Page No. :
Date .

Q1 Infix to Postfix & prefix

① (a+b) * c

→ Postfix : ab+c*
→ Prefix : *+abc

② (A + (B×C))

→ Postfix : ABC*+
→ Prefix : +A*BC

③ (X+Y+2) * (A+B-C)

→ Postfix: XY+2 +AB+C-*
→ Prefix: *++XY2 -+ABC

④ ((A/B) - C) + (D * E)

→ Post fix : AB/C-DE*+
→ Prefix : +-/ABC*DE

⑤ (A+B+C/D) * (E-F*G)

→ Post fix: AB+CD/+EFG*-*
→ Prefix: *++AB/CD-E*FG

Page No. :
Date .

⑥ (A/B/C) + (D*E-F)

→ Postfix : AB/C/DE*+F-
→ Prefix : -+//ABC*DEF

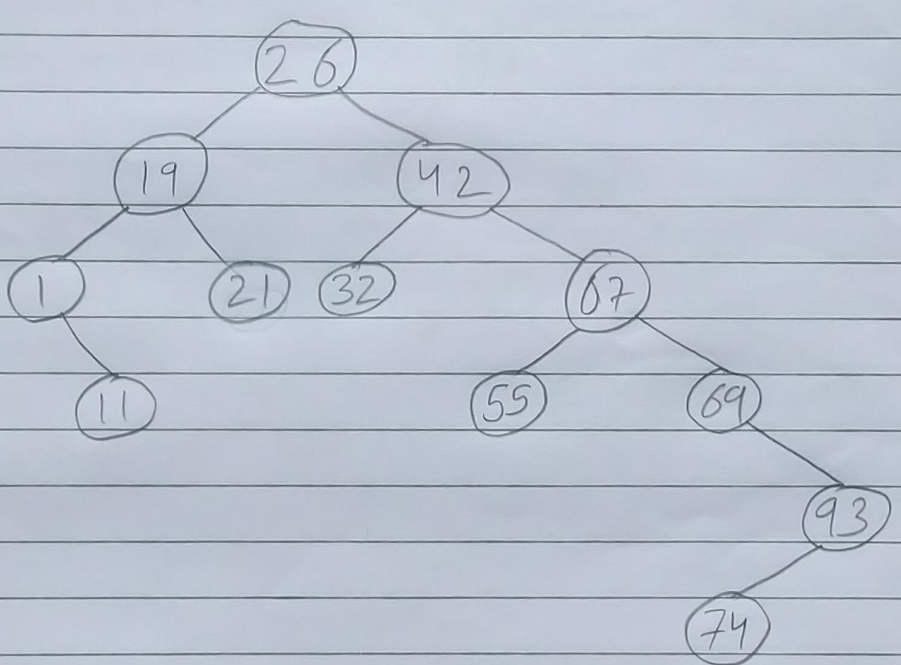
⑦ (x*y*z*w) - (A+B*C-D)

→ Postfix : XYZW***ABC*+D--
→ Prefix : -***XYZW-+A*BC

Page No. :
Date .

Q2 Create binary search tree

① 26, 19, 42, 67, 21, 1, 69, 55, 93, 74, 32, 11



→ Inorder:- 1, 11, 19, 21, 26, 32, 42, 55, 67, 69, 74, 93

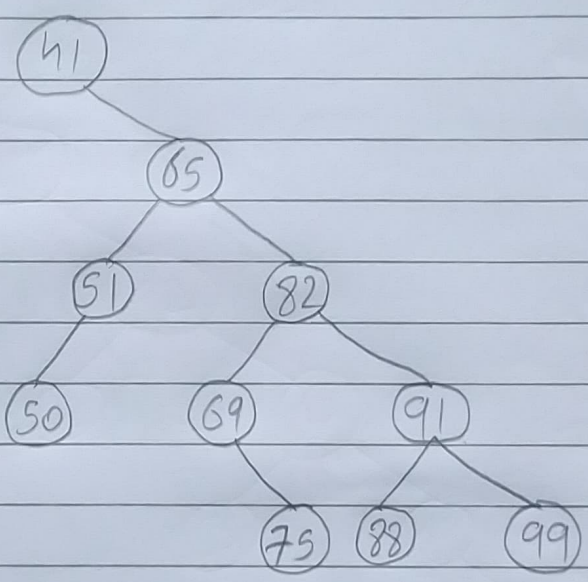
→ Preorder:- 26, 19, 1, 11, 21, 42, 32, 67, 55, 69, 93, 74

→ Postorder:- 11, 1, 21, 19, 32, 55, 74, 93, 69, 67, 42, 26.

Page No. :

Date .

② 41, 65, 82, 91, 51, 50, 69, 75, 88, 99



Inorder :- 41, 50, 51, 65, 69, 75, 82, 88, 91, 99

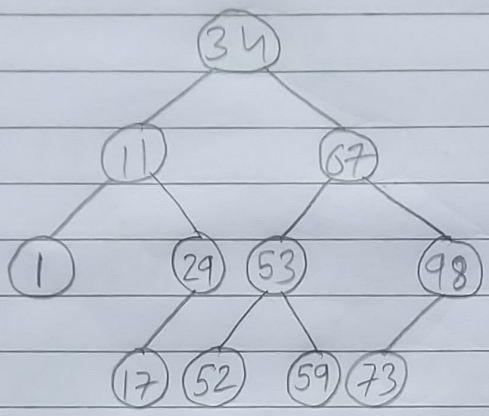
Preorder :- 41, 65, 51, 50, 82, 69, 75, 91, 88, 99

Postorder :- 50, 51, 75, 69, 88, 99, 91, 82, 65, 41

Page No. :

Date .

(3) 34, 11, 67, 98, 1, 73, 53, 29, 52, 17, 59



In order :- 1, 11, 17, 29, 34, 52, 53, 59, 67, 73, 98

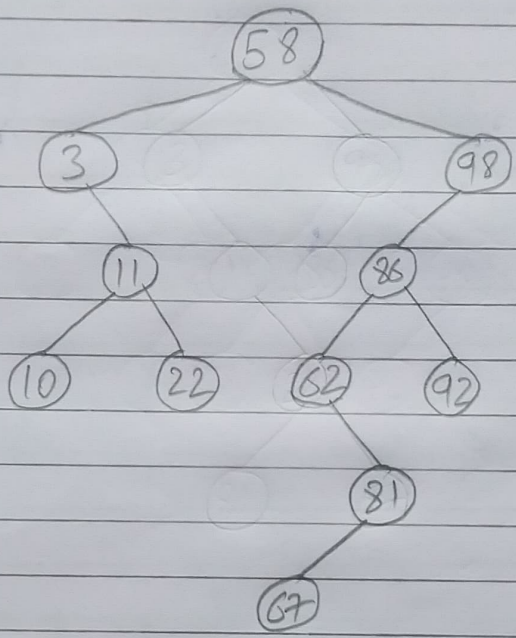
Preorder :- 34, 11, 1, 29, 17, 67, 53, 52, 59, 98, 73

Post order :- 1, 17, 29, 11, 52, 59, 53, 73, 98, 67, 34

Page No. :

Date .

(4) 58, 3, 98, 11, 86, 22, 62, 67, 81, 92, 10



Inorder :- 3, 10, 11, 22, 58, 62, 67, 81, 86, 92, 98

Preorder :- 58, 3, 11, 10, 22, 98, 86, 62, 81, 67, 92

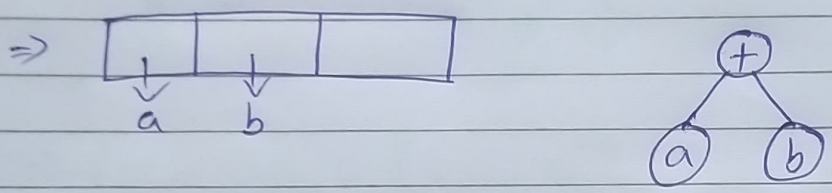
Postorder :- 10, 22, 11, 3, 67, 81, 62, 92, 86, 98, 58

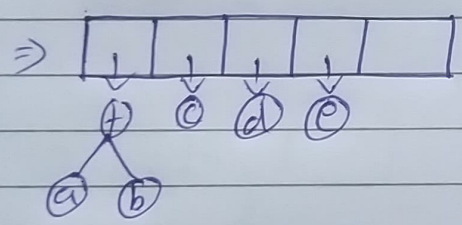
Page No. :

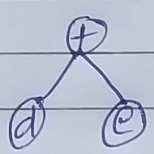
Date .

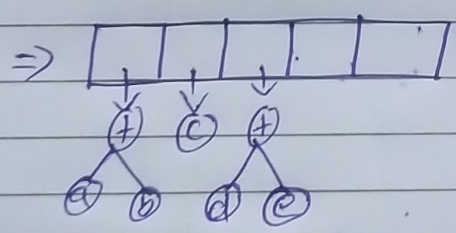
Q3 Create Expression Tree

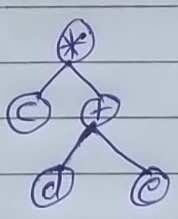
① ab+cde+**

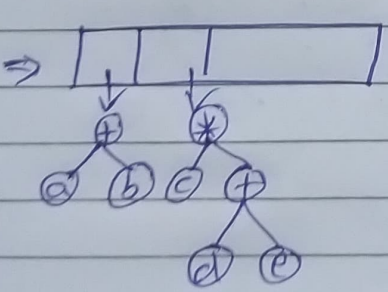


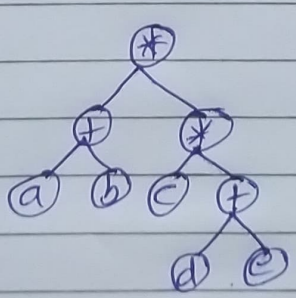








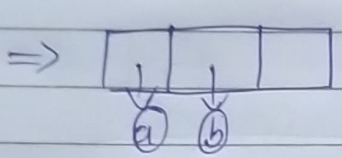
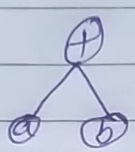


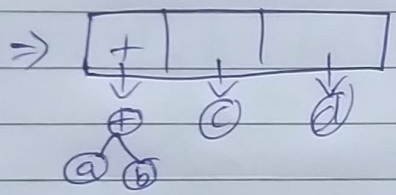
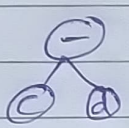


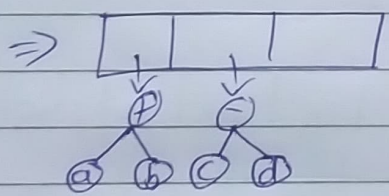
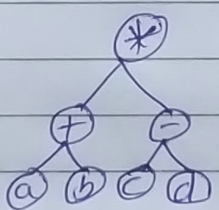
Page No. :

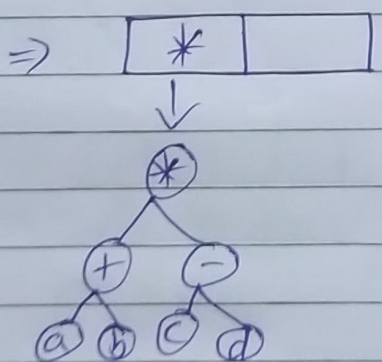
Date .

② $(a+b)*(c-d) \Rightarrow ab+cd-*$

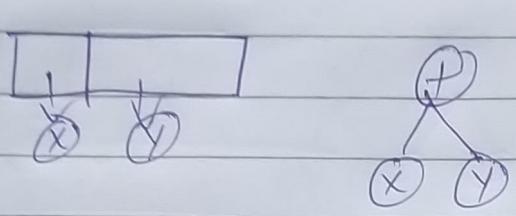


Page No. :

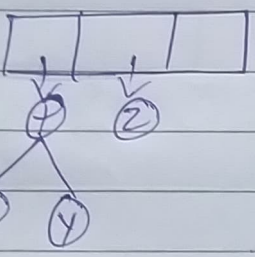
Date .

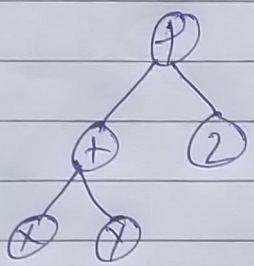
③ X Y + 2 + A B + C − *

⇒

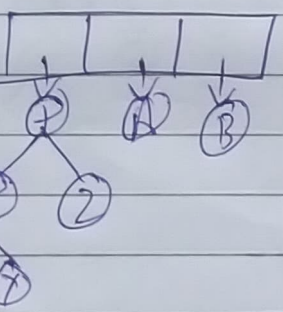


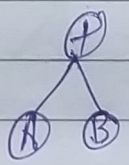
⇒



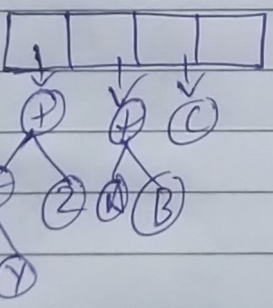


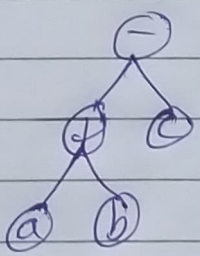
⇒



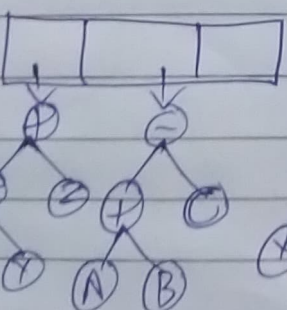


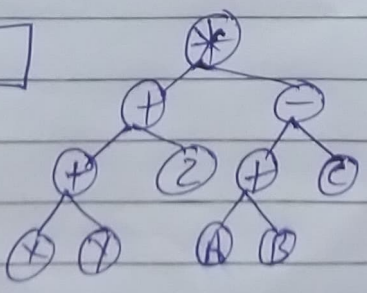
⇒





⇒

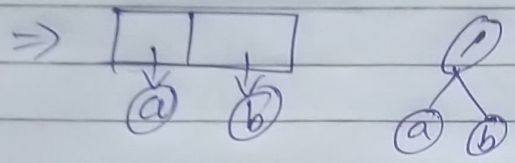


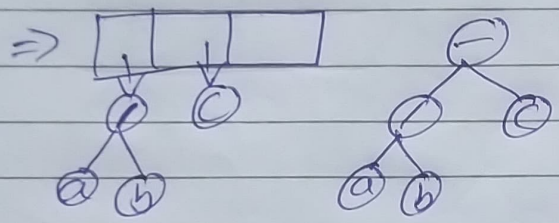


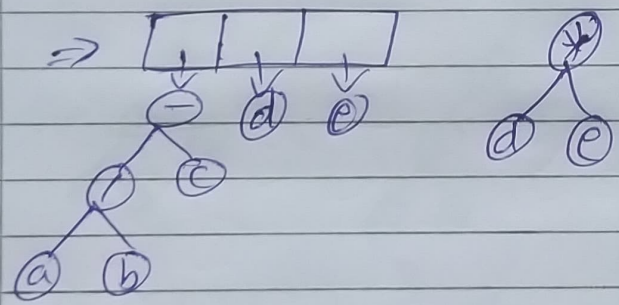
Page No. :

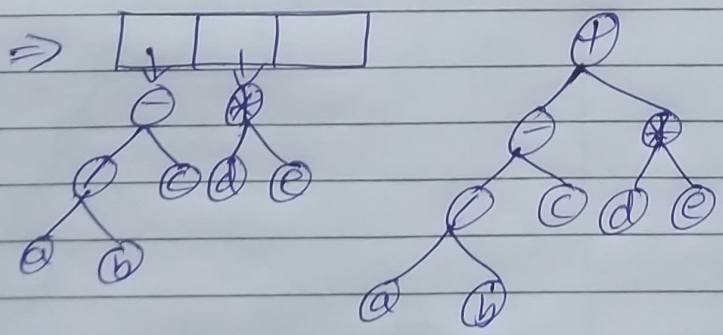
Date .

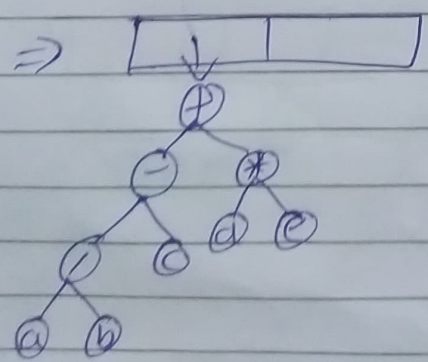
(4) ab/c -de*+







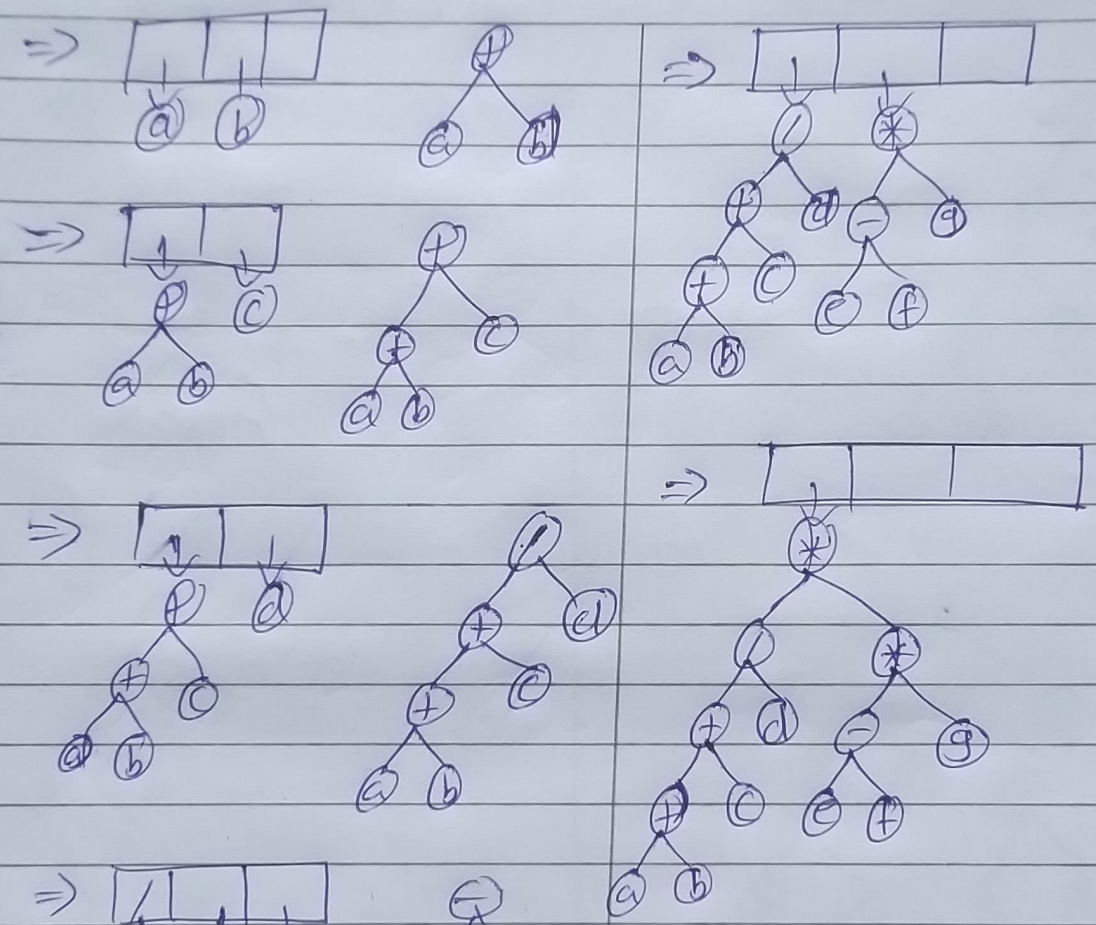




Page No. :

Date .

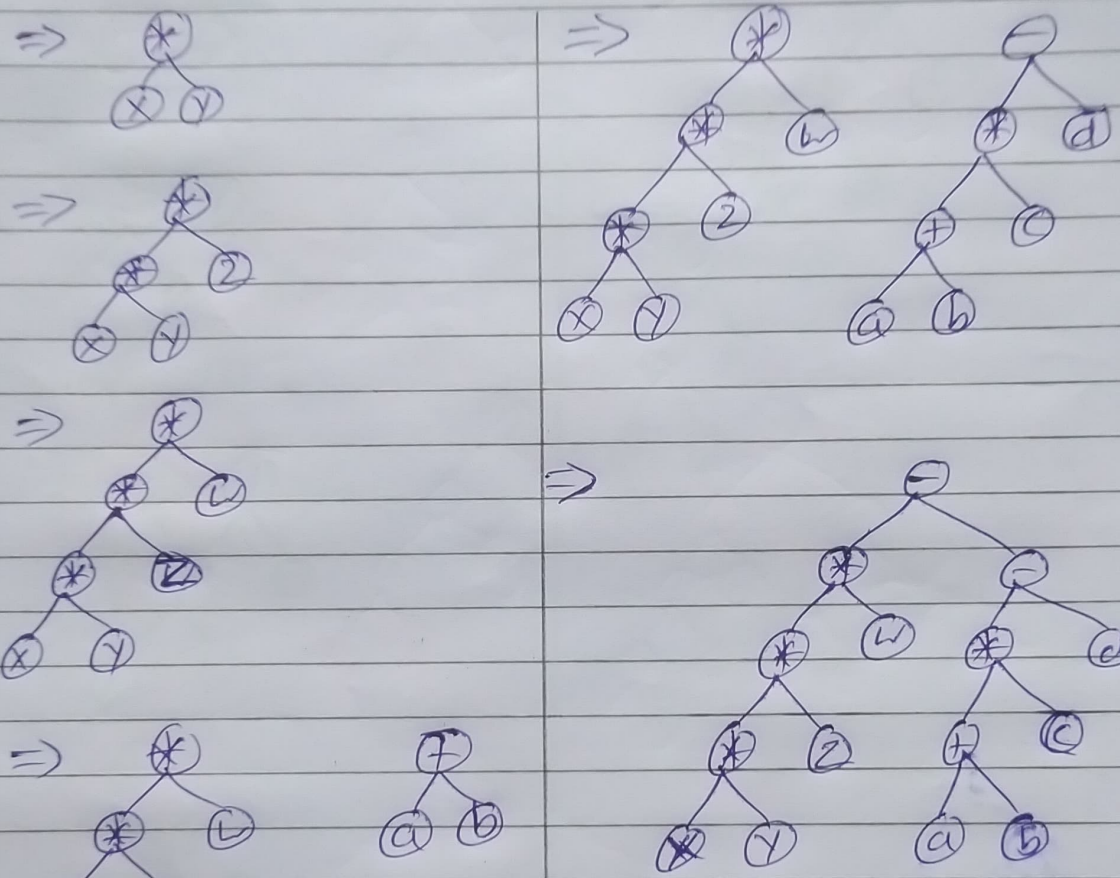
(5) $ab+c+d/ef-g**$



Page No. :

Date .

(6) xy*2*w*ab+c*d--

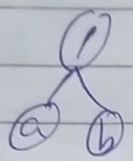


Page No. :

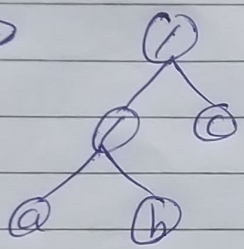
Date .

(7) ab/c/d+e*f-

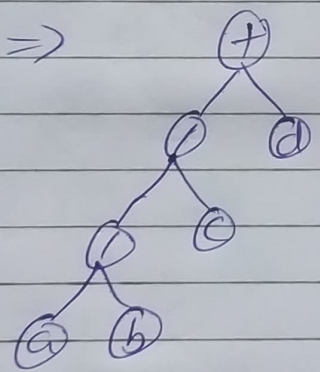
⇒



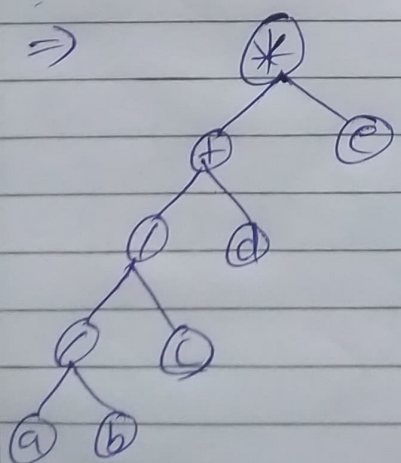
⇒



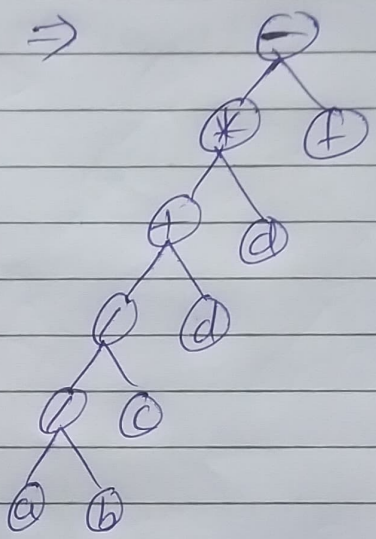
⇒



⇒



⇒

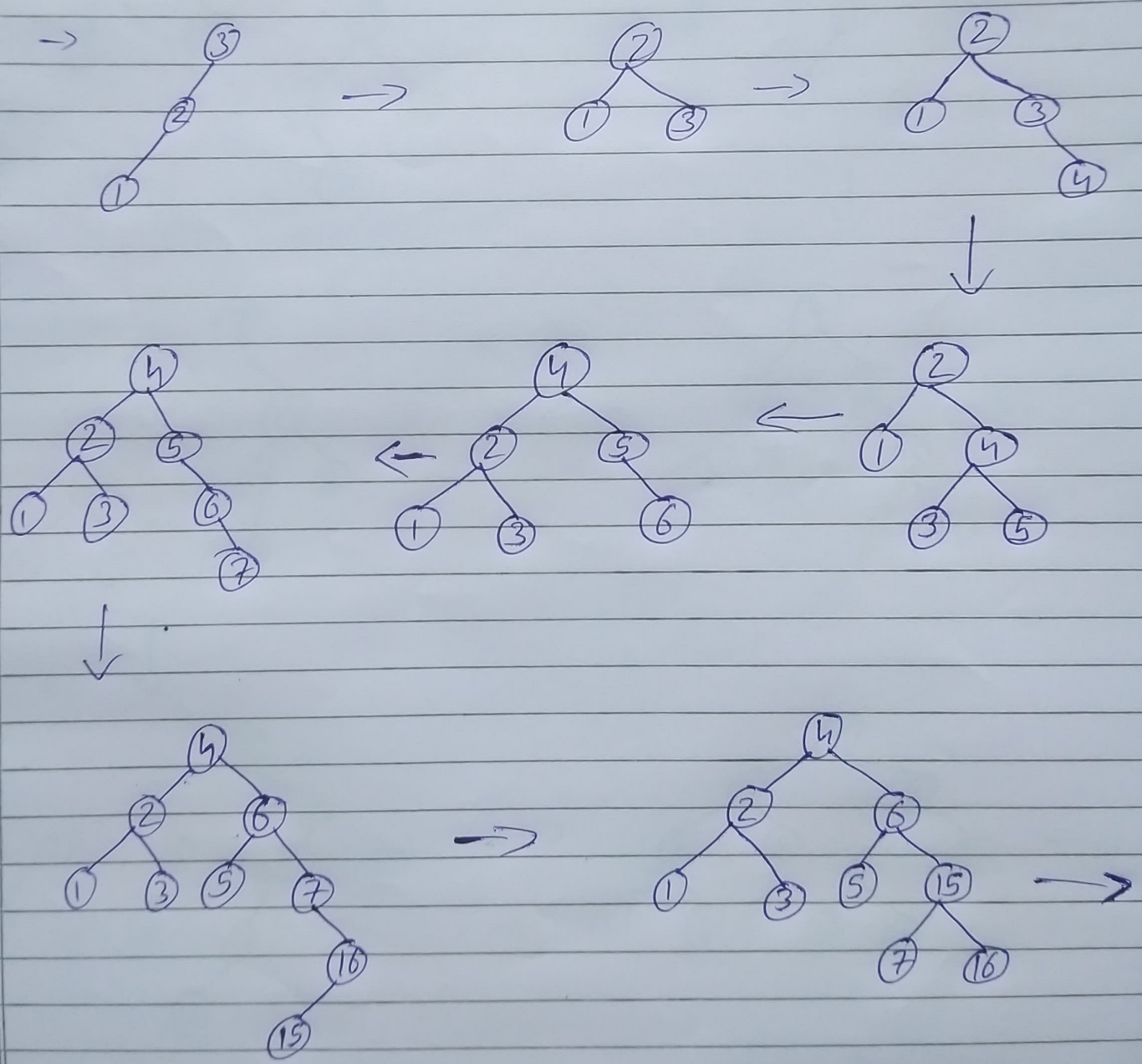


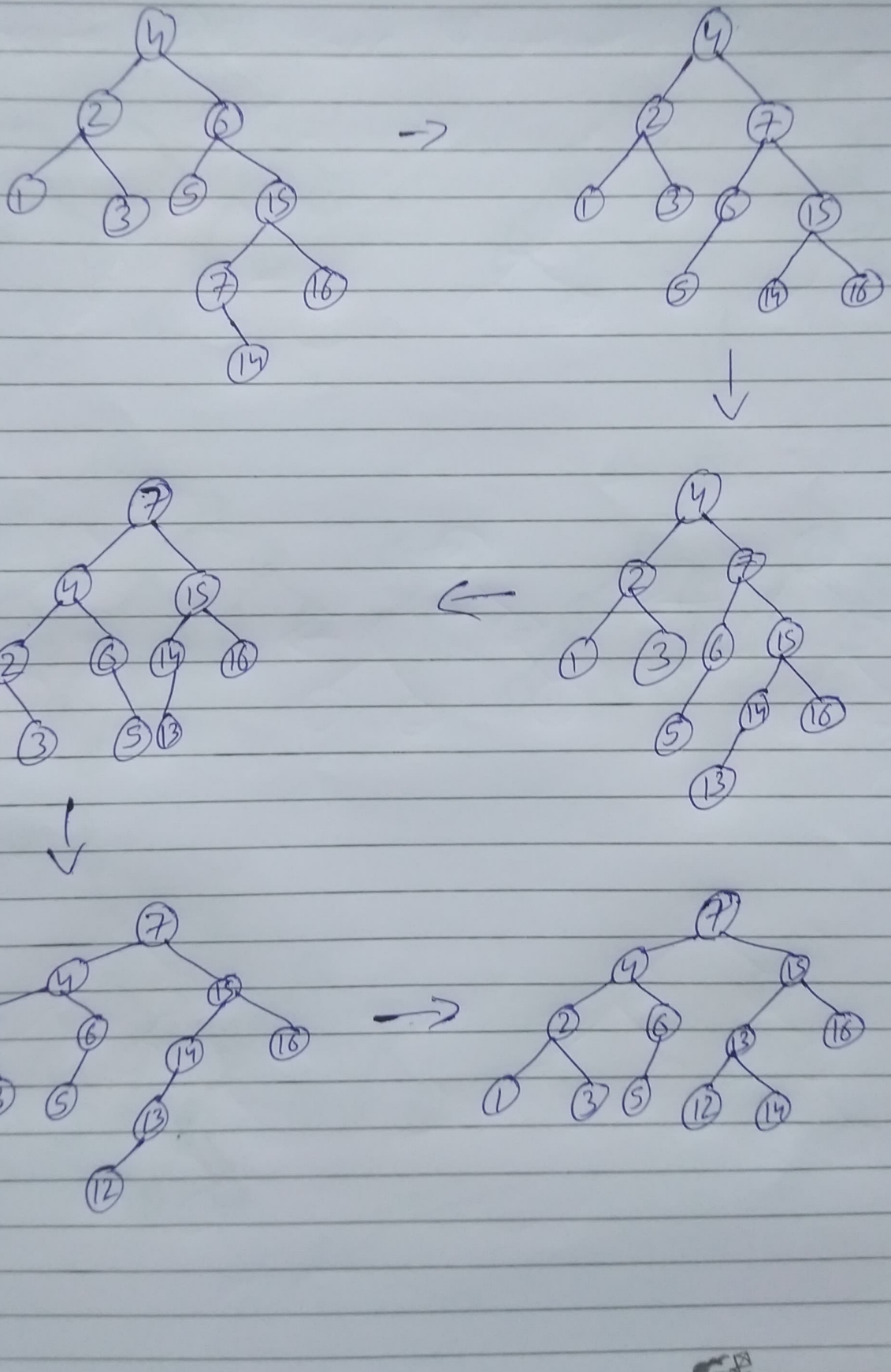
Page No. :

Date .

Q4 Create AVL tree from the following data

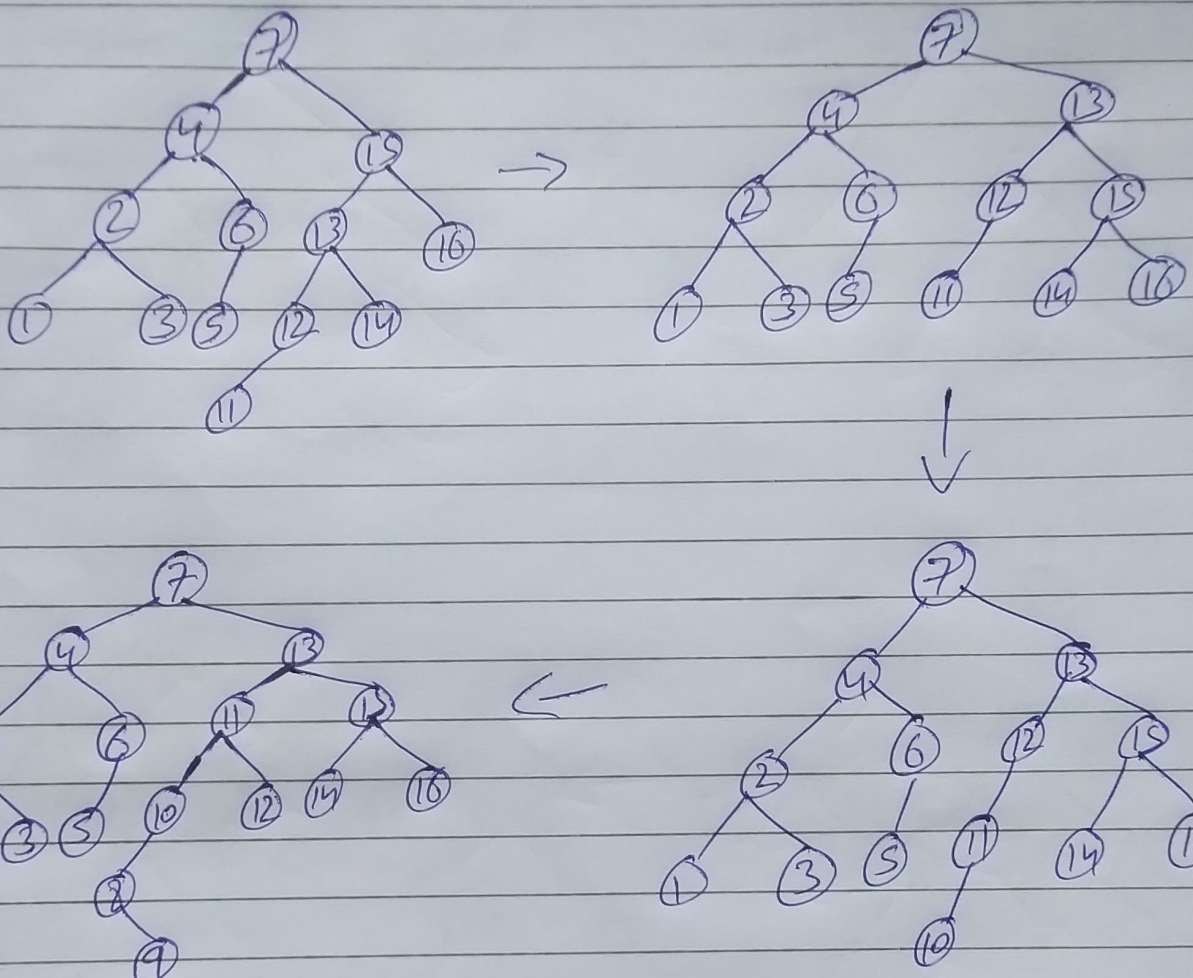
① 3, 2, 1, 4, 5, 6, 7, 16, 15, 14, 13, 12, 11, 10, 8, 9





Page No. :

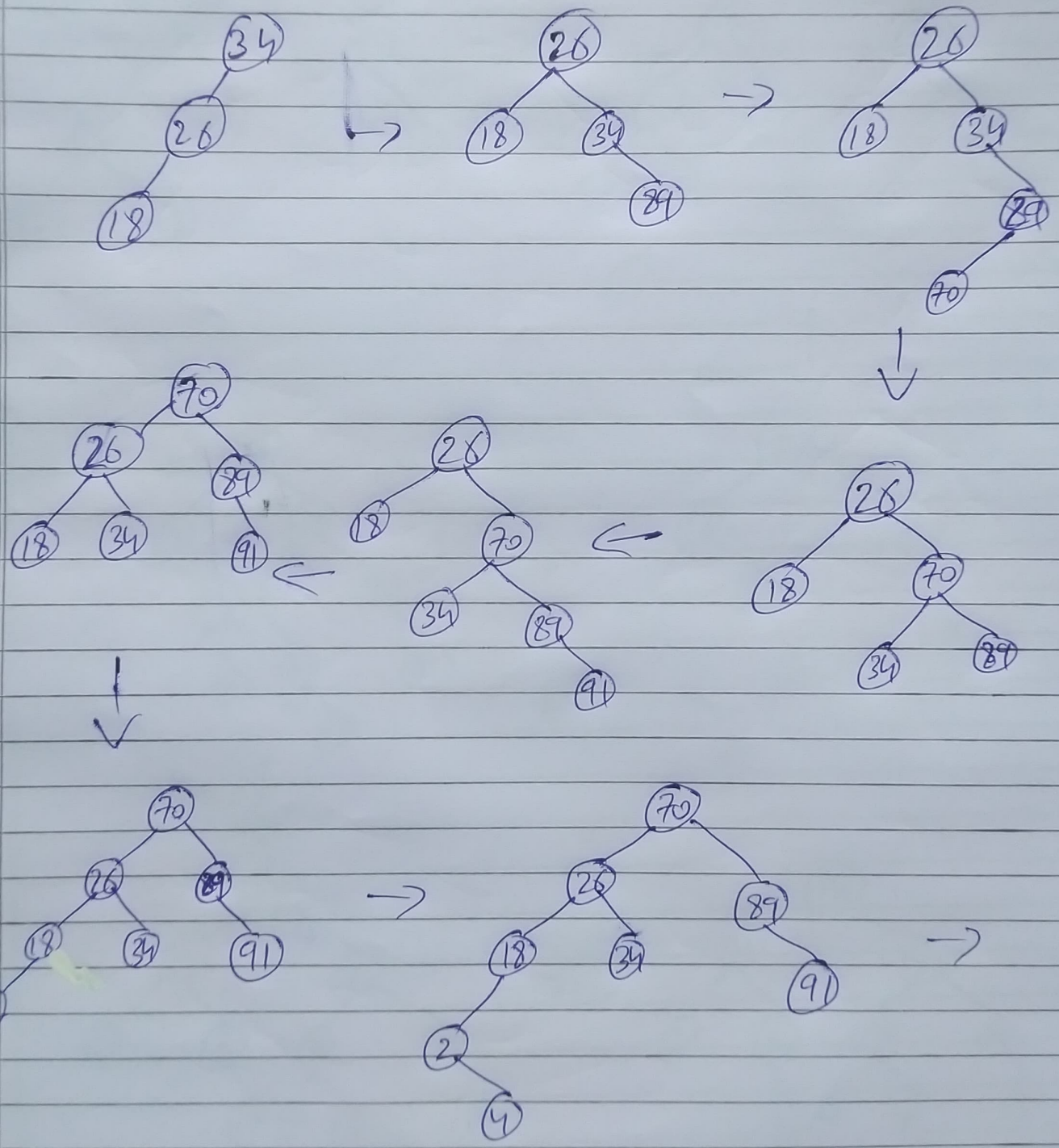
Date .



Page No. :

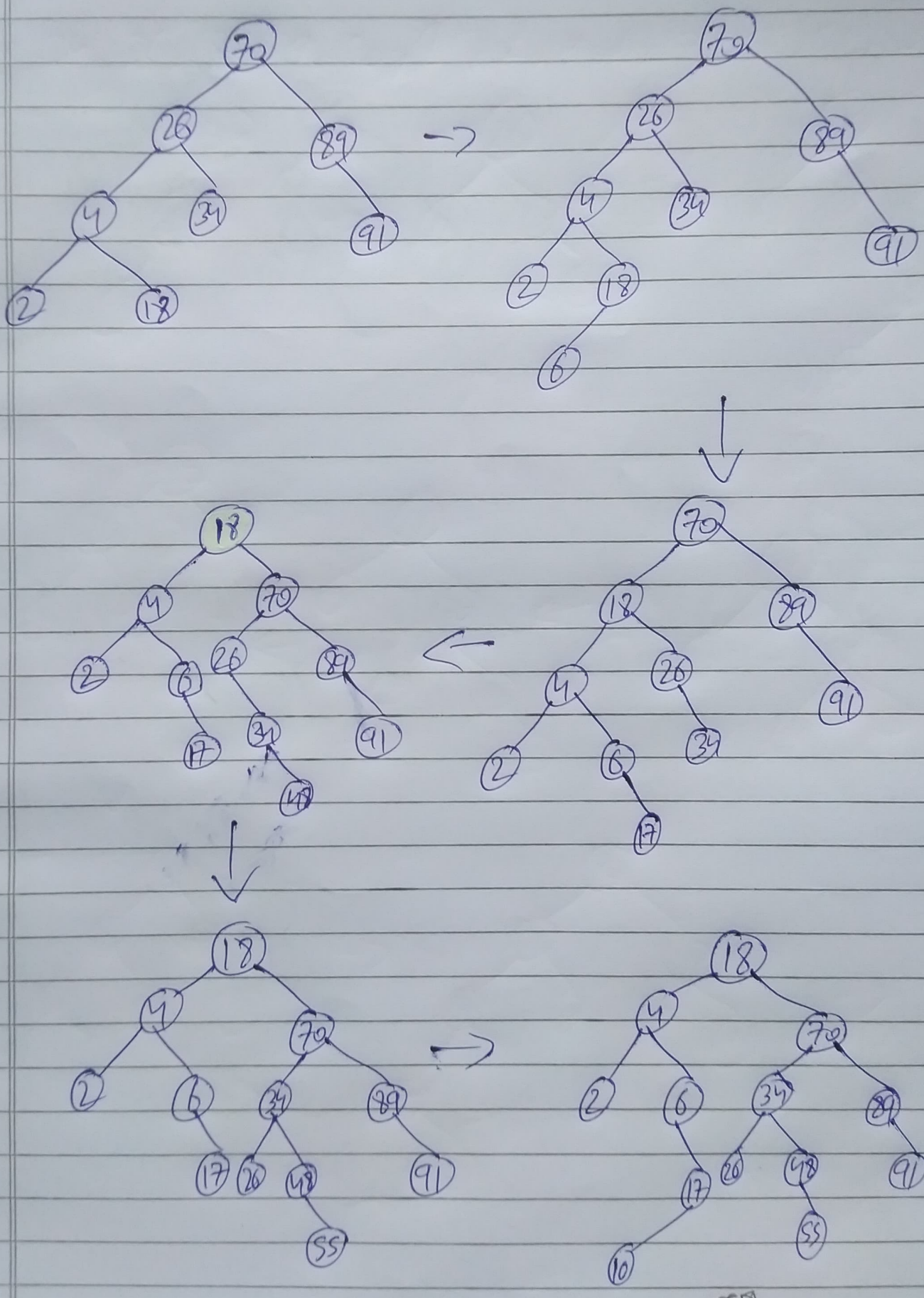
Date .

(2) 34, 26, 18, 89, 70, 91, 2, 4, 6, 17, 48, 55, 10



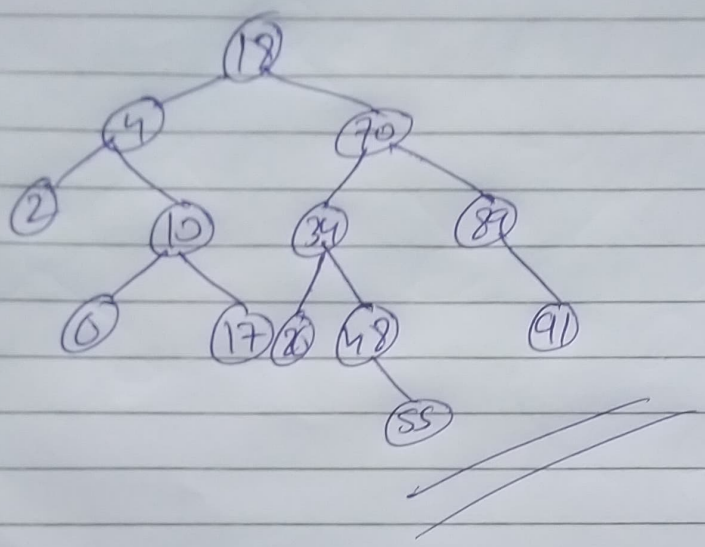
Page No. :

Date .



Page No. :

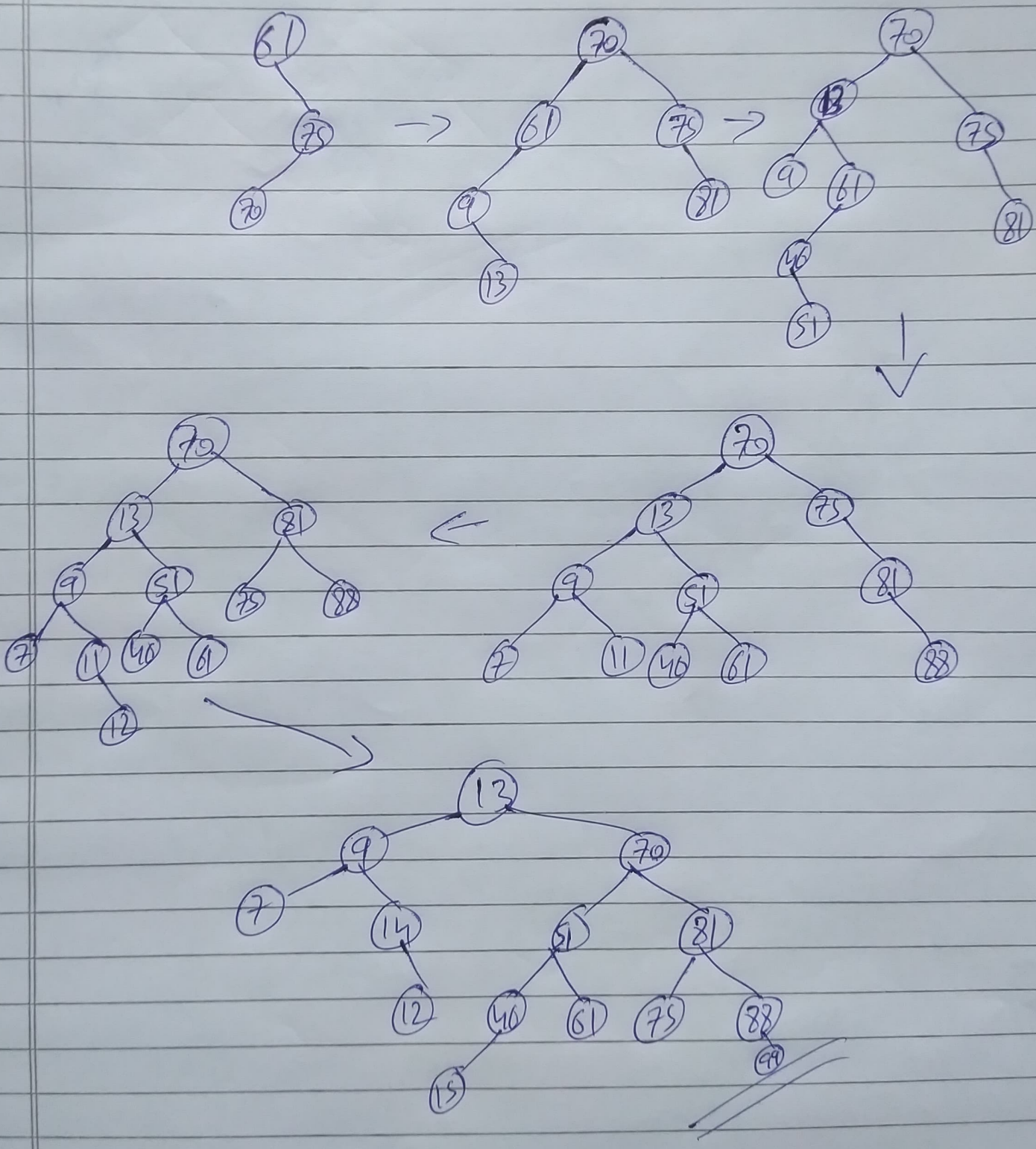
Date .



Page No. :

Date .

(3) 61, 75, 70, 81, 9, 13, 46, 51, 11, 7, 88, 99, 12, 15



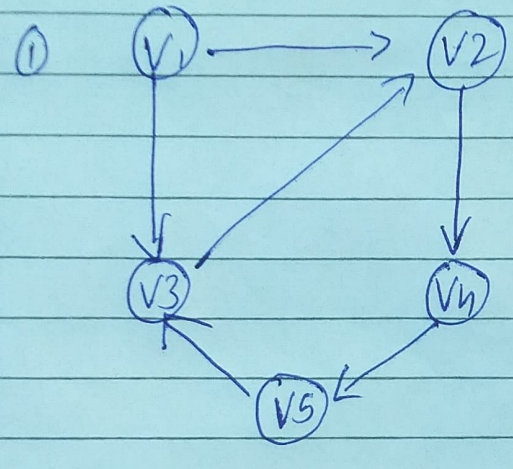
Assignment -2

Graph

Page No. :

Date .

① Unweighted shortest Path Algorithm



initial State

V	Known	dv	Pv
V_1	0	0	0
V_2	0	∞	0
V_3	0	∞	0
V_4	0	∞	0
V_5	0	∞	0

V	Known	dv	Pv
V_1	1	0	0
V_2	0	1	V_1
V_3	0	1	V_1
V_4	0	∞	0
V_5	0	∞	0

V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	0	1	V_1
V_4	0	2	V_4
V_5	0	0	0

V	Known	dv	P
V_1	1	0	0
V_2	1	1	V_1
V_3	1	1	V_1
V_4	0	2	V_4
V_5	0	∞	0

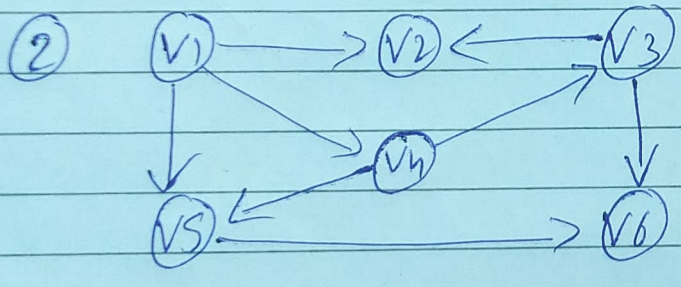
V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	1	1	V_1
V_4	1	2	V_4
V_5	1	3	V_4

Page No. :

Date .

v	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	1	1	v_1
v_4	1	2	v_4
v_5	1	3	v_4

(2)



initial value

v	known	dv	pv
v_1	0	0	0
v_2	0	∞	0
v_3	0	∞	0
v_4	0	∞	0
v_5	0	∞	0
v_6	0	∞	0

v	known	dv	pv
v_1	1	0	0
v_2	0	1	v_1
v_3	0	∞	0
v_4	0	1	v_1
v_5	0	1	v_1
v_6	0	∞	0

v	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	0	∞	0
v_4	0	1	v_1
v_5	0	1	v_1
v_6	0	∞	0

Page No. :

Date .

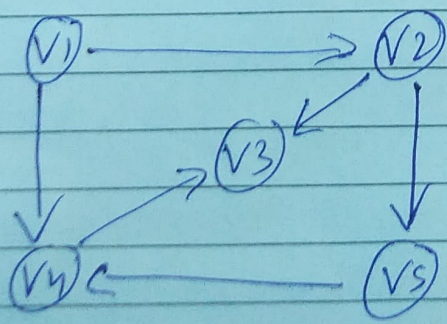
V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	0	2	V_4
V_4	1	1	V_1
V_5	0	1	V_1
V_6	0	∞	0

V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	0	2	V_4
V_4	1	1	V_1
V_5	1	1	V_1
V_6	0	2	V_5

V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	1	2	V_4
V_4	1	1	V_1
V_5	1	1	V_1
V_6	0	2	V_5

V	Known	dv	Pv
V_1	1	0	0
V_2	1	1	V_1
V_3	1	2	V_4
V_4	1	1	V_1
V_5	1	1	V_1
V_6	1	2	V_5

(3)



initial table

V	Known	dv	Pv
V_1	0	0	0
V_2	0	∞	0
V_3	0	∞	0
V_4	0	∞	0
V_5	0	∞	0

Page No. :

Date .

V	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	0	1	v_1
v_4	0	1	v_1
v_5	0	2	v_2

V	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	1	1	v_1
v_4	0	1	v_1
v_5	0	2	v_2

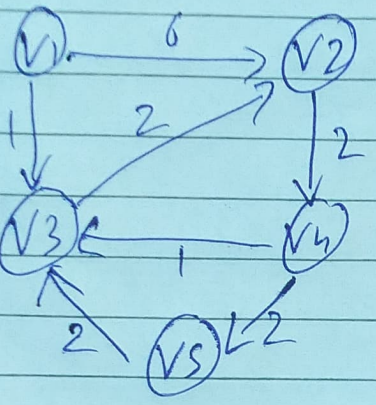
V	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	1	1	v_1
v_4	1	1	v_1
v_5	0	2	v_2

V	known	dv	pv
v_1	1	0	0
v_2	1	1	v_1
v_3	1	1	v_1
v_4	1	1	v_1
v_5	1	2	v_2

Page No. :

Date .

② Weighted shortest Path Algorithm (Dijkstra's)

① 

Initial table

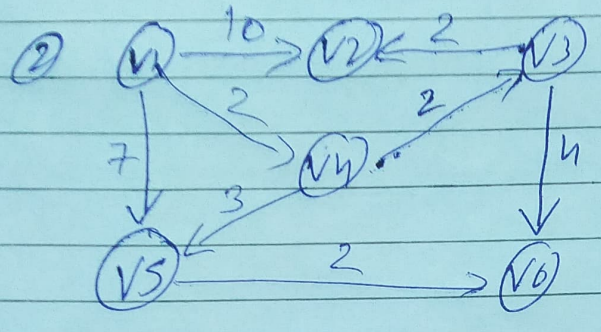
V	known	dv	Pv
V_1	0	0	0
V_2	0	∞	0
V_3	0	∞	0
V_4	0	∞	0
V_5	0	∞	0

V	known	dv	Pv
V_1	1	0	0
V_2	0	1	V_1
V_3	0	1	V_1
V_4	0	∞	0
V_5	0	∞	0

V	known	dv	Pv
V_1	1	0	0
V_2	0	3	V_3
V_3	1	1	V_1
V_4	0	∞	0
V_5	0	∞	0

V	known	dv	Pv
V_1	1	0	0
V_2	1	3	V_3
V_3	1	1	V_1
V_4	0	5	V_2
V_5	0	∞	0

V	known	dv	Pv
V_1	1	0	0
V_2	1	3	V_3
V_3	1	1	V_1
V_4	1	5	V_2
V_5	0	7	V_4



Initial table

V	known	dv	pv
V1	0	0	0
V2	0	∞	V1
V3	0	∞	Vh
Vh	0	∞	V1
V5	0	∞	Vh
V6	0	∞	0

V	known	dv	pv
V1	1	0	0
V2	0	10	V1
V3	0	∞	0
Vh	0	2	V1
V5	0	7	V1
V6	0	∞	0

V	known	dv	pv
V1	1	0	0
V2	0	10	V1
V3	0	4	Vh
Vh	1	2	V1
V5	0	5	Vh
V6	0	∞	0

V	known	dv	pv
V1	1	0	0
V2	0	6	V3
V3	1	4	Vh
Vh	1	2	V1
V5	0	5	Vh
V6	0	8	V3

V	known	dv	pv
V1	1	0	0
V2	0	6	V3
V3	1	4	Vh
Vh	1	2	V1
V5	1	5	Vh
V0	0	7	V5

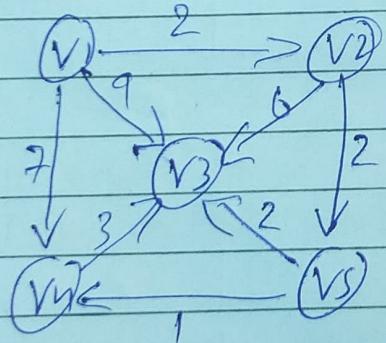
Page No. :

Date .

V	known	dv	pv
v_1	1	0	0
v_2	0	6	v_3
v_3	1	4	v_4
v_4	1	2	v_1
v_5	1	5	v_4
v_6	1	7	v_5

V	known	dv	pv
v_1	1	0	0
v_2	1	6	v_3
v_3	1	4	v_4
v_4	1	2	v_1
v_5	1	5	v_4
v_6	1	7	v_5

(3)



initial table

V	known	dv	pv
v_1	0	0	0
v_2	0	∞	0
v_3	0	∞	0
v_4	0	∞	0
v_5	0	∞	0

V	known	dv	pv
v_1	1	0	0
v_2	0	2	v_1
v_3	0	9	v_1
v_4	0	7	v_1
v_5	0	∞	0

V	known	dv	pv
v_1	1	0	0
v_2	1	2	v_1
v_3	0	6	v_2
v_4	0	7	v_1
v_5	0	4	v_2

v	known	dv	pv
v_1	1	0	0
v_2	1	2	v_1
v_3	0	6	v_2
v_4	0	5	v_5
v_5	1	4	v_2

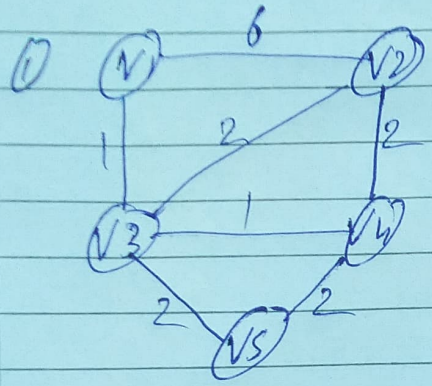
v	known	dv	pv
v_1	1	0	0
v_2	1	2	v_1
v_3	0	6	v_2
v_4	1	5	v_5
v_5	1	4	v_2

v	known	dv	pv
v_1	1	0	0
v_2	1	2	v_1
v_3	1	6	v_2
v_4	1	5	v_5
v_5	1	4	v_2

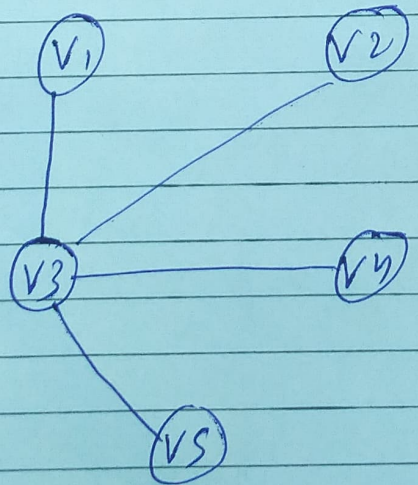
Page No. :
Date .

(3) Kruskal's Minimum spanning Tree Algorithm

(1)

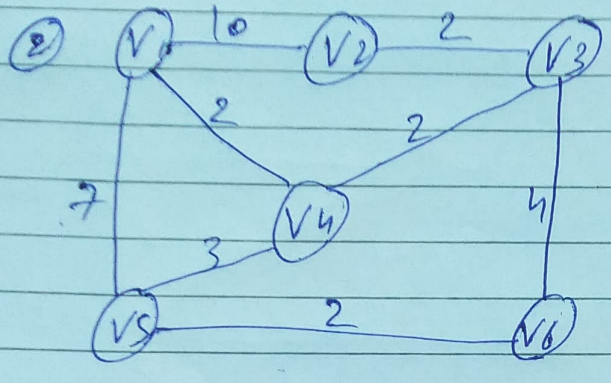


(V_1, V_3)	Accepted
(V_3, V_4)	Accepted
(V_3, V_2)	Accepted
(V_3, V_5)	Accepted
(V_4, V_5)	Rejected
(V_4, V_2)	Rejected
(V_1, V_2)	Rejected.

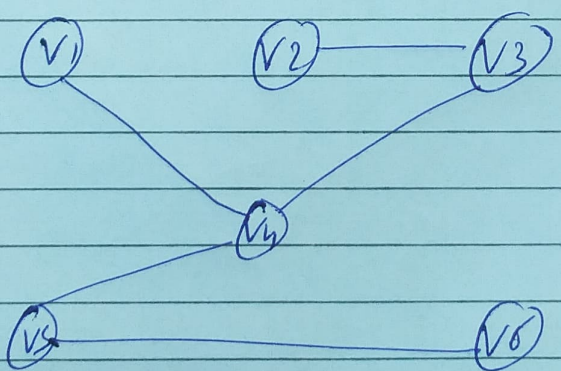


Minimum spanning tree

(2)



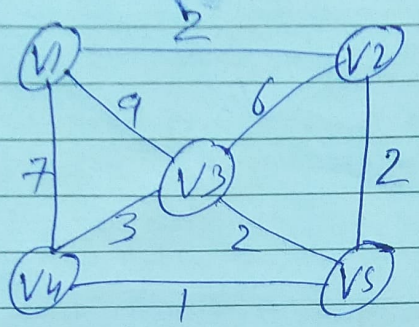
(V_2, V_3)	Accepted
(V_1, V_4)	Accepted
(V_4, V_3)	Accepted
(V_5, V_6)	Accepted
(V_5, V_4)	Accepted
(V_3, V_6)	Rejected
(V_1, V_5)	Rejected
(V_1, V_2)	Rejected



Page No. :

Date .

5)



(v_4, v_5)	Accepted
(v_1, v_2)	Accepted
(v_3, v_5)	Accepted
(v_2, v_5)	Accepted
(v_3, v_4)	Rejected
(v_3, v_2)	Rejected
(v_1, v_4)	Rejected