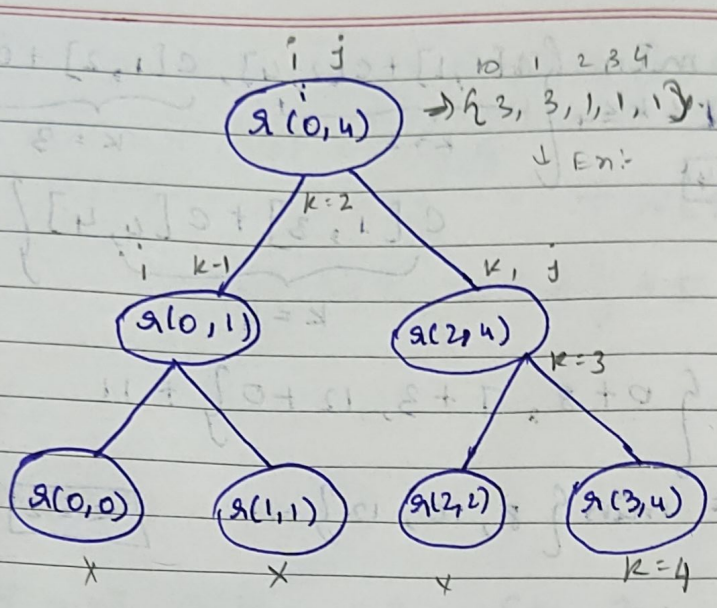
$\boxed{k=1,2,3,4}$

$$= \min\{0+19, 8+8, 19+3, 25+0\} + 16$$

$$= \min\{19, 16, 22+25\} + 16$$

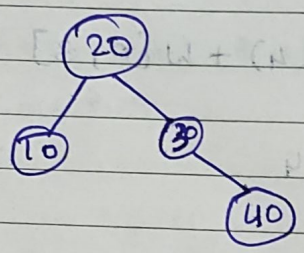
$$= 16 + 16 \qquad \boxed{k=2}$$

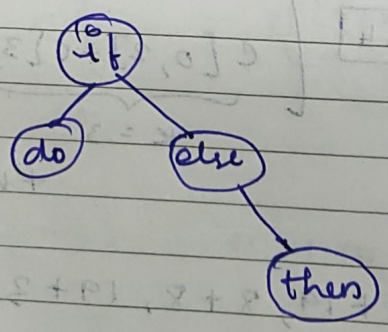
$$= 32$$



↓

1 2 3 4
{10, 20, 30, 40}



↓

1 2 3 4
{do, if, else, then}