Date 03/03/23

Page

* Kruskal Algorithm:-

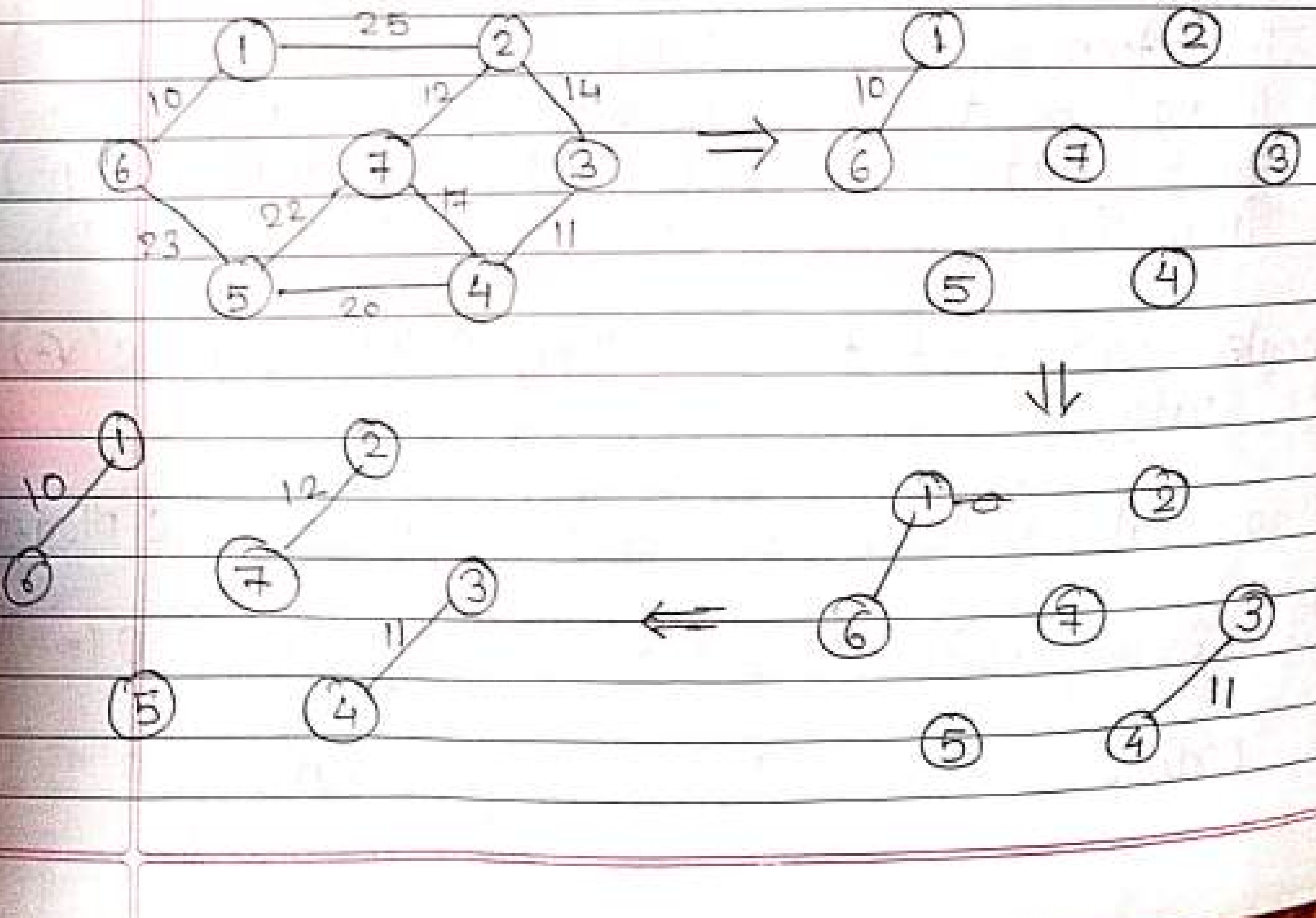
- Select the edges is minimum weight it is not necessary that selected optimum edge is adjacent.

(i) Solve all the edges in decreasing order of there weight.

(ii) Pick the Smallest edge and check if it forms the cycle. if the cycle is not form include this edge else discard it.

(iii) Repeat Step (ii) Until there are V-1 edges in the Spanning tree.

Qo 1] Find the Cost of minimumspanning tree by using Kruskal Algorithm.



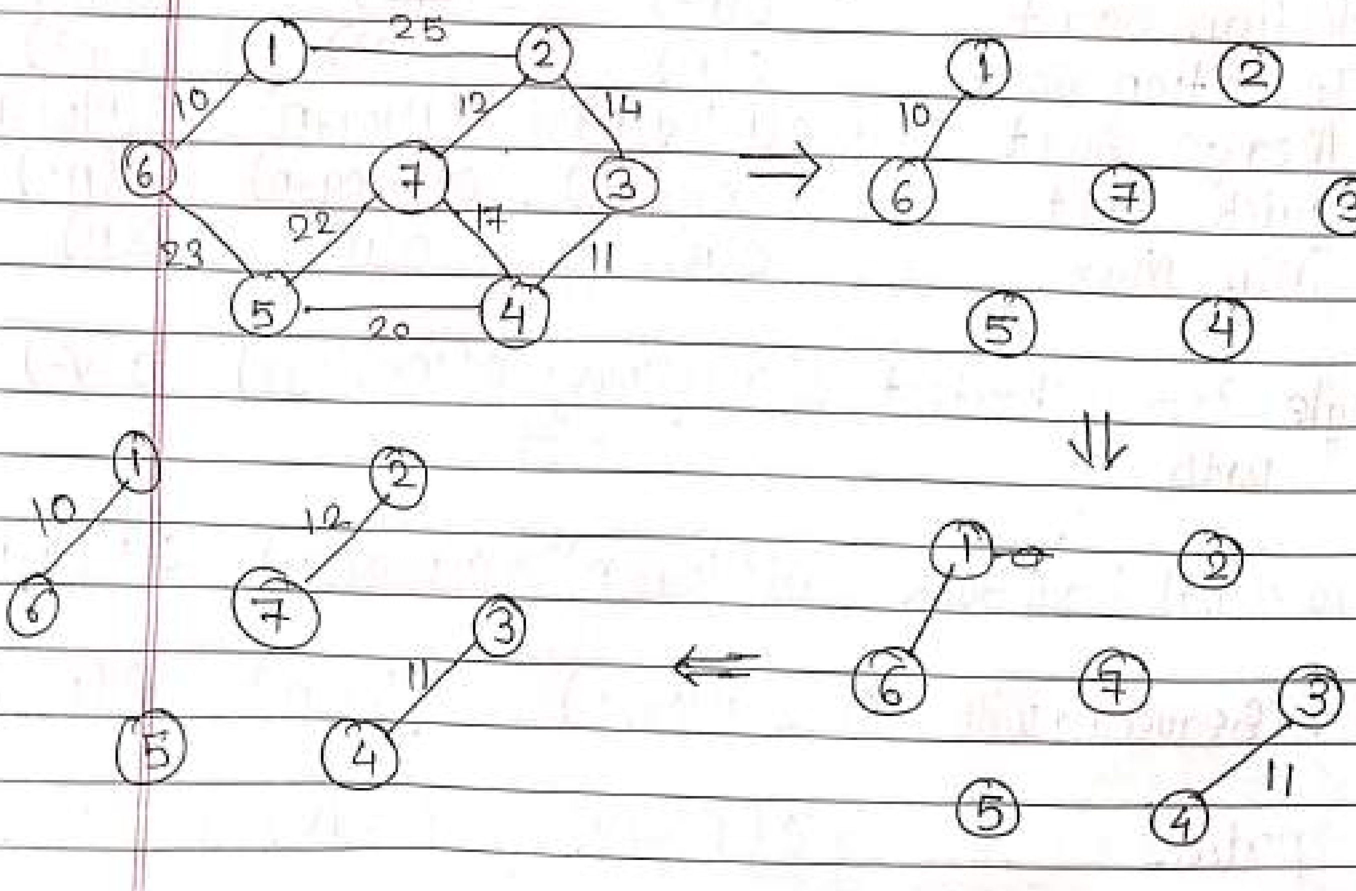
Date 03/03/25
Page

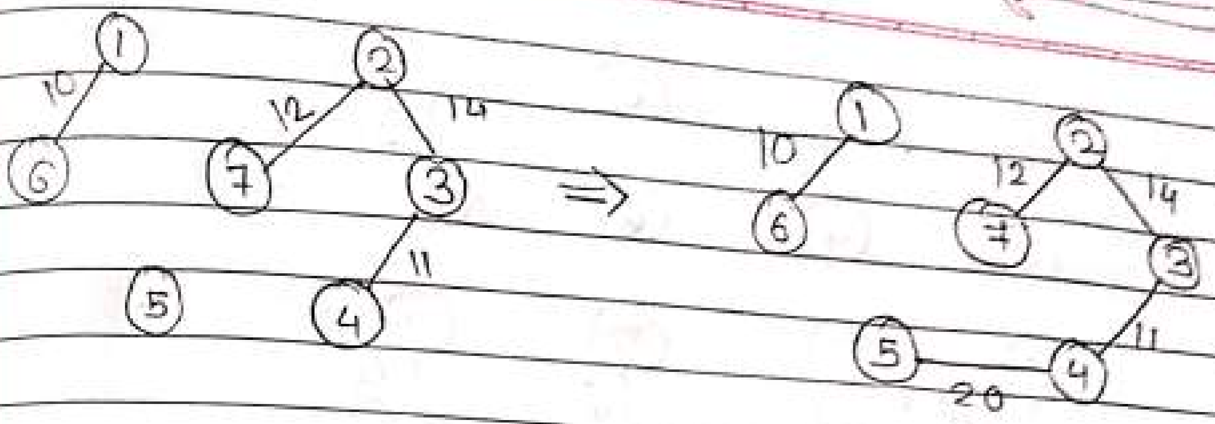
* Kruskal Algorithm:-

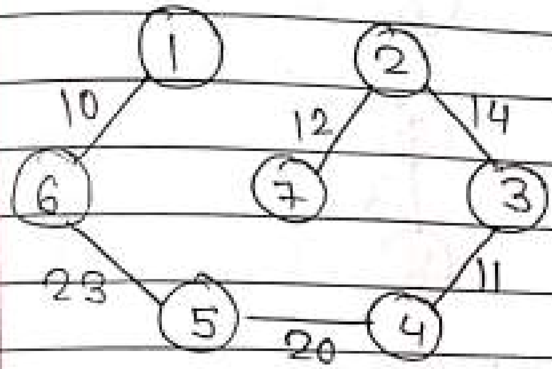
- Select the edges is minimum weight it is not necessary that selected optimum edge is adjacent.

(i) Solve all the edges in decreasing order of there weight.
(ii) Pick the smallest edge and check if it forms the cycle. if the cycle is not form include this edge else discard it.
(iii) Repeat step (ii) until there are V-1 edges in the spanning tree.

Q.1] Find the cost of minimum spanning tree by using kruskal Algorithm.

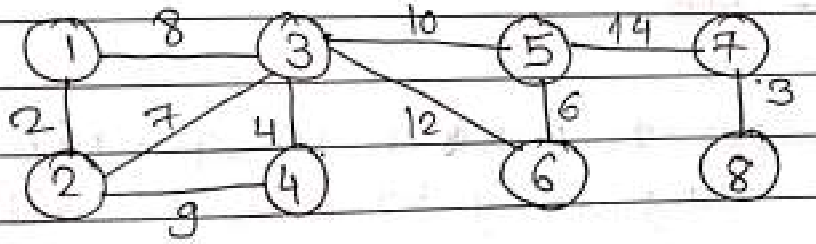


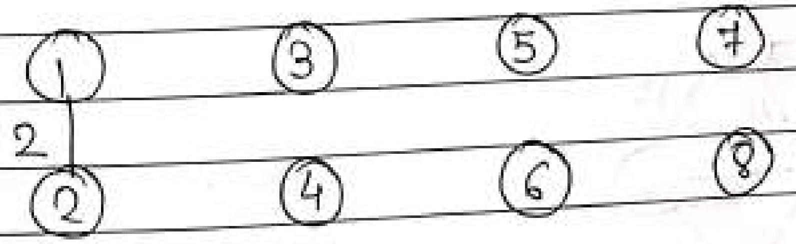


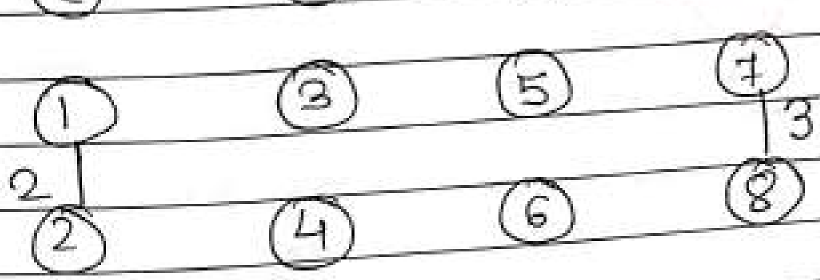


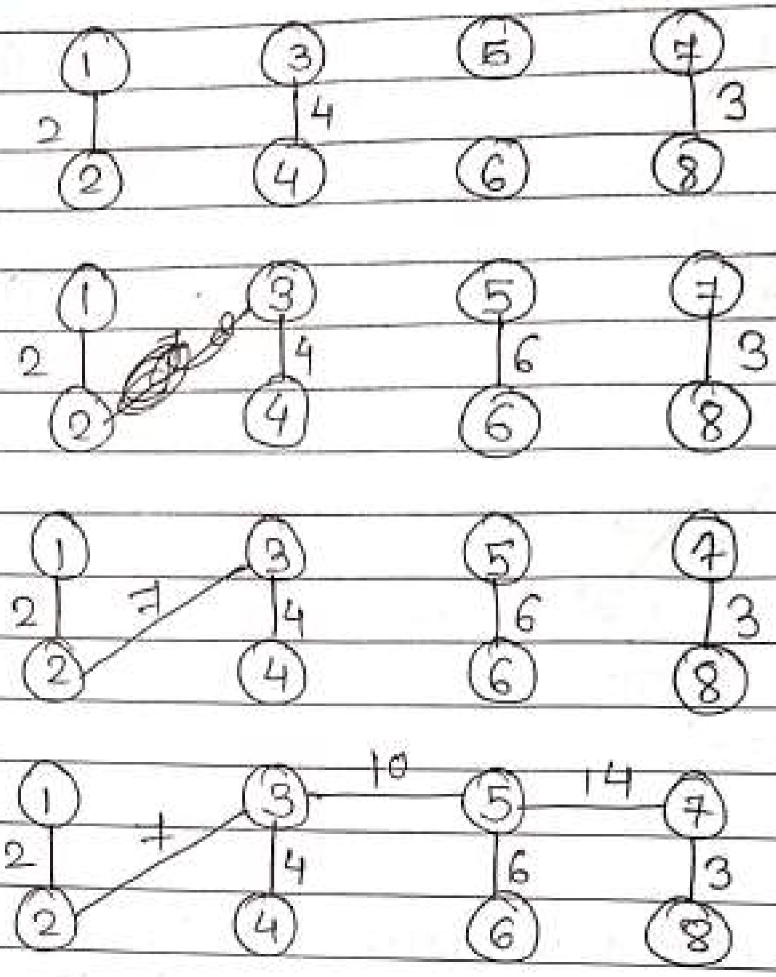
Cost = 90

Q.2] Find the cost of minimum spanning tree for the following graph by using kruskal Algorithm.



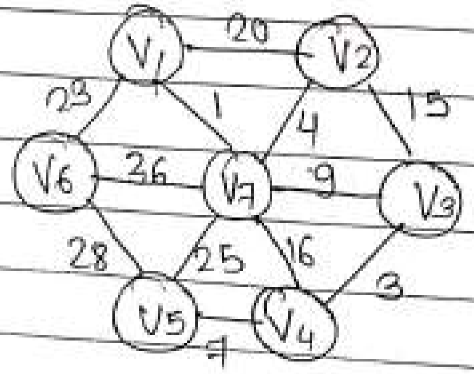


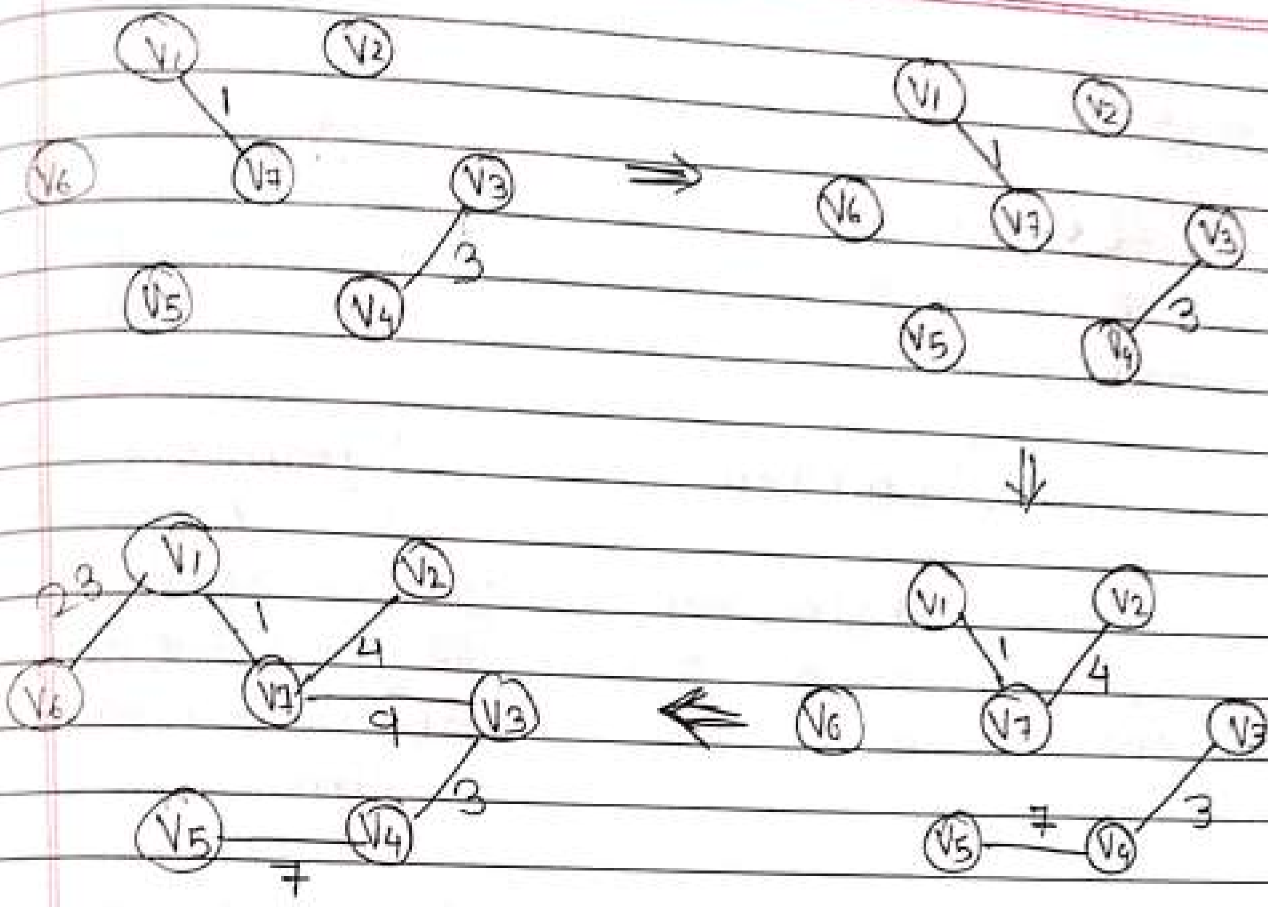




Cost = 46

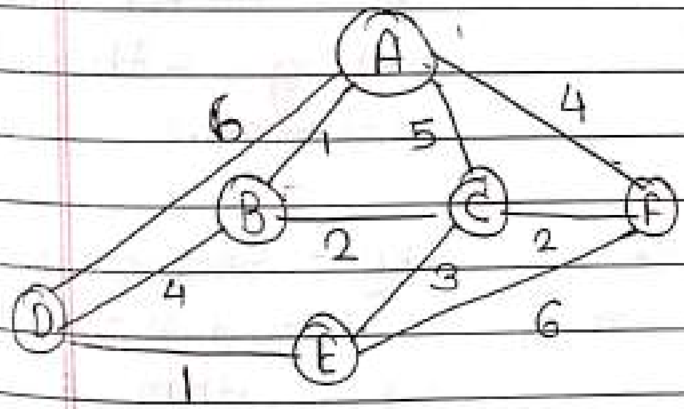
Q. 3] Find the cost of minimum Spanning tree for the following using kruskal Algorithm.

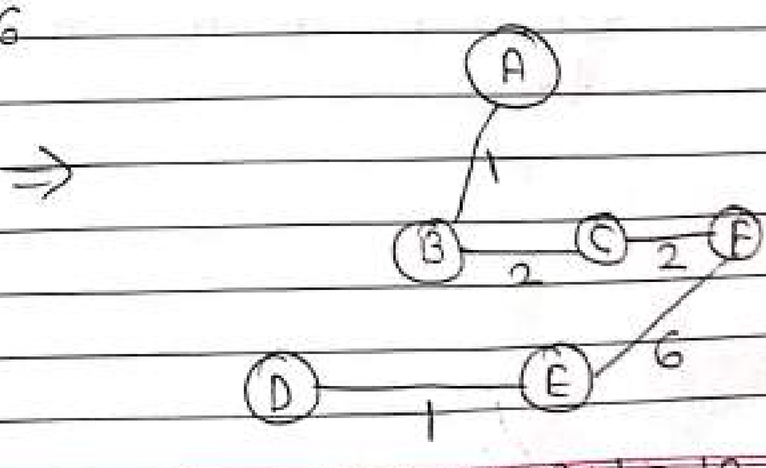




Cost = 47

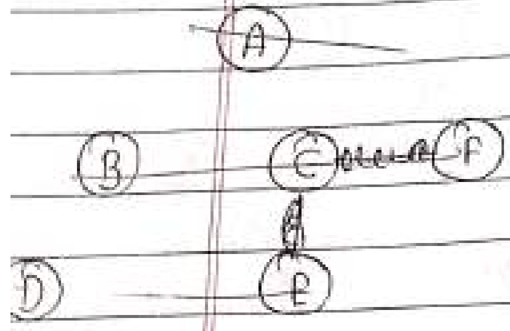
Q04] Find cost of minimum spaning tree for the following tree by using prims.





1+2+2+6+1 = 12

Module - IV



Greedy method	Dynamic programming Apparach
In this method decision is taken best of the inform	It is time varying it gives the information about particular time
There set feasible Solutions & picks up the optimal Solutions.	There is no special set of feasible Solutions.
There is no garenty to get the optimal Solution.	It is garented that DPA gives the optimal Solution.
eg:- fractional knapsack problem, minimum Spanning tree.	eg:- Belman flood, floyd Warshall Algorithm, zero1 knapsack problem. assembly Scheduling problem & longest Common Subsquence.

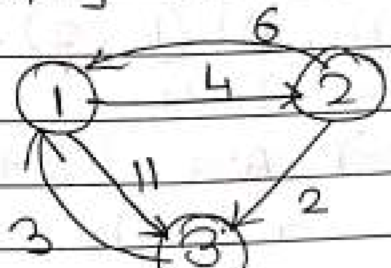
* All pair shortest path OR Floyd Warshall Algorithm :-

- It is a weighted graph is represented by weight matrix the object is used to fine shortest path between every pair of nodes.

All pair shortest path or Floyd warshall Algorithm
Algorithm All - pairs
cost [1---n]. is cost of square matrix of graph with n vertices
A [i, j] is cost of shortest path from vertex i to j

```
cost (i, i) = 0 ,   1 ≤ i ≤ n
  for i = 1 to n do {
   for j = 1 to n do {
    A [i, j] = cos [i, j] }}
  for k = 1 to n do {     // for matrix
  for i = 1 to n do {     // for row
  for j = 1 to n do {     // for column
 A^k (i, j) = min (A^(k-1) (i, j), A^(k-1) (i, k) + A^(k-1) (k, j))
 }
}}
```

Q.1] Apply APSP of the following graph:



$$A^0 = \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 11 \\ 2 & 6 & 0 & 2 \\ 3 & 3 & \infty & 0 \end{array}$$

$k = 1$

$A^1(1,1) = \min\left(A^0(1,1), A^0(1,1) + A^0(1,1)\right)$

$= \min(0, 0+0) = 0$

$A^1(1,2) = \min\left(A^0(1,2), A^0(1,1) + A^0(1,2)\right)$

$= \min(4, 0+4) = 4$

$A^1(1,3) = \min\left(A^0(1,3), A^0(1,1) + A^0(1,3)\right)$

$= \min(11, 0+11) = 11$

$$A^1 = \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 11 \\ 2 & 6 & 6 & 2 \\ 3 & 3 & 7 & 0 \end{array}$$

$A^1(2,1) = \min\left(A^0(2,1), A^0(2,1) + A^0(1,1)\right)$

$= \min(6, 6+0) = 6$

$A^1(2,2) = 0$

$A^1(2,3) = \min\left(A^0(2,3), A^0(2,1) + A^0(1,3)\right)$

$= \min(2, 6+11) = 2$

$A^1(3,1) = \min\left(A^0(3,1), A^0(3,1) + A^0(1,1)\right)$

$= \min(3, 3+0) = 3$

$A^1(3,2) = \min\left(A^0(3,2), A^0(3,1) + A^0(1,2)\right)$

$= \min(\infty, 3+4) = 7$

$A^1(3,3) = 0$

$k = 2$

$A^2(1,2) = \min\left(A^1(1,2), A^1(1,2) + A^1(2,2)\right)$

$= \min(4, 4+0) = 4$

$A^2(1,3) = \min\left(A^1(\ \ ), A^1(\ \ ) + A^1(\ \ )\right)$

$= \min(\ \ ,\ \ +\ \ )$

$A^2(1,1) = 0$

$A^2(2,1) = \min(A^1(2,1), A^1(2,2) + A^1(2,1))$
$= \min(6, 0+6) = 6$
$A^2(2,2) = 0$
$A^2(2,3) = \min(A^1(2,3), A^1(2,2) + A^1(2,3))$
$= \min(2, 0+2) = 2$

$A^2(3,1) = \min(A^1(3,1), A^1(3,2) + A^1(2,1))$
$= \min(3, 7+6) = 3$
$A^2(3,2) = \min(A^1(3,2), A^1(3,2) + A^1(2,2))$
$= \min(7, 7+0) = 7$
$A^2(3,3) = 0$

$$A^2 = \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 6 \\ 2 & 6 & 0 & 2 \\ 3 & 3 & 7 & 0 \end{array} \qquad A^3 = \begin{bmatrix} 0 & 4 & 6 \\ 5 & 0 & 2 \\ 3 & 7 & 0 \end{bmatrix}$$

$k = 3$
$A^3(1,1) = 0$
$A^3(1,2) = \min(A^2(1,2), A^2(1,3) + A^2(3,2))$
$= \min(4, 6+7) = 4$
$A^3(1,3) = 6$

$A^3(2,1) = \min(A^2(2,1), A^2(2,3) + A^2(3,1))$
$= \min(6, 2+3) = 5$
$A^3(2,2) = 0$
$A^3(2,3) = 2$

3rd row remain Same.

* Single Source shortest path & Bellman ford Algorithm

- It is used to find shortest path from Source to destination.
- It is slower than dijkstra's Algorithm.
- It can handle negative weights.
- If the graph conteins negative weight then it is not possible to find shortest path.
- In dijkstra Algorithm vertex is processed that vertex is not processed again & again.
- In Bellman ford Algorithm if the vertex is proccessed, that vertex is processed again & again until obtained we get optimal solution.

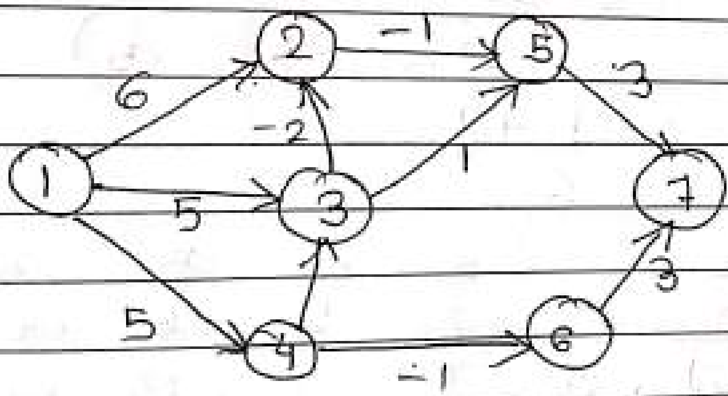
Algorithm Bellman-ford

I/p : Weighted graph.
$w(u, v)$ = weight of edge (u, v)
o/p : Shortest distance of each vertex from given Source vertex

// Initialization
for each $v \in V$ do
  $d[v] = \infty$
  $\pi[v] = nil$
end
  $d[s] \leftarrow 0$; $\pi[s] \leftarrow nil$

for each adjacent node v of u do
Val (v) = dist (u) + wt (u, v)
If Val (v) < dist (v)
dist (v) = Val (v)
π (v) $\leftarrow$ u
end
end
// check for negative Cycle
for each edge (u, v) $\in$ E do
if d [u] + w (u, v) < d [v] then
error " Graph Contains negative cycle"
end
end

Solve the Shortest path from Source 1 to 7



Vertex V	1	2	3	4	5	6	7
dist [V]	0	∞	∞	∞	∞	∞	∞
Parent π [V]	nil	nil	nil	nil	nil	nil	nil

1 $\rightarrow$ Unprocessed Vertex having min distance
adjacent of 1 are 2, 3, 4

$$Val(2) = dist(1) + wt(1,2)$$
$$= 0 + 6 = 6$$
$$Val(2) < dist(2)$$
$$dist(2) = 6$$

$$Val(3) = 5$$
$$Val(4) = 5$$

Vertex V	1	2	3	4	5	6	7
dist[v]	0	6	5	5	∞	∞	∞
π[v]	nil	1	1	1	nil	nil	nil

3 → Unprocessed Vertex having minimum distance

adjacent of 3 = 2, 5

$$Val(2) = dist(3) + wt(3,2)$$
$$= 5 - 2$$
$$= 3$$
$$Val(5) = dist(3) + wt(3,5)$$
$$= 5 + 1$$
$$= 6$$

Vertex V	1	2	3	4	5	6	7
dist[v]	0	3	5	5	6	∞	∞
π[v]	nil	3	1	1	3	nil	nil

2 → Unprocessed Vertex having min distance

adjacent of 2 = 5

$$Val(5) = dist(2) + wt(2,5)$$
$$= 3 - 1 = 2$$

Vertex V	1	2	3	4	5	6	7
dist[V]	0	3	5	5	2	∞	∞
π[V]	nil	3	1	1	2	nil	nil

5 → Unprocessed vertex having minimum distance.

adjacent of 5 = 7

$Val(7) = dist(5) + wt(5,7)$

$= 2+3$

$= 5$

Vertex V	1	2	3	4	5	6	7
dist[V]	0	3	5	5	2	∞	5
π[V]	nil	3	1	1	2	nil	5

4 → Unprocessed vertex having minimum distance

adjacent of 4 = 3, 6

$Val(3) = dist(4) + wt(4,3)$

$= 5-2 = 3$

$Val(6) = dist(4) + wt(4,6)$

$= 5-1 = 4$

Vertex V	1	2	3	4	5	6	7
dist[V]	0	3	3	5	2	4	5
π[V]	nil	3	4	1	2	4	5

3 → unprocessed vertex having minimum distance.

adjacent of 3 = 2, 5

$Val(2) = dist(3) + wt(3,2)$

$= 3-2 = 1$

$Val(5) = dist(3) + wt(3,5)$
$= 3+1 = 4$
$Val(5) > dist(5)$
No change

Vertex V	1	2	3	4	5	6	7
dist[v]	0	1	3	5	2	4	5
π[v]	nil	3	4	1	2	4	5

2 ← Unprocessed Vertex having minimum distance
adjacent of 2 = 5
$Val(5) = dist(2) + wt(2,5)$
$= 1-10$
$= 0$

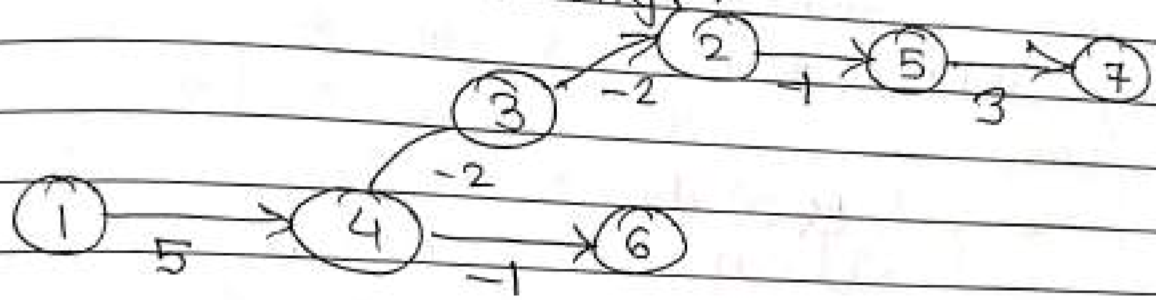
Vertex V	1	2	3	4	5	6	7
dis[V]	0	1	3	5	0	4	5
π[V]	nil	3	4	1	2	4	5

5 ← Unprocessed Vertex
adjacent of 5 = 7
$Val(7) = dist(5) + wt(5,7)$
$= 0+3$
$= 3$

Vertex V	1	2	3	4	5	6	7
dis[v]	0	1	3	5	0	4	3
π[v]	nil	3	4	1	2	4	5

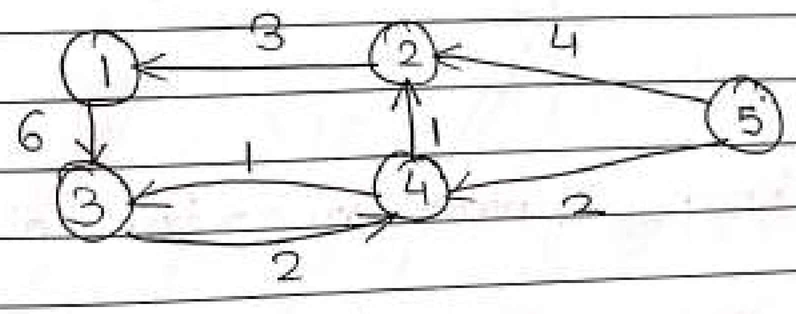
Date ______
Page ______

6 → Unprocessed Vertex
adjacent of 6 = 7
Val(7) = dist(6) + wt(6,7)
= 4 + 3
= 7
No change.



- Base Case :-
  O(E) (No. of edges)
- Worst Case :- (Average Case)
  O(E V) (No. of edges × No. of Vertices)

Q02 Find the shortest path for the source Vetex is 5



* 0/1 knapsack problem :-

Algorithm binary-knapsack
// I/P :- weight, profit, Capacity
// O/P :- Array V which holds Solution of problems

```
for i=0 to n do
    V[i, 0] ← 0
end
for j=0 to M do
    V[0, j] = 0
end
for j=1 to M do
for i=1 to n do
  if w[i] ≤ j
V[i, j] = max {V(i-1, j), Vi + V[i-1, j-w[i]]}
else
  V[i, j] ← V[i-1, j]     // j< w[i]
end
end
end
```

Q01] Apply 0/1 knapsack problem on the following data.

P_i = (10, 12, 15), weight (4, 6, 8), M = 4

→ j

.		j	0	1	2	3	4
P_i	W_i	0	0	0	0	0	0
10	4	1	0	0	0	0	10
12	6	2	0	0	0	0	10
15	8	3	0	0	0	0	10

x_1	x_2	x_3
1	0	0

weight = 4 = M
profit = 10

Fill Ist Column

V [i, j], V [1, 1], i = 1, j = 1, $w_i = 4$ $j < 4$, $1 < 1$

$V[i, j] = V[i-1, j]$

$V[1, 1] = V[0, 1]$

$= 0$

Fill 2nd Column

Fill 3rd Column

Fill 4th Column

V [i, j], V [1, 4], i = 1 j = 4, $w_i = 4$, $j = w_i$

$V(1,4) = \max(V(0,4), 10 + V[0,0])$

$= \max(0, 10 + 0)$

$= 10$

Qo2]

→ P_i (10, 12, 15), weight = (4, 6, 8), M = 10

$j > w_i$ choose the maximum value.

→ j

		i	0	1	2	3	4	5	6	7	8	9	10
P_i	w_i	0	0	0	0	0	0	0^b	0	0	0	0	0
10	4	1	0	0	0	0	10^a	10	10	10	10	10	10
12	6	2	0	0	0	0	10	10	12	12	12	12	22
15	8	3	0	0	0	0	10	10	12	12	15	15	22

x_1	x_2	x_3
1	1	0

$j < w_i$ then choose the previous value.

Weight of 2nd object is = 6

If we consider weight of first object

Then total weight = 6 + 4 = 10 = Capacity

Weight = 4 + 6 = 10

profit = 10 + 12 = 22

Q.3] $P_i (1,4,5,7)$ $W_i = 1,3,4,5$ $M = 7$

		i	0	1	2	3	4	5	6	7
P_i	W_i	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	1	1
4	3	2	0	1	1	4	5	5	5	5
5	4	3	0	1	1	4	5	6	6	9
7	5	4	0	1	1	4	5	7	8	9

x_1 x_2 x_3 x_4
0 1 1 0

$9 - 5 = 4$
$4 - 4 = 0$

weight $= 4 + 3 = 7$
profit $= 5 + 4 = 9$

* Longest Common Subsequence :-

String 1 a b c d e f g h i j

String 2 e c d g i

Common Sequence = c d g i (longest)

String 1 a b c d e f g h i j

String 2 e c d g i

Common Sequence = e g i

Algorithm LCS (x, y)
// x is String of length n
// y is String of length m

```
for i=0 to m do
  LCS [i, 0] = 0
end
for j=0 to n do
  LCS [0, j] = 0
end
for i=1 to m do
for j=1 to n do
  if xi = = yj then
  LCS [i, j] = LCS [i-1, j-1] + 1
  else
  if LCS [i-1, j] ≥ LCS [i, j-1] then
    LCS [i, j] = LCS [i-1, j]
  else
    LCS [i, j] = LCS [i, j-1]
  end
  end
  end
```

Q1] Find Longest Common Subsequence of the following Strings.

String 1: a b c d

String 2: b d

			a	b	c	d
		0	1	2	3	4
	0	0	0	0	0	0
b	1	0	0	1	1	1
d	2	0	0	1	1	2

if Not match then take maximum

if match then Consider daigonal + that element

Common = b d

Page

[Q 2] String 1 : l o n g e s t
String 2 : s t o n e

			l	o	n	g	e	s	t
		0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
s	1	0	0	0	0	0	0	1	1
t	2	0	0	0	0	0	0	1	2
o	3	0	0	1	1	1	1	1	2
n	4	0	0	1	2	2	2	2	2
e	5	0	0	1	2	2	3	3	3

Longest Common Subsequence = one

[Q 3] String 1 : A B C
String 2 : A B C

			A	B	C
		0	1	2	3
	0	0	0	0	0
A	1	0	1	1	1
B	2	0	1	2	2
C	3	0	1	2	3

Longest Common Subsequence = A B C

Date 19/03/25

Q.4] String 1 : A B C
String 2 : A C B

			A	B	C
		0	1	2	3
	0	0	0	0	0
A	1	0	1	1	1
C	2	0	1	1	2
B	3	0	1	2	2

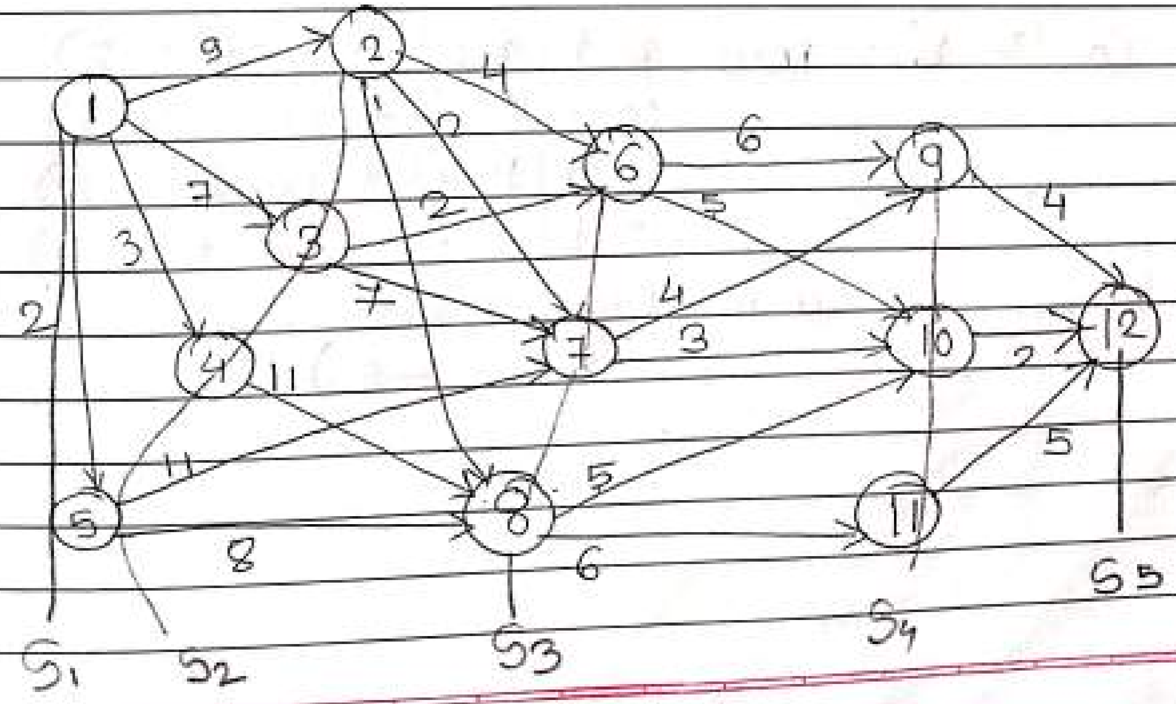
LCS = AC

* Multistage Graph :-
- Multistage graph is used to find
- Minimum cost path from source to destination.
- There are two approaches :-

a) forward Approach
b) Backward Approach.

Q.] Find minimum cost path for the following graph by Using backward approach of multistage graph.

→



Cost (I, 1) = 0
Cost (II, 2) = 9
Cost (II, 3) = 7
Cost (II, 4) = 3
Cost (II, 5) = 2

Cost (III, 6) = 9 (3 - 6)
Cost (III, 7) = 11 (2 - 7)
Cost (III, 8) = 10 (2-8, 5-8)

Cost (III, 6) = min (cost (II, 2) + cost (2, 6),
cost (II, 3) + cost (3, 6),
cost (II, 4) + cost (4, 6),
cost (II, 5) + cost (5, 6))
= min (9 + 4, 7 + 2, 3 + ∞, 2 + ∞)
= 9 (3 - 6)

Cost (III, 7) = min (cost (II, 2) + cost (2, 7),
cost (II, 3) + cost (3, 7),
cost (II, 4) + cost (4, 7)
cost (II, 5) + cost (5, 7))
= min (9 + 2, 7 + 7, 3 + ∞, 2 + 11)
= 11 (2 - 7)

Cos (III, 8) = min (cost (II, 2) + cost (2, 8)
cost (II, 3) + cost (3, 8)
cost (II, 4) + cost (4, 8)
cost (II, 5) + cost (5, 8))
= min (9 + 1, 7 + ∞, 3 + 11, 2 + 8)
= 10 (2 - 8, 5 - 8)

$$\text{cost}(\text{IV}, 9) = \min\big(\text{cost}(\text{III},6) + \text{cost}(6,9),$$
$$\text{cost}(\text{III},7) + \text{cost}(7,9),$$
$$\text{cost}(\text{III},8) + \text{cost}(8,9)\big)$$
$$= \min(9+6,\ 11+4,\ 10+\infty)$$
$$= 15\ (6-9,\ 7-9)$$

$$\text{cost}(\text{IV}, 10) = \min\big(\text{cost}(\text{III},6) + \text{cost}(6,10),$$
$$\text{cost}(\text{III},7) + \text{cost}(7,10),$$
$$\text{cost}(\text{III},8) + \text{cost}(8,10)\big)$$
$$= \min(9+5,\ 11+3,\ 10+5)$$
$$= 14\ (6-10,\ 7-10)$$

$$\text{cost}(\text{IV}, 11) = \min\big(\text{cost}(\text{III},6) + \text{cost}(6,11),$$
$$\text{cost}(\text{III},7) + \text{cost}(7,11),$$
$$\text{cost}(\text{III},8) + \text{cost}(8,11)\big)$$
$$= \min(9+\infty,\ 11+\infty,\ 10+6)$$
$$= 16\ (8-11).$$

$$\text{cost}(\text{V}, 12) = \min\big(\text{cost}(\text{IV},9) + \text{cost}(9,12),$$
$$\text{cost}(\text{IV},10) + \text{cost}(10,12),$$
$$\text{cost}(\text{IV},11) + \text{cost}(11,12)\big)$$
$$= \min(15+4,\ 14+2,\ 16+5)$$
$$= 16\ (10-12)$$

$$12-10-6-3-1\ \big((2+5+2+7) = 16\big)$$
$$12-10-\cancel{7-2}-1\ \big((2+3+2+9) = 16\big)$$

- Algorithm Backward Approach (G, k, n, p)
  cost [1] = 0
  for j = 2 to n
  {
  let r be such that (r, j) is an edge.
  of G and cost [r] + c [r] [j] is min
  cost [j] = cost [r] + c [r] [j]
  d [j] = assign path to d [j]
  }
  for j = k to 1
  {
  Find the overall path
  }

Q.2] String 1: A B C A B E
String 2: A C B A E D

		0	1	2	3	4	5
A	0	A	B	C	A	B	E
A	1						
B	2						
B	3						
A	4						
E	5						
D	6						

Date 20/03/25
Page

* Assembly line Scheduling:-

It is the manufacturing of problem. Manufacturing of large items like Car and truck undergoes through multiple Station.

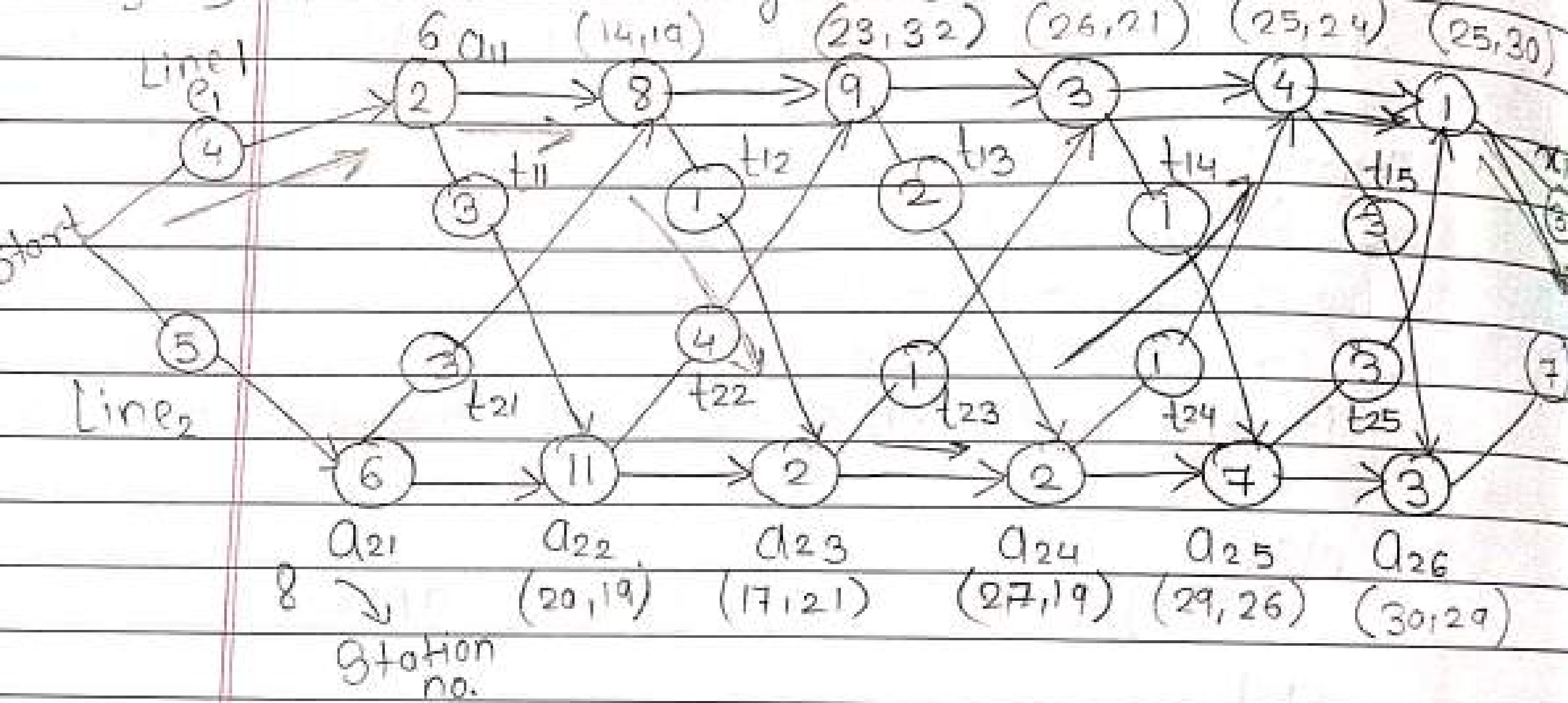
eg:- Manufacturing of Car may be done in several Stages like enginefitting, Colouring, light fitting and so on. If load at Station J on assembly line one very high then components are transperant to Station J on assembly line two.

Algorithm Assembly line Scheduing (n, e, a, t, x)

$f_1[1] = e_1 + a_{11}$
$f_2[1] = e_2 + a_{21}$
for $j = 2$ to n do
if $f_1[j-1] + a_{1j} \leq f_2[j-1] + t_{2,j-1} + a_{2j}$ then
$f_1[j] = f_1[j-1] + a_{1j}$
else
$f_1[j] = f_2[j-1] + t_{2,j-1} + a_{1j}$
end
if $f_2[j-1] + a_{2,j} \leq f_1[j-1] + t_{1,j-1} + a_{2j}$ then
$f_2[j] = f_2[j-1] + a_{2j}$
else
$f_2[j] = f_1[j-1] + t_{1,j-1} + a_{2j}$
end
if $f_1(n) + x_1 \leq f_2(n) + x_2$ then
$f^* = f_1(n) + x_1$
else
$f^* = f_2(n) + x_2$
end
end

- n :- no. of stations = n
- e
- a
- t
- x
- $f_1[j]$

Q.1] Find ALS for given data :-



	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
$f_1[j]$	6	14	23	21	24	25
$f_2[j]$	8	19	17	19	26	29

$$f^* = \min(f_1(n) + x_1, f_2(n) + x_2)$$
$$= \min(25+3, 29+7)$$
$$= 28$$

Date 24/03/25
Page

* N Queens problems:-

2 Queens problem is not soluable

Q	Q

Q	Q

Q	
Q	

	Q
	Q

	Q
Q	

4 Queens problem is Soluable

~~Q~~	Q_1		
		~~Q~~	Q_2
Q_3			
		Q_4	

Place Q_1 on (1,1) — Successful
place Q_2 on (2,1) — fail
Place Q_2 on (2,2) – fail
place Q_2 on (2,3) - Successful
place Q_3 on (3,1) - fail
place Q_3 on (3,2) - fail
place Q_3 on (3,3) - fail
place Q_3 on (3,4) - fail
Algorithm backtracks
and places Q_2 on (2,4)
So place Q_3 on (3,2)

Place Q_4 on (4,1) – fail
Place Q_4 on (4,2) – fail
Place Q_4 on (4,3) – fail
Place Q_4 on (4,4) – fail
Algorithm backtracks and
Places Q_1 on (1,2)

Q.2] 5 Queens problem is Solvable.

Q_1				
		Q_2		
				Q_3
	Q_4			
			Q_5	

	Q_1			
			Q_2	
Q_3				
		Q_4		
				Q_5

- Algorithm N queens problem

```
   for i=1 to n do
        if place (k, i)
then
{
x{k} = 1   // if k^th queen Can be placed on i^th row
 if (k == n) then
      write x[1...n] // if all queens are processed
else                  // then display solution tuple
 Nqueen (k+1, n) // row by row each queen is
                         placed.
                 // by Statisfying Constraint
end if
end if
end for
```

function place (k, i)
{
This function checks two queens can be placed on same column or same diagonal.
then
return false
else
return true
}

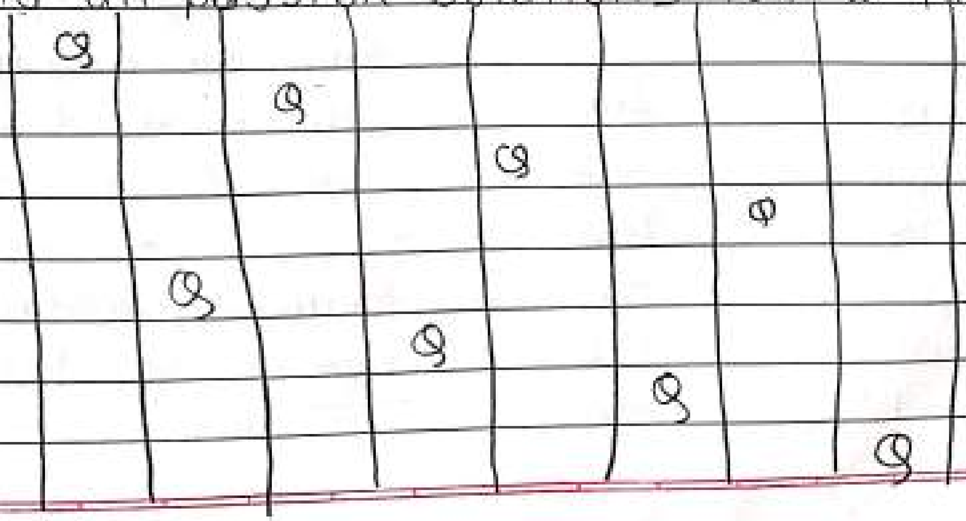
• Complexity :-

N choices to select first queen
N-1 choices to select second queen
N-2 choices to select third queen

$$= O(N * (N-1) * (N-2) ---- 1)$$
$$= O(N!)$$

Best Case
$O(1) \rightarrow$ Solution is obtained immediately.

• Find all possible solutions for 8 queens problem



Page

* Sum of Subset:-

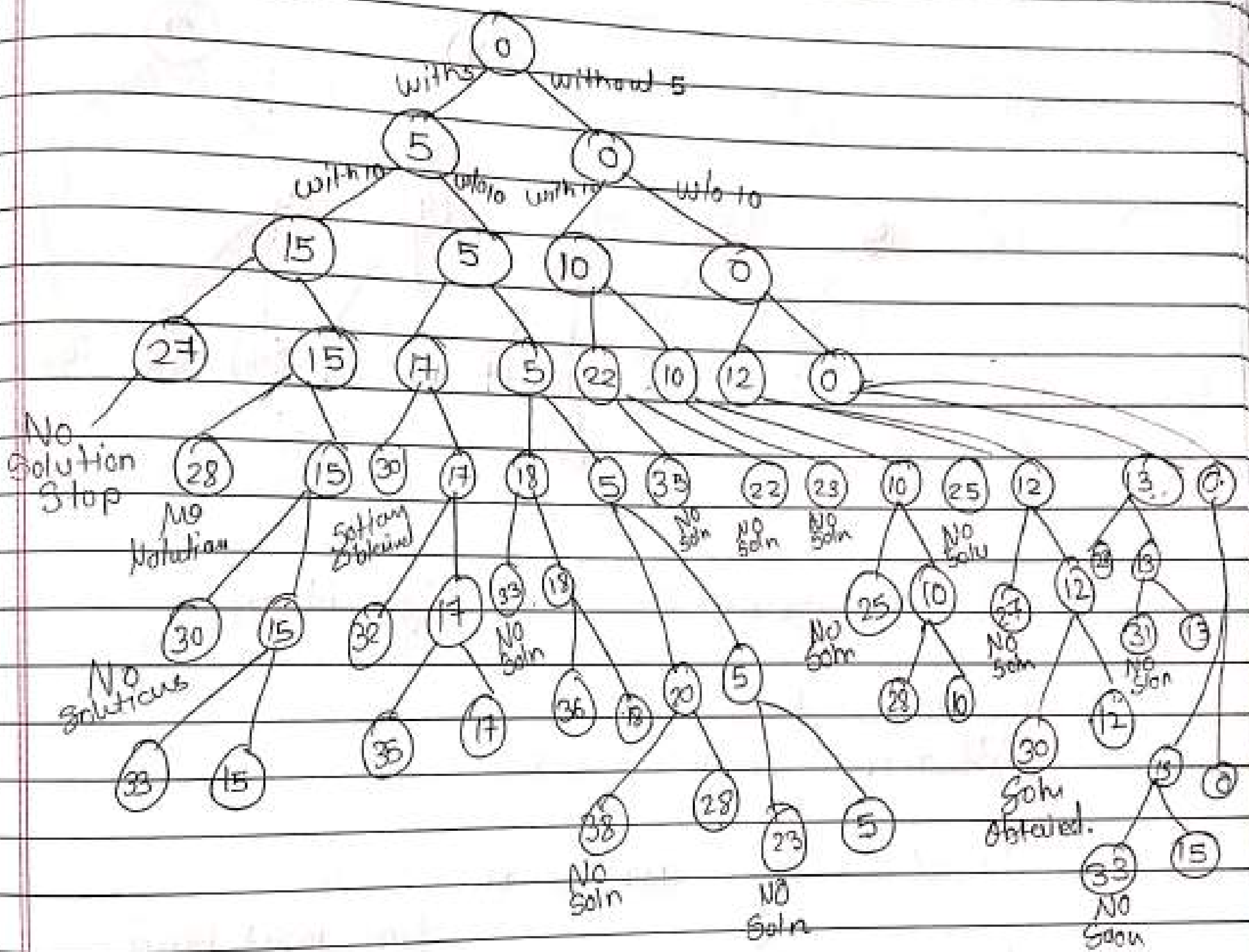
- Start with an empty State.
- Add to the Subset next element from the list.
- If the Subset is having Some D then Stop with a Subset as a Solution.
- If the Subset is not visible or if we have reach the end of the Subset then back track to the Subset Until we find most Suitable Value.
- If the Subset is visible then repeat Step (ii).
- If we have visited all the elements without finding a Suitable Subset and if no back tracking is possible then Stop without Solution.

IMP Q. 1] Consider a Set S = 5, 10, 12, 13, 15, 18 and d = 30 Solve it obtaining Sum of Subset.

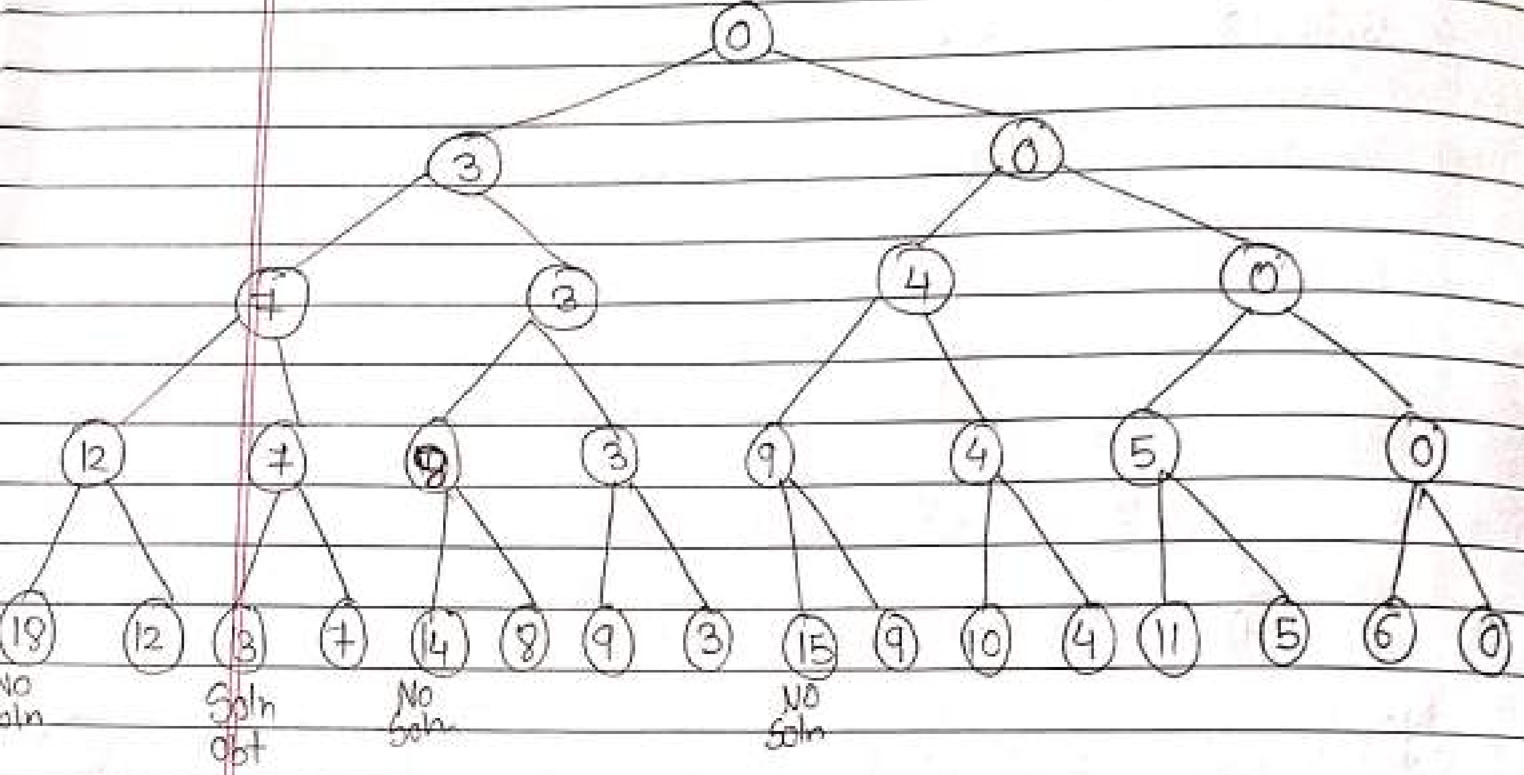
Initial Subset	Sum	Remark
5	5	Add next element
5, 10	15	Sum < 30, add next element
5, 10, 12	27	Sum < 30, add next element
5, 10, 12, 13	40	Sum exceeds d = 30 algo back track
5, 10, 12, 15	42	—— 11 ——
5, 10, 12, 18	45	—— 11 ——
5, 10, 13	28	Sum < 30, add next element
5, 10, 13, 15	43	Sum exceeds d = 30 algo bt.
5, 10, 13, 18	46	—— 11 ——

5,10,15	30	Solution obtained
5,10,15,18	48	Sum exceeds d=30 algo bt
5,10,18	33	—— 〃 ——

Tree diagram



Q02] S = 3, 4, 5, 6 Sum = 13



* Naive String Matching Algorithm :-

Text :- Roman
Patteren :- man

Text : R a m a n

Pattern : m a n
0 1 2

Not matching
Shift to right

R a m a n
m a n

Not matching
Shift to right

0 1 2 3 4
R a m a n
m a n

Match is found at index 2
Return index 2.

Text: raman likes mango
pattern: mango

Match is found at index12
Return index12

- Algorithm Naive (T(1....n), P(1---- m))
  for (s=0 to n-m) do
  {
  if P(1....m) = T(s+1 .... s+m) then
  print ("Match found")
  }
  }

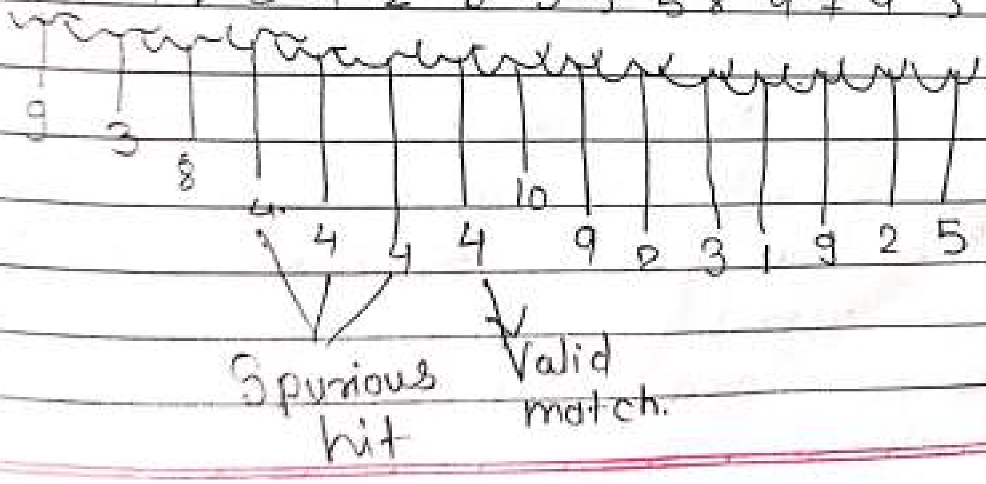
- Complexity :-

Worst case :- O(mn)
Best case :- (O(n))

* Rabin karp Algorithm :-

Text :- 3 1 4 1 5 9 2 6 5 3 5 8 9 7 9 3
pattern :- 26
Modulo q = 11
26 mod 11 = 4 (Remainder)

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



- Algorithm Robin krap:-

```
for (i=1 to n)
  {
    p = p(i) mod q
   tc = T(i) mod q
  }
for (s=0 to n-m)
  {
  if (p=tc) then
  {
  if (p(1···m) = T(s+1 ··· s+m) then
     write ("pattern found")
  }
  }
```

- Complexity:-

Worst Case :- o(mn)

Best Case :- o(m+n)

* knuth morris pratt algorithm:-

The basic idea behind this algorithm is to build prefix array. This array is also called as π array. This prefix array is build Using prefix and Suffix inforamation of pattern. Suffix is word part added to the end of word

prefix is word part added to the beginning of a word.

pattern	a	b	a	d	a	b
prefix array	0	0	1	0	1	2

Consider ab
prefix a
Suffix b

Consider aba
prefix a ab
Suffix a ba

Consider abad
prefix a ab aba
Suffix d ad bad.

Consider abada
prefix a ab aba abad
Suffix a da ada bada

Consider abadab
prefix a ab aba abad abada
Suffix b ab dab adab badab