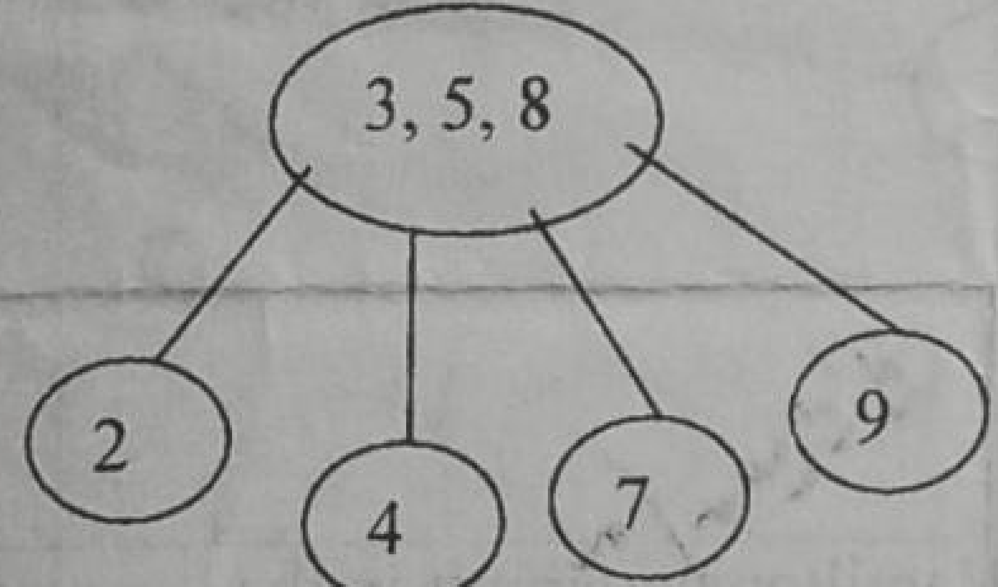
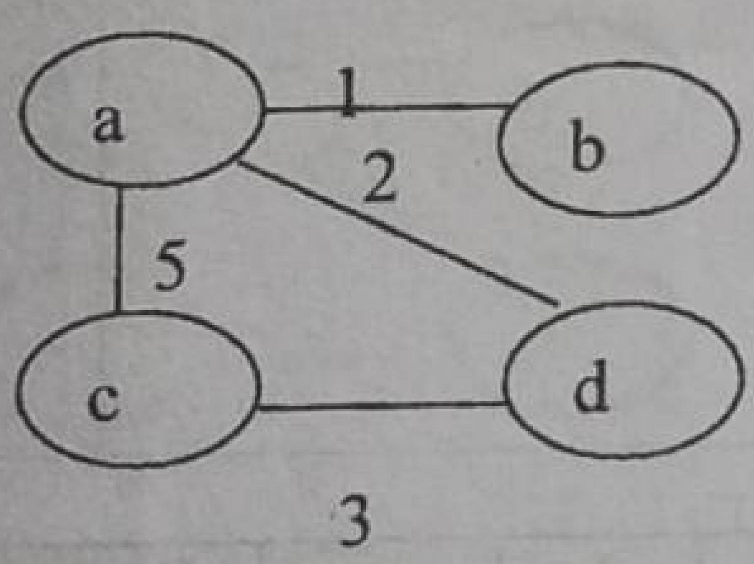
Roll No.

KONGU ENGINEERING COLLEGE, PERUNDURAI 638 060
EVEN SEMESTER 2017-2018
CONTINUOUS ASSESSMENT TEST III – April 2018
(Regulations 2014)

Programme : BE Branch : CSE Semester : IV	Date : 11.04.2018 Time : 9.15 am – 10.45 am
Course Code : 14CST43 Course Name : Design and Analysis of Algorithms	Duration : 1 ½ Hours Max. Marks : 50

PART - A (10 × 2 = 20 Marks)

ANSWER ALL THE QUESTIONS

1.	Define the property of heap.	[CO4]	[K1]
2.	Write warshall's algorithm.	[CO4]	[K2]
3.	Find out the total number of binary search trees when number of key is 5.	[CO4]	[K2]
4.	Insert 7 into the following 2-3 tree.  	[CO4]	[K3]
5.	Compare back tracking and branch and bound.	[CO5]	[K4]
6.	Draw the minimum spanning tree of the following graph.  	[CO4]	[K3]
7.	Compare tractable and intractable problem.	[CO5]	[K4]
8.	What is optimal solution? How it is differ from feasible solution?	[CO5]	[K2]
9.	Define balancing factor.	[CO4]	[K1]
10.	Write the formula to calculate warshalls algorithm.	[CO4]	[K2]

Part – B (3× 10 = 30 Marks)

ANSWER ANY THREE QUESTIONS

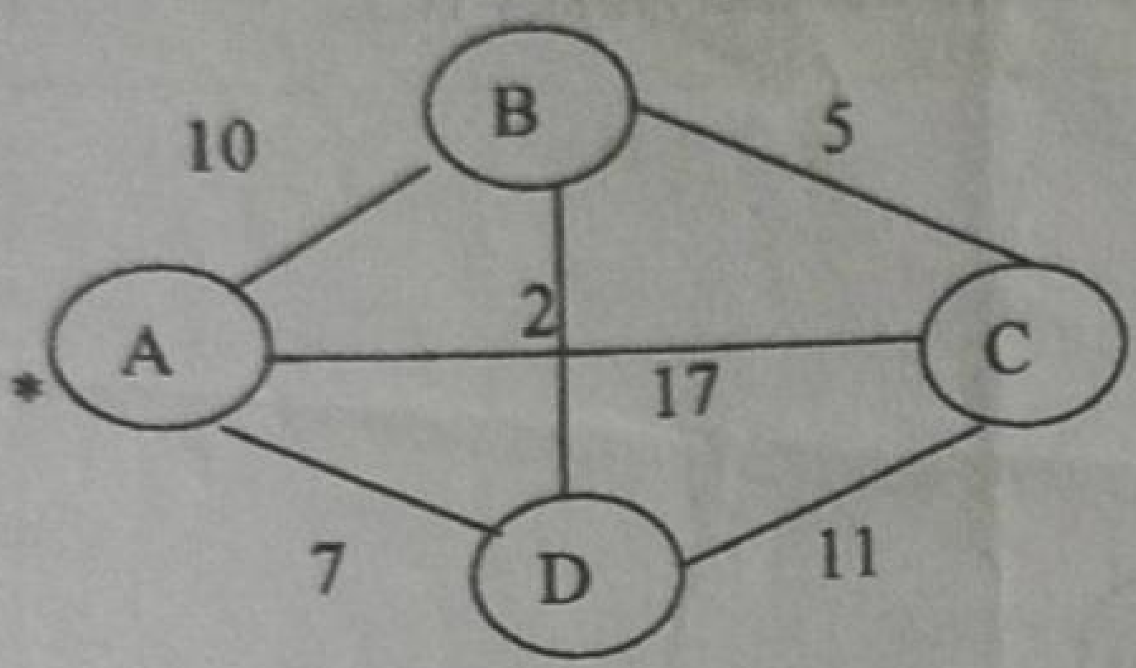
11. Apply bottom up dynamic programming algorithm to the following instance of knapsack problem. W=10 (10) [CO4]

Item	Weight	value
1	7	42
2	3	12
3	4	40
4	5	25

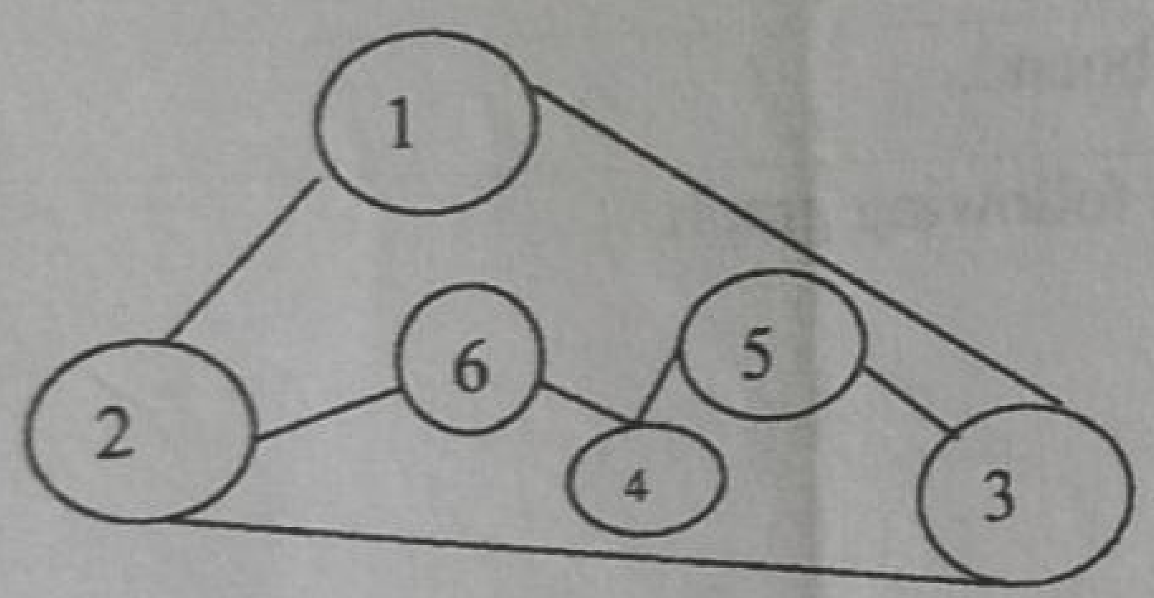
12. Construct the binary tree using the following data. (10) [CO4]

key	P	Q	R	S	T
Probability	0.2	0.05	0.35	0.15	0.25

13 Apply the branch and bound algorithm to solve travelling salesperson problem. (10) [CO5]



14 Apply backtracking to the problem to find Hamiltonian circuit in the following graph. (10) [CO5]



Bloom's Taxonomy Level	Remembering (K1)	Understanding (K2)	Applying (K3)	Analysing (K4)	Evaluating (K5)	Crea (K
Percentage	6	13	73	6	-	

DAA - Module Test III

Scheme of Valuation

Part A

1. Property of Heap
   1. Structure property
   2. Parental dominance

2. Warshall's Algorithm

Algorithm Warshall (A[1..n, 1..n])

$R^{(0)} = A$

for k = 1 to n do

for i = 1 to n do

for j = 1 to n do

$R^{(k)}[i,j] = R^{(k-1)}[i,j]$ or

$R^{(k-1)}[i,k]$ and $R^{(k-1)}[k,j]$

return $R^{(n)}$

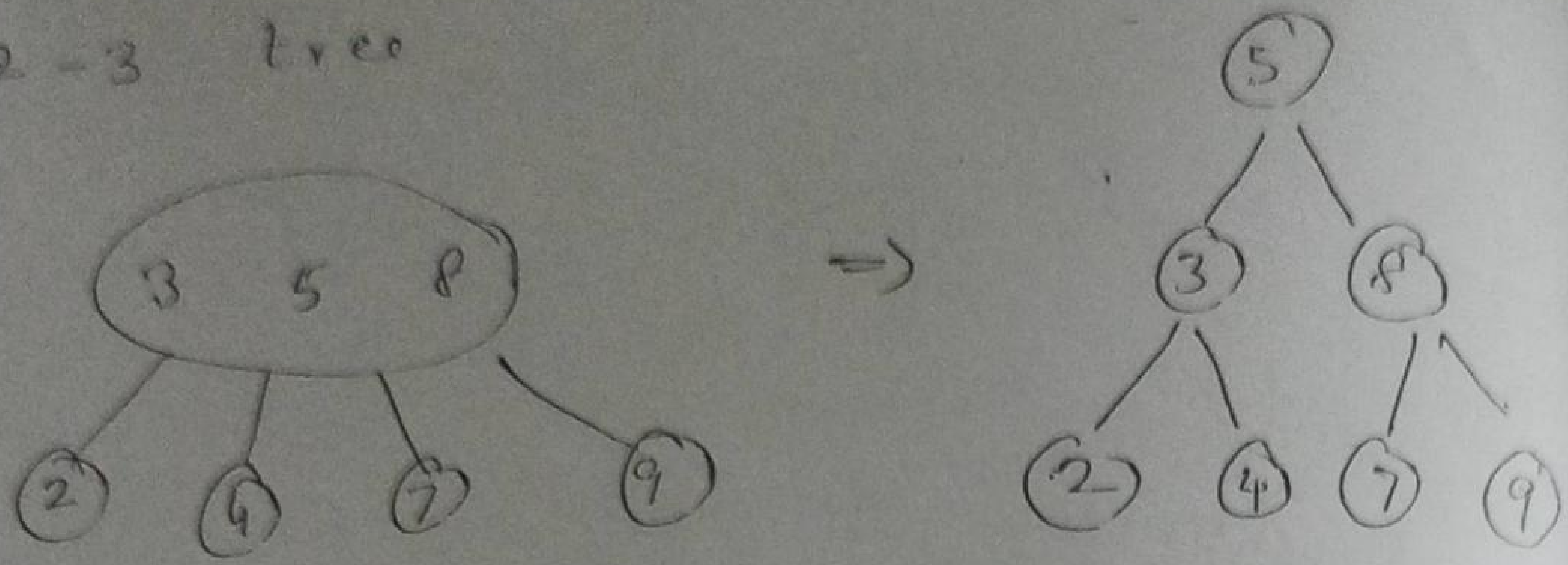
3. No of keys = 5

No. of binary search trees $= \binom{2\times5}{5} \cdot \frac{1}{5+1}$

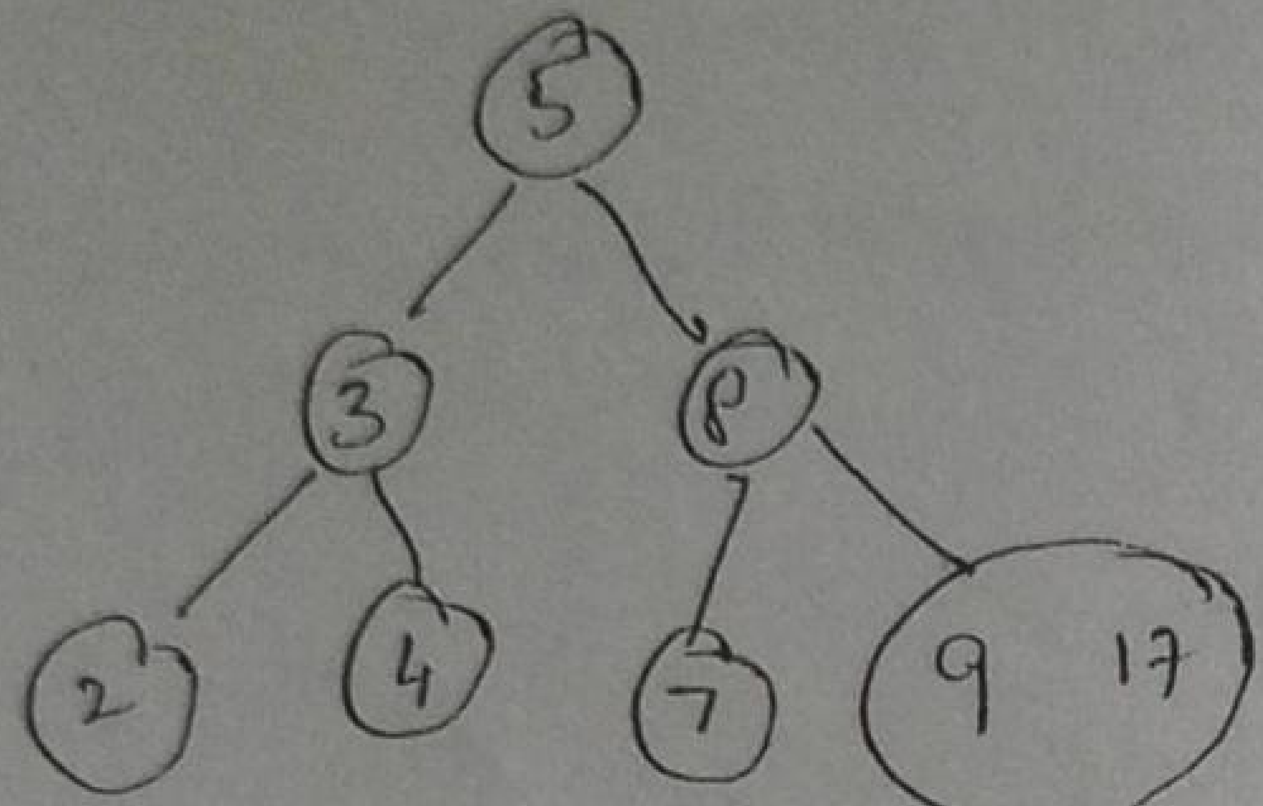
$$= \frac{10!}{5!\,5!} \cdot \frac{1}{6} = \frac{10\times9\times8\times7\times\cancel{6}\times\cancel{5!}}{5!\times\cancel{5!}\times\cancel{6}}$$

$$= \frac{\overset{2}{\cancel{10}}\times\overset{3}{\cancel{9}}\times\cancel{8}\times7}{\cancel{5}\times\cancel{4}\times\cancel{3}\times\cancel{2}} = \underline{\underline{42}}$$

4. 2-3 tree



Insert 17 :

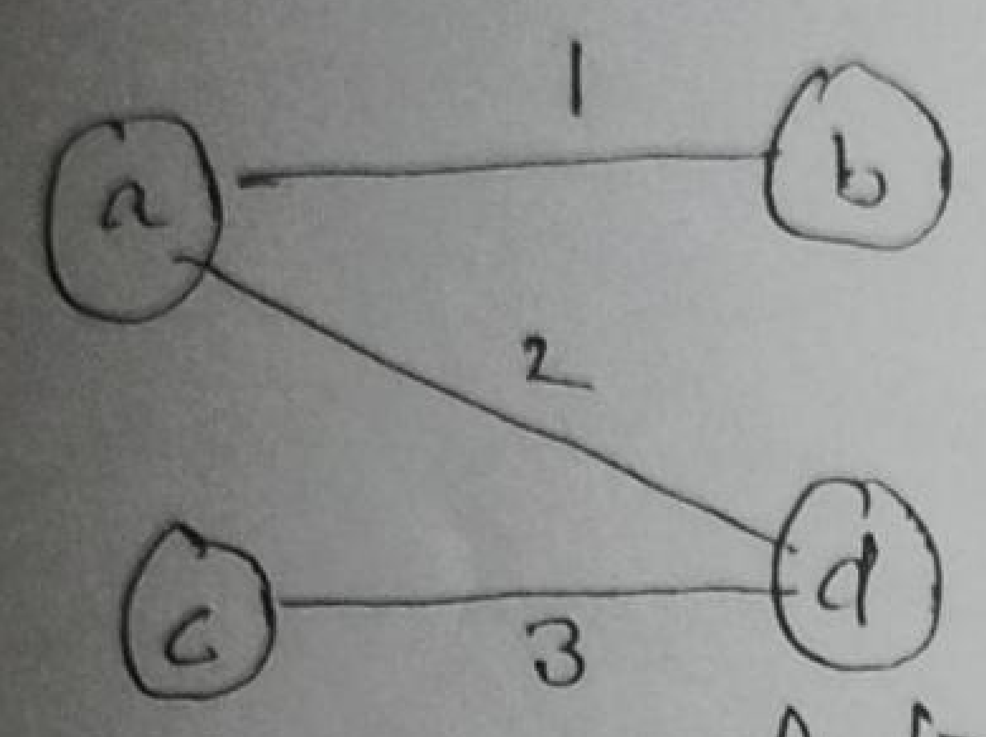


5. Back tracking:

Principal idea - construct solution one comp at a time and evaluate such partially constructed solution If this solution is cons further without violating the problem's cons then we can proceed to select the next component. If it violates, take the next op ~~for~~ If no legitimate option for next component, replace the last constructed solution with next option.

Backtracking - gives feasible solution
Branch-and-bound gives optimal solution

6. MST



weight = 6

7. Tractable ~~solution~~ Problem - Problems solved in polynomial time

Intractable problem - Problems can not be solved in polynomial ti

8. Feasible solution - solution that satisfies a problem constraints.

Optimal solution - It is the best feasi solution.

9. Balancing factor - Difference between the heights of the node's left and right sub trees.

10. Warshalls Algo:

$R^{(k)}[i,j] = R^{(k-1)}[i,j]$ or $[R^{(k-1)}[i,k]$ and $R^{k-1}[k,$

Part B

11 Knapsack problem using dynamic programming

Item	Weight	Value
1	7	42
2	3	12
3	4	40
4	5	25

$W = 10.$

$0 + 12 +$ 12

$V[1,3]$, $v_2 + V[1,0]$

$$V[i,j] = \begin{cases} \max\left(V[i-1,j],\ v_i + V[i-1, j-w_i]\right) & \text{if } j-w_i \geq 0 \\ V[i-1,j] & \text{if } j - w_i < 0 \end{cases}$$

$V[0,j] = 0$ for $j \geq 0$ $\quad V[i,0] = 0$ for $i \geq 0$

capacity j

i	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	[illegible]
1	0	0	0	0	0	0	0	42	42	42	[illegible]
2	0	0	0	12	12	12	12	42	42	42	[illegible]
3	0	0	0	12	40	40	40	52	52	52	[illegible]
4	0	0	0	12	40	40	40	52	52	65	[illegible]

Subset of items : $\{4, 3\}$

Value of the knapsack = 65 ✓

$i=1, j=7$

$(1,0)$ $i=1$ $j=1$

$1-7 < 0$

$-6 < 0$

$V[2,3]$
$i=2$ $j=3$
$3 - 3 \leq 0$

$V[0,1]$

12. Optimal Binary Search Tree [1, 5]

key	P	Q	R	S	T
Prob.	0.2	0.05	0.35	0.15	0.25

$$c[i,j] = \min_{i \le k \le j} \; c[i,k-1] + c[k+1,j] + \sum_{s=i}^{j} p_s$$

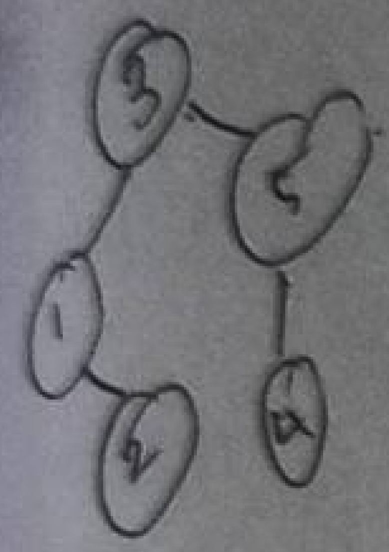
for $1 \le i \le$

Main Table

	0	1	2	3	4	5
1	0	0.2	0.3	0.9	1.2	1.85
2		0	0.05	0.45	0.75	1.4
3			0	0.35	0.65	1.3
4				0	0.15	0.55
5					0	0.25
6						0

Root Table.

	0	1	2	3	4
1		1	1	3	3
2			2	3	3
3				3	3
4					5
5					
6					



1, 5

$$c[1,2] = \begin{cases} k=1 & c[1,0] + c[2,2] + \sum_{s=1}^{2} ps = 0 + 0.05 + 0.25 \\ k=2 & c[1,1] + c[3,2] + \sum_{s=1}^{2} ps = 0.2 + 0 + 0.25 = 0. \end{cases}$$

$$c[2,3] = \begin{cases} k=2 & c[2,1] + c[3,3] + \sum_{s=2}^{3} ps = 0.00 + 0.35 + 0.4 = 0 \\ k=3 & c[2,2] + c[4,3] + \sum_{s=2}^{3} ps = 0.05 + 0 + 0.4 = \underline{0.} \end{cases}$$

$$c[3,4] = \begin{cases} k=3 & c[3,2] + c[4,4] + \sum_{s=3}^{4} ps = 0 + 0.15 + 0.5 = \underline{0} \\ k=4 & c[3,3] + c[5,4] + \sum_{s=3}^{4} ps = 0.35 + 0 + 0.5 = 0 \end{cases}$$

$$c[4,5] = \begin{cases} k=4 & c[4,3] + c[5,5] + \sum_{s=4}^{5} ps = 0 + 0.25 + 0.4 = 0 \\ k=5 & c[4,4] + c[6,5] + \text{"} = 0.15 + 0 + 0.4 = \underline{0.} \end{cases}$$

$$C[1,3] = \begin{cases} k=1 & C[1,0] + C[2,3] + \sum_{s=1}^{3} ps = 0 + 0.45 + 0.6 = \\ k=2 & C[1,1] + C[3,3] + \text{"} = 0.2 + 0.35 + 0.6 = \\ k=3 & C[1,2] + C[4,3] + \text{"} = 0.3 + 0 + 0.6 = 0 \end{cases}$$

$$C[2,4] = \begin{cases} k=2 & C[2,1] + C[3,4] + \sum_{s=2}^{4} ps = 0 + 0.65 + 0.55 = 1 \\ k=3 & C[2,2] + C[4,4] + \text{"} = 0.05 + 0.15 + 0.55 = \\ k=4 & C[2,3] + C[5,4] + \text{"} = 0.45 + 0 + 0.55 = \end{cases}$$

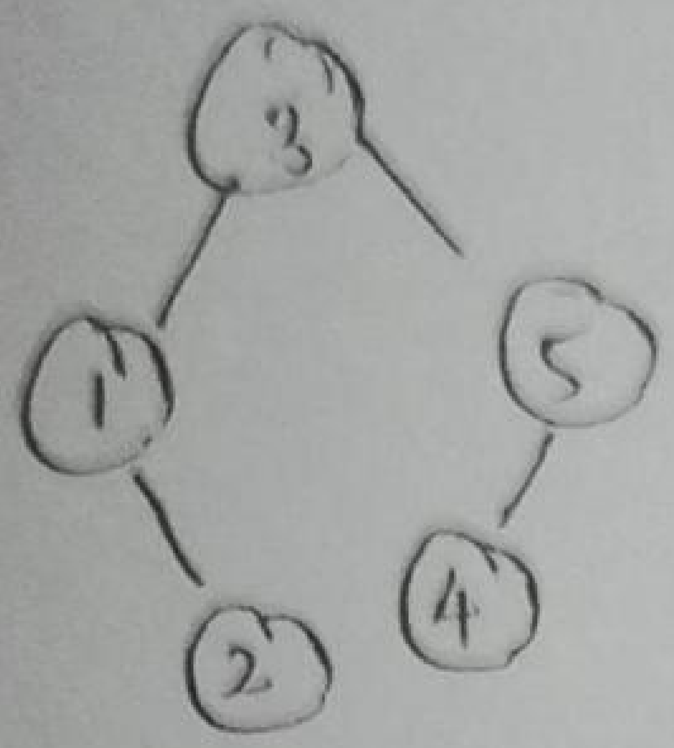
$$C[3,5] = \begin{cases} k=3 & C[3,2] + C[4,5] + \sum_{s=3}^{} ps = 0 + 0.55 + 0.75 = 1 \\ k=4 & C[3,3] + C[5,5] + \text{"} = 0.35 + 0.25 + 0.75 = 1 \\ k=5 & C[3,4] + C[6,5] + \text{"} = 0.65 + 0 + 0.75 = 1.4 \end{cases}$$

$$C[1,4] = \begin{cases} k=1 & C[1,0] + C[2,4] + \sum_{s=1}^{4} ps = 0 + 0.75 + 0.75 = 1.5 \\ k=2 & C[1,1] + C[3,4] + \text{"} = 0.2 + 0.65 + 0.75 = 1.6 \\ k=3 & C[1,2] + C[4,4] + \text{"} = 0.3 + 0.15 + 0.75 = 1.2 \\ k=4 & C[1,3] + C[5,4] + \text{"} = 0.9 + 0 + 0.75 = 1.6 \end{cases}$$

$$C[2,5] = \begin{cases} k=2 & C[2,1] + C[3,5] + \sum_{s=2}^{5} ps = 0 + 1.3 + 0.8 = 2.1 \\ k=3 & C[2,2] + C[4,5] + \text{"} = 0.05 + 0.55 + 0.8 = 1.4 \\ k=4 & C[2,3] + C[5,5] + \text{"} = 0.45 + 0.25 + 0.8 = 1.5 \\ k=5 & C[2,4] + C[6,5] + \text{"} = 0.75 + 0 + 0.8 = 1.55 \end{cases}$$

$$C[1,5] = \begin{cases} k=1 & C[1,0] + C[2,5] + \sum_{s=1}^{5} ps = 0 + 1.4 + 1 = 2.4 \\ k=2 & C[1,1] + C[3,5] + \text{"} = 0.2 + 1.3 + 1 = 2.5 \\ k=3 & C[1,2] + C[4,5] + \text{"} = 0.3 + 0.55 + 1 = 1.85 \\ k=4 & C[1,3] + C[5,5] + \text{"} = 0.9 + 0.25 + 1 = 2.15 \\ k=5 & C[1,4] + C[6,5] + \text{"} = 1.2 + 0 + 1 = 2.2 \end{cases}$$

Optimal Binary Search Tree



13. TSP

start
lb =

lb = [(10 + 7) + (2 + 5) + (5 + 11) + (2 +
= 25

a, b
lb = 27

a, c
lb = 31

a, d
lb 25

a, d, b, c
c → a
cost = 31

a, d, c, b
b → a
cost = 33

a, b

lb = [(7 + 10) + (8 + 10) + (5 + 11) + (2 + 7)] / 2 = 27

a, c

lb = [(7 + 17) + (2 + 5) + (17 + 5) + (2 + 17)] / 2 = 31

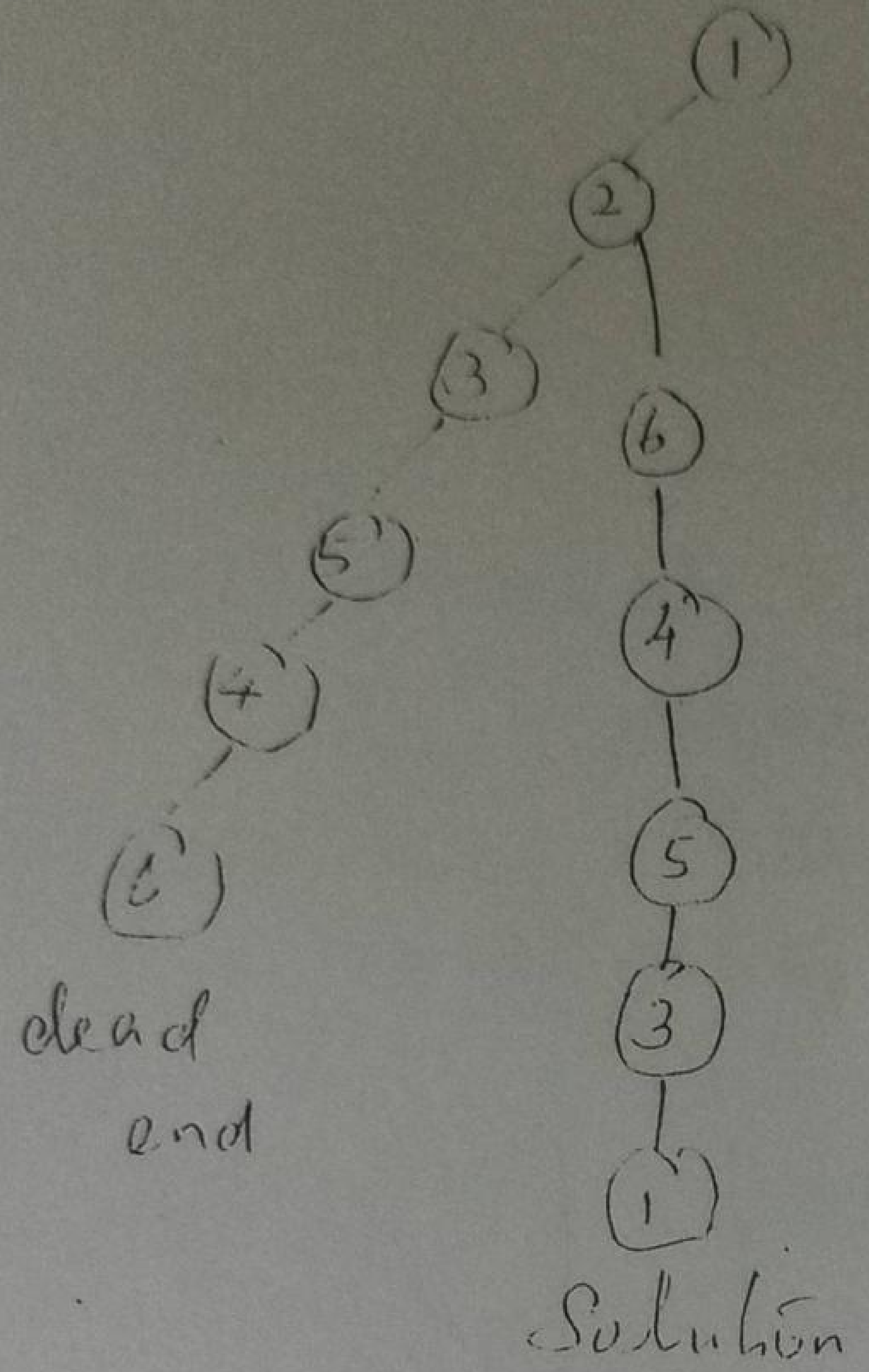
a, d

lb = [(10 + 7) + (2 + 5) + (5 + 11) + (2 + 7)] / 2 = 25

Best Tour a → d → b → c → a

Cost 31

14. Hamiltonian circuit



Scanned by TapScanner

DEPT. OF COMPUTER SCIENCE & ENGG.
KONGU ENGINEERING COLLEGE,
THOPPUPALAYAM (PO)
PERUNDURAI (TK), ERODE - 638 060

Dr. R. MANJULA Dean	[illegible]
Name and Signature of Hall Supdt. with Date	

KONGU ENGINEERING COLLEGE

PERUNDURAI ERODE - 638 060.

(Autonomous)

Estd : 1984

Name of the Student	V.P. Hariharan	Register No.	1 6 C S R 0 6 7
Programme	BE	Branch & Semester	CSE IV
Course Code and Name	14CST43 Design and Analysis of Algorithms.	Date	28.02.18 — No. of Pages Used: 21

MARKS TO BE FILLED IN BY THE EXAMINER

PART - A		PART - B			Grand Total Max. Marks : 50
Question No.	Max Marks : 2	Question No.		Max Marks : 10	
1		11	i)	~~10~~ 9	Verified by V.P. Hariharan
2	2		ii)		
3		12	i)	1	31/50
4			ii)	3	
5	2	13	i)		
6	2		ii)		
7	2	14	i)	10	
8			ii)		
9					
10					
TOTAL	8	TOTAL		23	

Total Marks in Words : Three one

INSTRUCTION TO THE CANDIDATE

1. Check the Question Paper, Programme, Course Code, Branch Name etc., before answering the questions.
2. Use both sides of the paper for answering questions.
3. POSSESSION OF ANY INCRIMINATING MATERIAL AND MALPRACTICE OF ANY NATURE IS PUNISHABLE AS PER RULES.

Name of the Examiner: [illegible]

Signature of the Examiner with Date: [illegible]

Part-B.

11). Quick Sort - Average case.

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

$$T(1) = 0$$

$$T(n-1) = \frac{1}{n}T(1) + \frac{1}{n}T(2) + \frac{1}{n}T(3) \dots + \frac{1}{n}T(n-1)$$

$$T(n) = \frac{1}{n}T(n-1) + \frac{1}{n}T(n-2) \dots \frac{1}{n}T(2) + \frac{1}{n}T(1).$$

$$T(n) = \frac{1}{n}\sum_{j=0}^{n-1} T(j) + \frac{1}{n}\sum_{j=0}^{n-1} T(j) + n$$

$$= \frac{2}{n}\sum_{j=0}^{n-1} T(j) + n. \quad \text{--- } 1$$

Multiply by n on both sides,

$$nT(n) = 2\sum_{j=0}^{n-1} T(j) + n^2. \quad \text{--- } 2$$

Sub $n = n-1$.

$$(n-1)T(n-1) = 2\sum_{j=0}^{n-1} T(j) + (n-1)^2 \quad \text{--- } 3$$

Eq 2 - 3, we get.

$$nT(n) - (n-1)T(n-1) = 2T(n-1) + \cancel{n-1}\ n^2 - (n-1)^2$$

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

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

$$nT(n) = (n+1)T(n-1) + 2n. \quad \text{--- } 4$$

Scanned by TapScanner

Divide ④ $\log n(n+1)$ on both sides.

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

$$\frac{T(n)}{n+1} = \frac{(n-1)\,T}{n} + \frac{2}{n+1} \quad \text{— ⑤}$$

Sub $n = n-1$

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

Sub the value of $\frac{T(n-1)}{n}$ in ⑤.

$$\frac{T(n)}{n+1} = \frac{T(n-2)}{n-1} + \frac{2}{n} + \frac{2}{n+1} \quad \text{— ⑥}$$

Sub $n = n-2$

$$\frac{T(n-2)}{n-1} = \frac{T(n-3)}{n-2} + \frac{2}{n-1}$$

Sub the value of $\frac{T(n-2)}{n-1}$ in ⑥.

$$\frac{T(n)}{n+1} = \frac{T(n-3)}{n-2} + \frac{2}{n-1} + \frac{2}{n} + \frac{2}{n+1}$$

b + 1

General Equation. ($n = 4$) in R.H.S

$$\frac{T(n)}{n+1} = \frac{T(1)}{2} + \frac{2}{3} + \frac{2}{4} + \frac{2}{5} \ldots \frac{2}{n-1} + \frac{2}{n} + \frac{2}{n+1}$$

$T(1) = 0.$

$$= 2\sum_{j=3}^{n+1} \frac{1}{j}$$

$$= 2\sum_{j=3}^{n+1} \frac{1}{j} - \frac{1}{1} - \frac{1}{2}$$

$$= \log(n+1).$$

$$\frac{T(n)}{n+1} = \log(n+1)$$

$$T(n) = (n+1)\log(n+1)$$

$$T(n) \simeq n\log n$$

$$T(n) \simeq O(n\log n).$$

2

Algorith for Quick Sort

Alg-QuickSort (A [l.. r]
{
If (l_p < r_p)
{
P= Partition (A [l_p - r_p])
Quick Sort = S [A [l_p ... P-1)
Quick Sort = S [A [P+1 ... r_p])
}
Alg Partition (A [l_p .. r_p])
{
p = A [l_p]
i = l_p+1 = r_p
while (i<j)
{
while (A [i] < P)
i ++ ;
Swap [A [i], A [j])
i++, j--
Swap (A [j], A [l_p])
return j
}.

12) i) Algorithm for Insertion Sort :

(arrange the list of n numbers in ascending order)

Algorithm InsertionSort [A [0... n-1]

{.

for i=1 to n-1

{.

v = A [i]

while (v ≥ 0 and A [j] > i)

{

A [j+1] ... A [j]

j = v - 1.

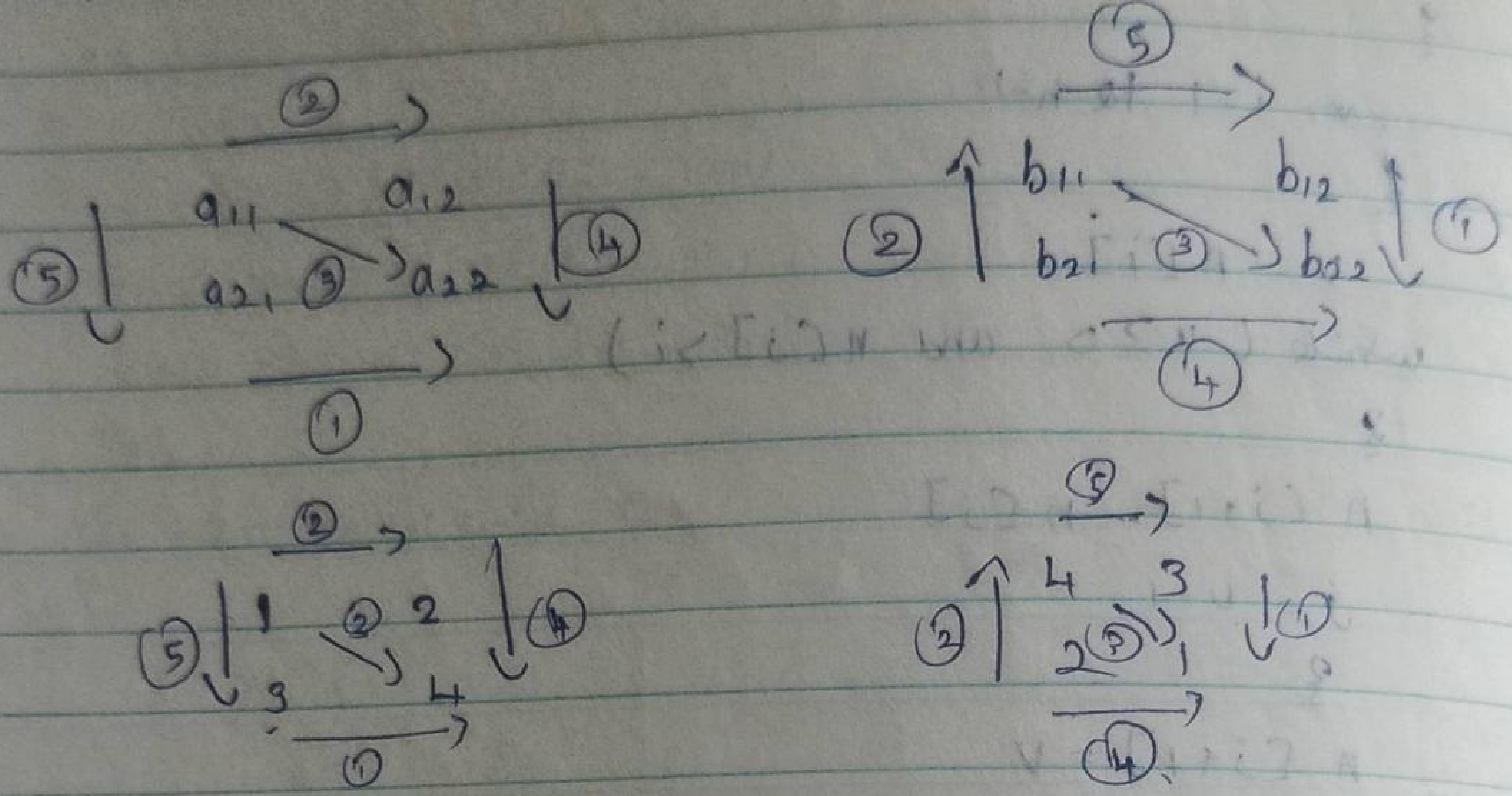
}

A [j+1] = v

}.

ii) Strassen's Matrix Multiplication Method

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$$



3

$B_{12} - B_{22} = 2$	$P_1 = A_{11} \cdot S_1$
$A_{21} + A_{22} = -1$	$P_2 = S_2 \cdot B_{11}$
$A_{11} + A_{21} = -1$	$P_3 = S_3 \cdot B_{12}$
$B_{21} - B_{11} = -2$	$P_4 = A_{12} \cdot S_4$
$A_{11} + A_{22} = -3$	$P_5 = S_5 \cdot S_6$
$B_{11} + B_{22} = 3$	$P_6 = S_7 \cdot S_8$
$A_{12} - A_{22} = -2$	$P_7 = S_9 \cdot S_{10}$
$B_{21} + B_{22} = 1$	$P_7 =$
$A_{11} - A_{21} = -2.$	$P_8 =$
$B_{11} + B_{21} = 1$	$P_9 =$

$$= \begin{pmatrix} -P_1 + P_3 + P_5 + P_7 & P_5 + P_6 \\ P_3 + P_4 & + P_2 - P_4 + P_8 + P_6 \end{pmatrix}$$

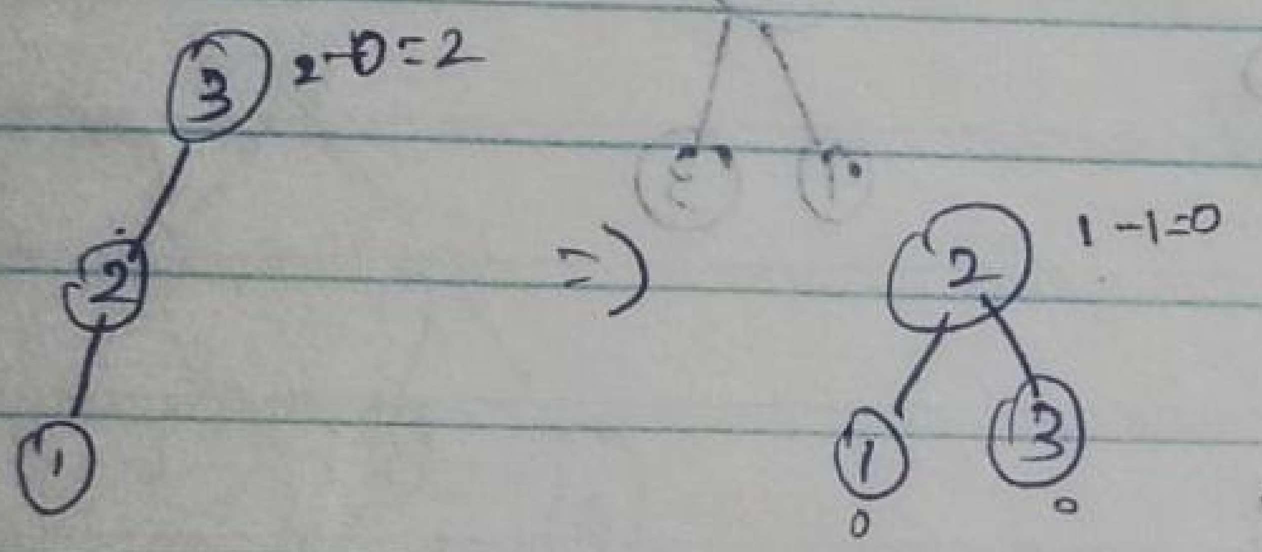
$$= \begin{pmatrix} 8 & 5 \\ 20 & 13 \end{pmatrix}.$$

14). Various types of rotations in AVL tree.

* Single Right rotation (or) Right Rotation.
* Single left rotation (or) left rotation.
* Double Right rotation
* Double left rotation.

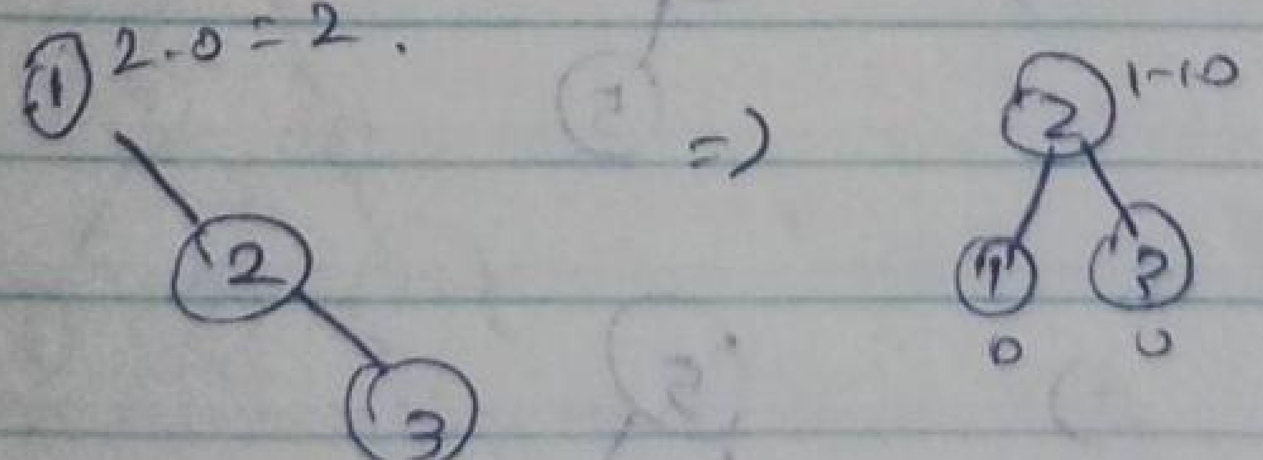
AVL Satisfying balancing nodes = -1, +1, 0.

Right Rotation.



This type of rotation happens when an extra key is add to a Right Subtree of a Right child, and the root node has an balancing factor of +1.

left Rotation.

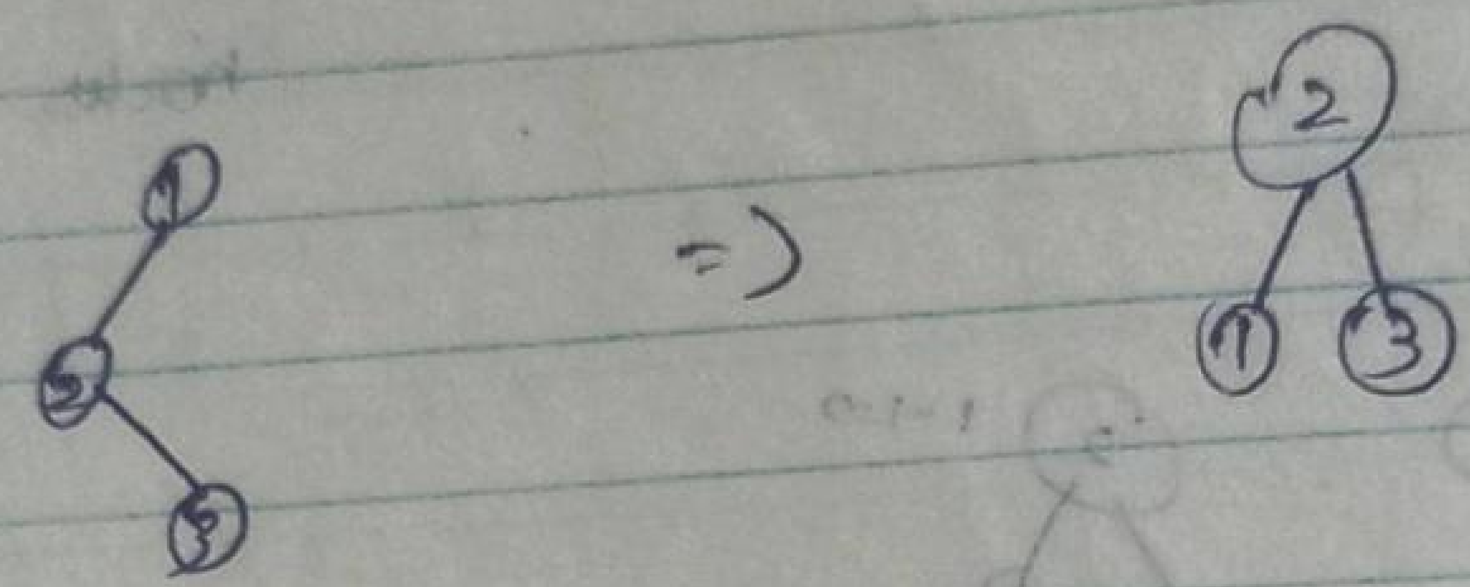


This type of rotation happens when an extra key is added to a left Subtree of a left child, and the root node has an balancing factor of -1.

Double Right Rotation.

This type of Rotation happens when an extra key is added to the Right Subtree of a left child.

Double left Rotation.

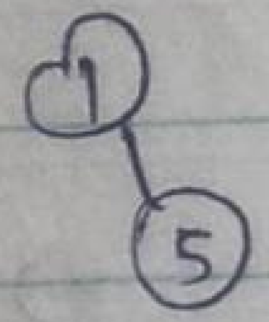


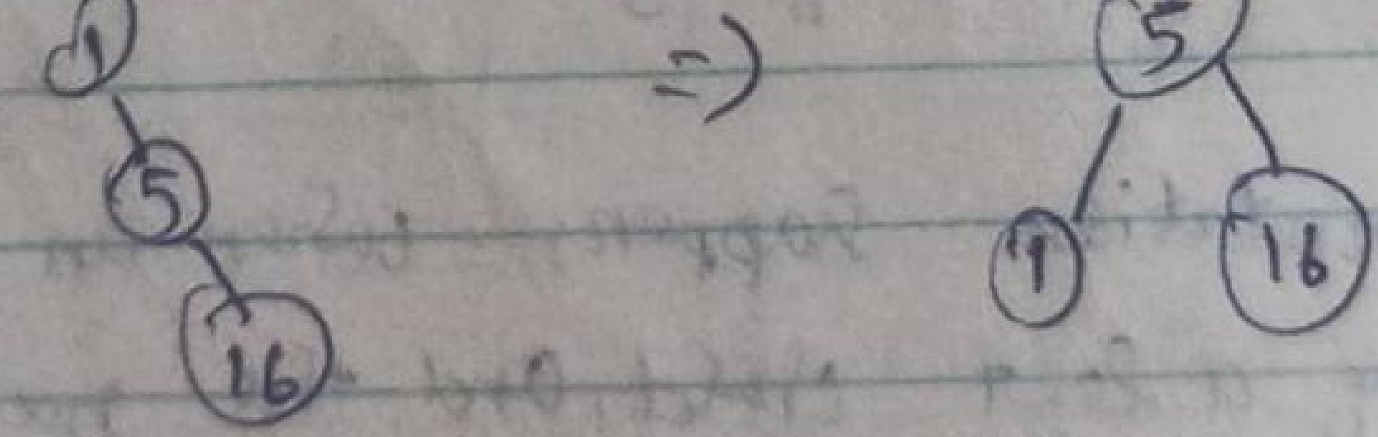
Balancing factor (node):

The Height of left Sub tree nodes − the height of right sub tree node from the root or parent node.

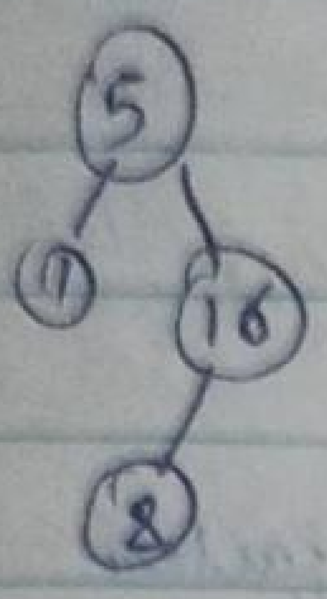
AVL. (1, 5, 16, 8, 10, 2, 22, 7, 6, 1)

1) 

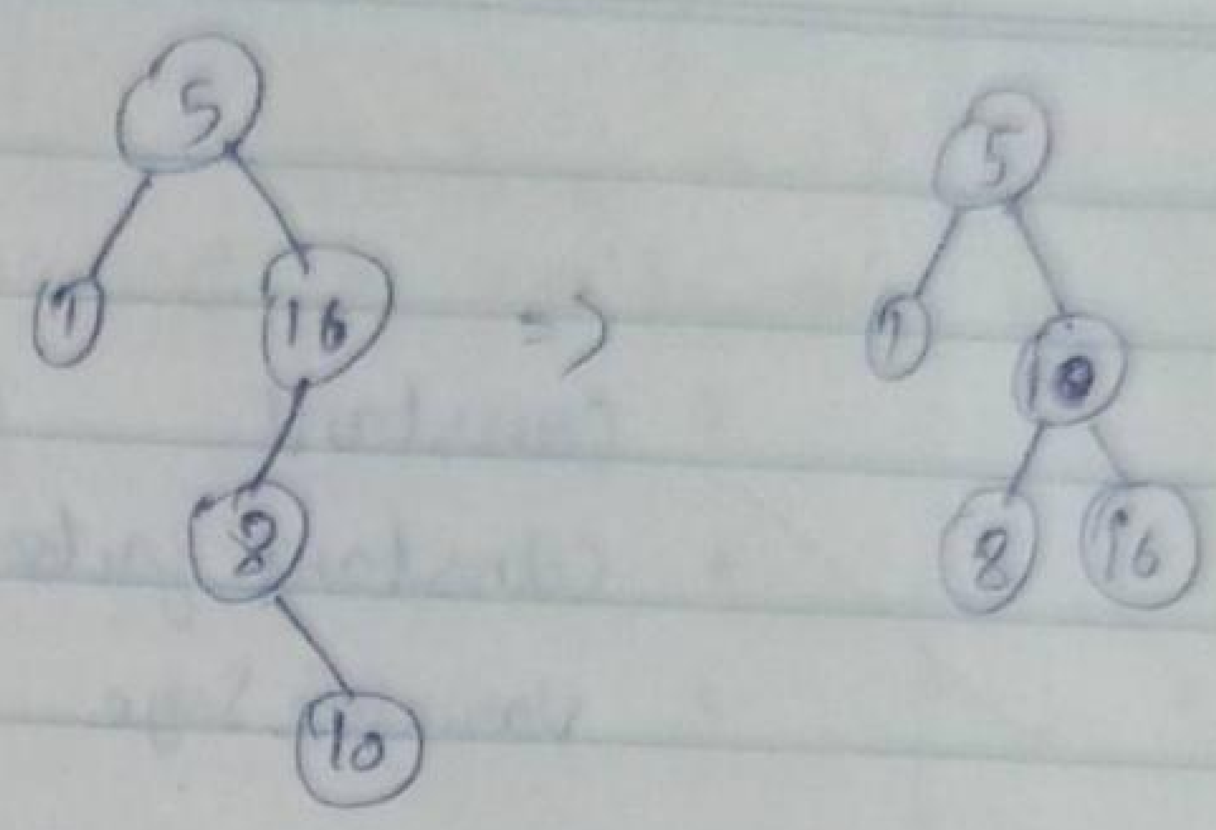
2) 

3) 

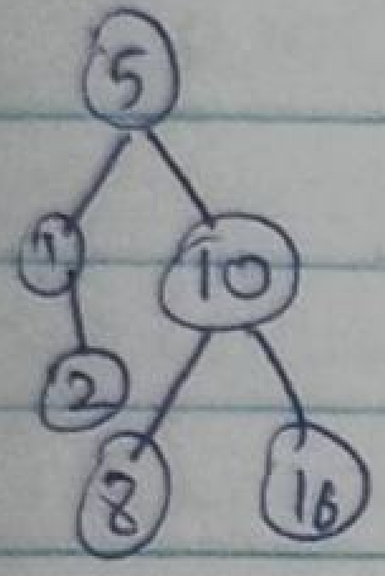
4)


5
7
16
8

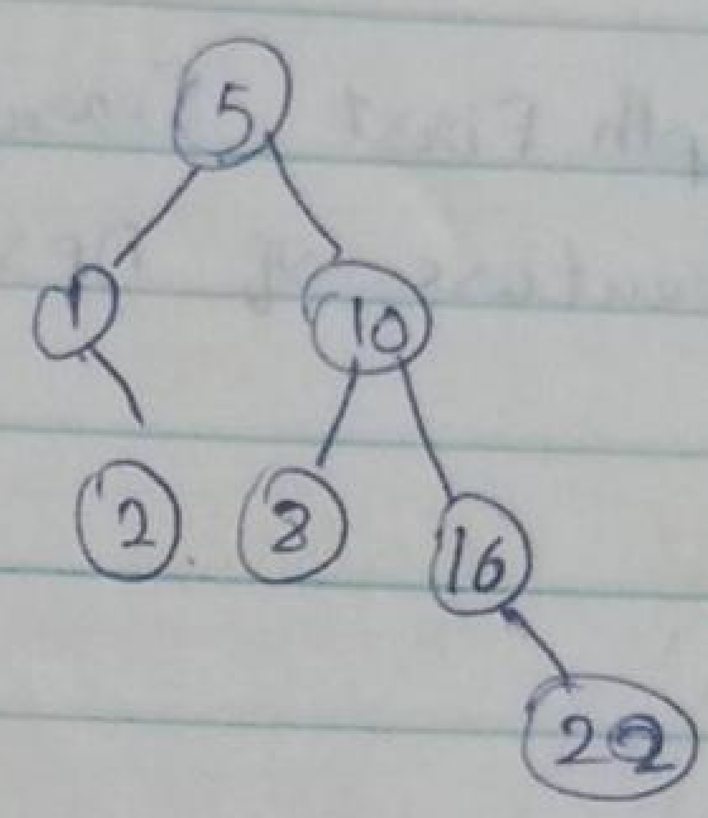
5)


5
7
16
8
10
=>
5
7
10
8
16

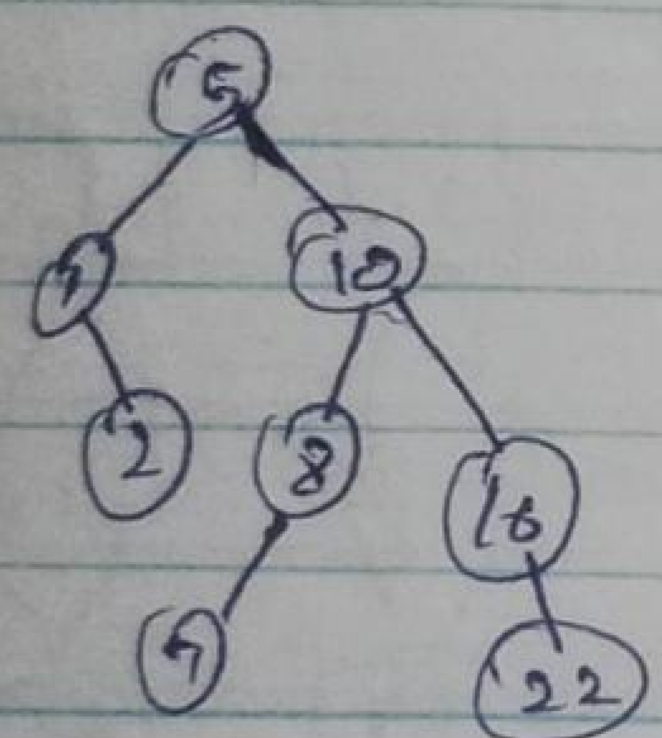
6)


5
7
10
2
8
16

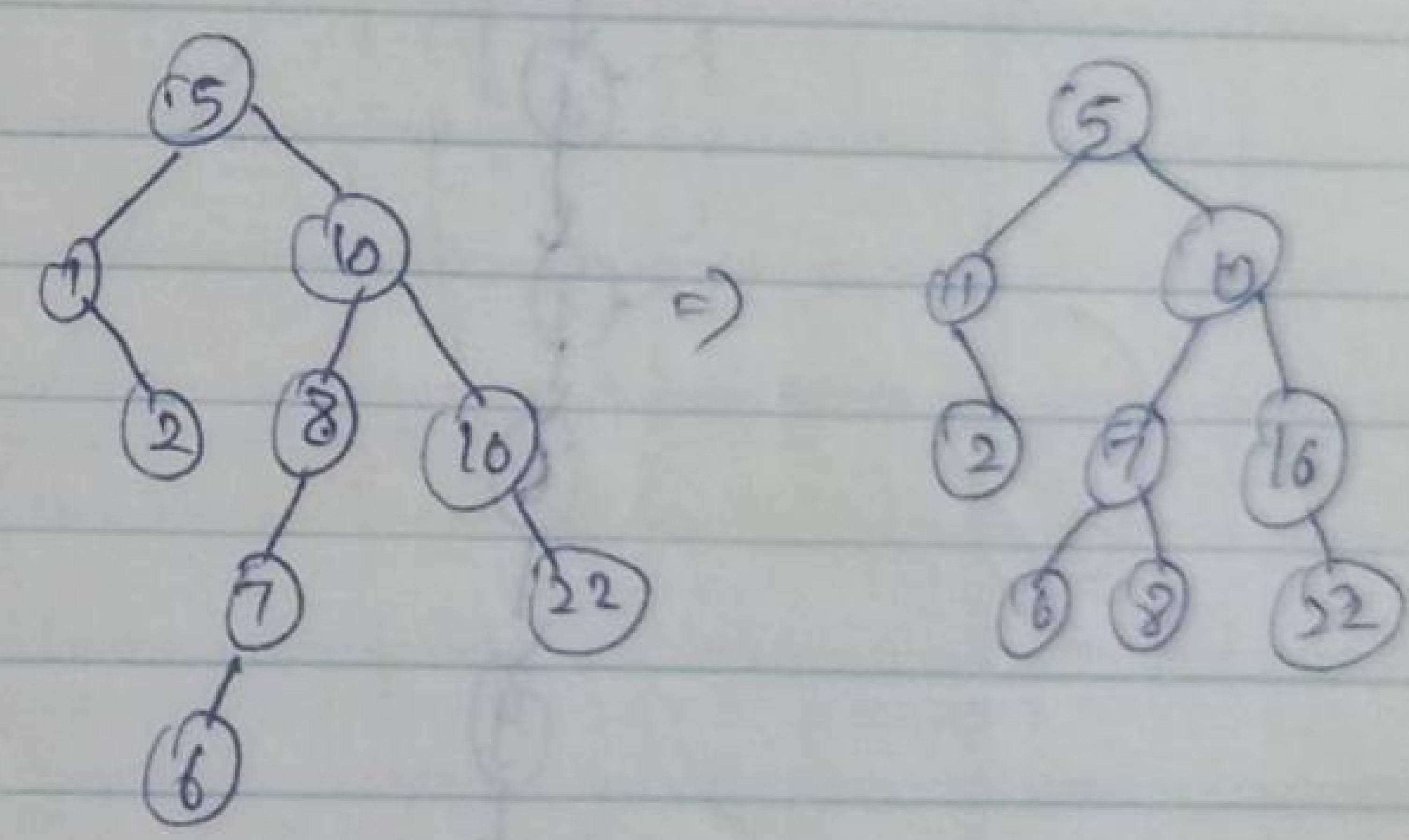
7):


5
7
10
2
8
16
22

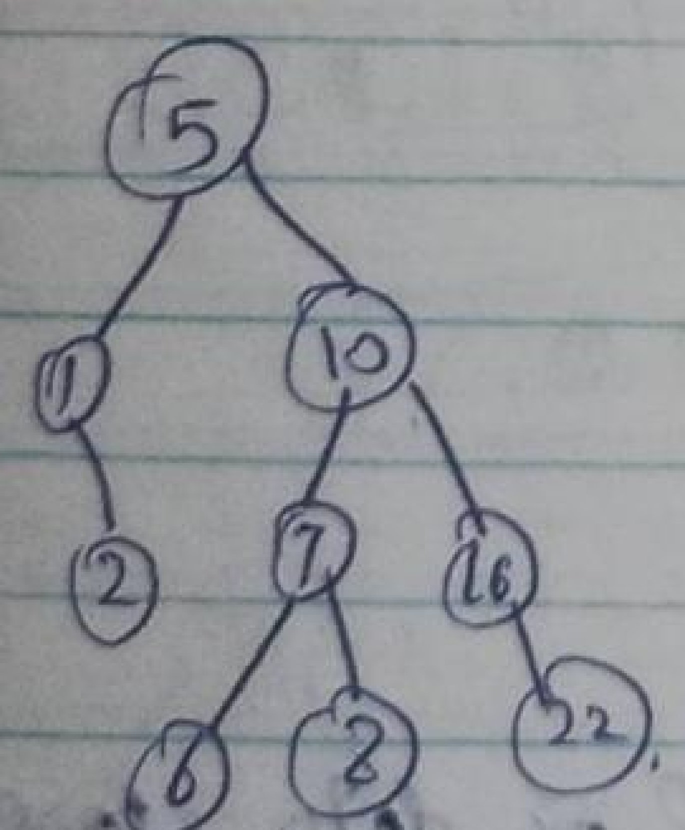
8)


5
7
10
2
8
16
7
22

9)


5
7
10
2
8
16
7
22
6
=>
5
7
10
2
7
16
6
8
22

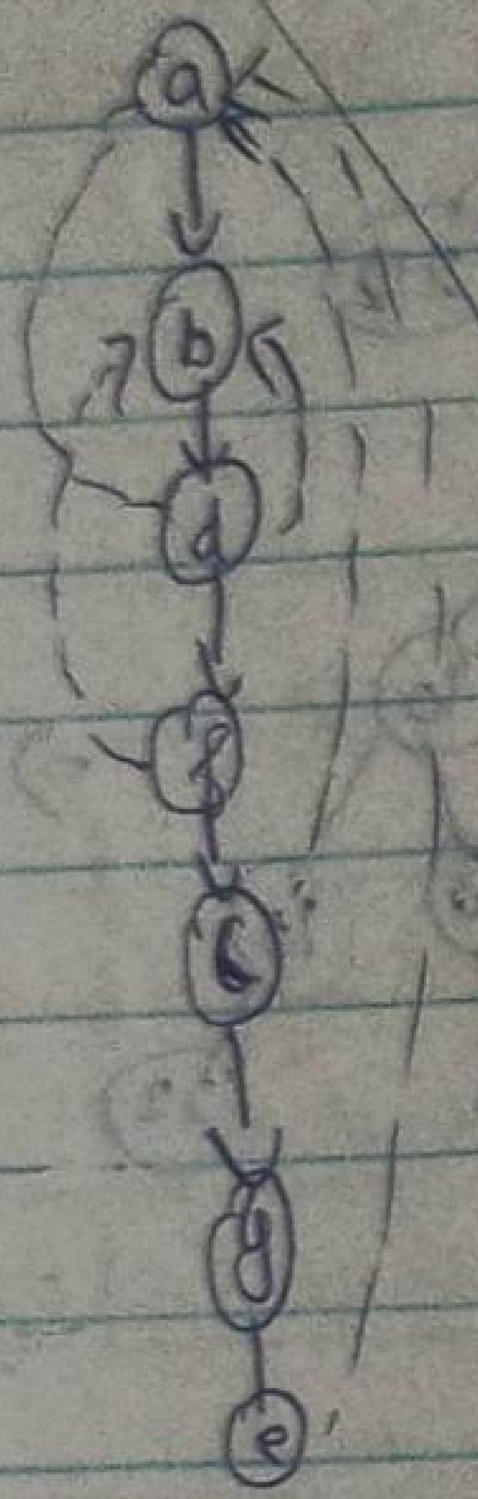
10)


5
7
10
2
7
16
6
8
22

Part-A.

5. Variations of Decrease-and-conquer approach
   * constant (Decrease by constant)
   * constant factor (Decrease by constant factor)
   * variable size (Decrease by variable size)

6. DFS - Depth First Search.
   Data Structure of DFS is Stack.



9. Advantages of balanced search tree.
   - Simple
   - Storage of Data.
   - Easy sorting of element
   - Easy tracking and retrival of data stored
   - Easy modify and storage.

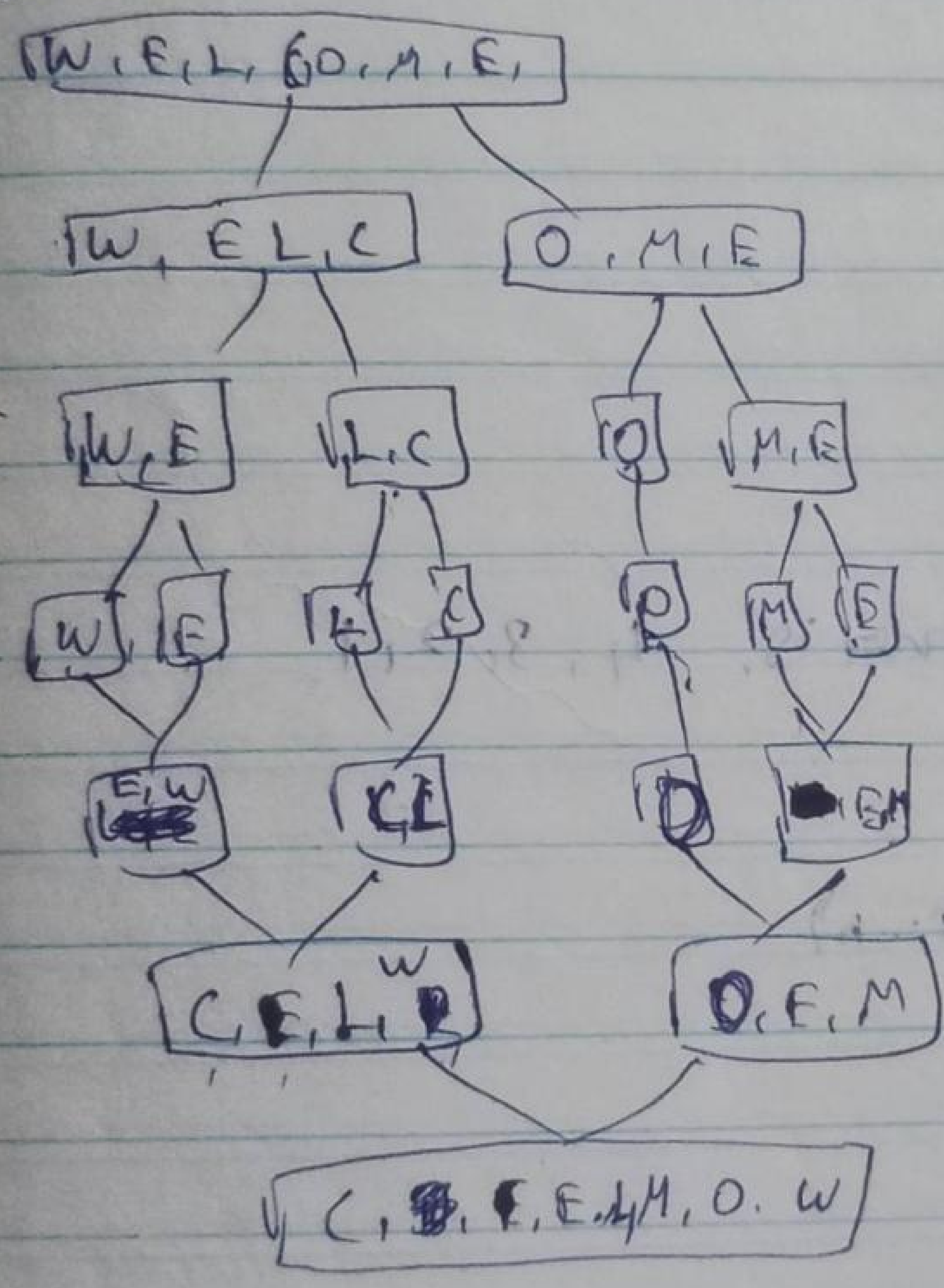
3. Height of a Binary tree = height of Right Subtree - height of ~~left~~ left Subtree.
   (or)
   ~~Total~~ no. of the longest node.

10. Divide-and-conquer algorithm.

This concept is nothing but dividing a big/huge problems into subproblem and then joining the solution of all the subproblem, we can get the result or solution of the big/huge problem. This the ~~the~~ backend or concept behind the term Divide-and-conquer.

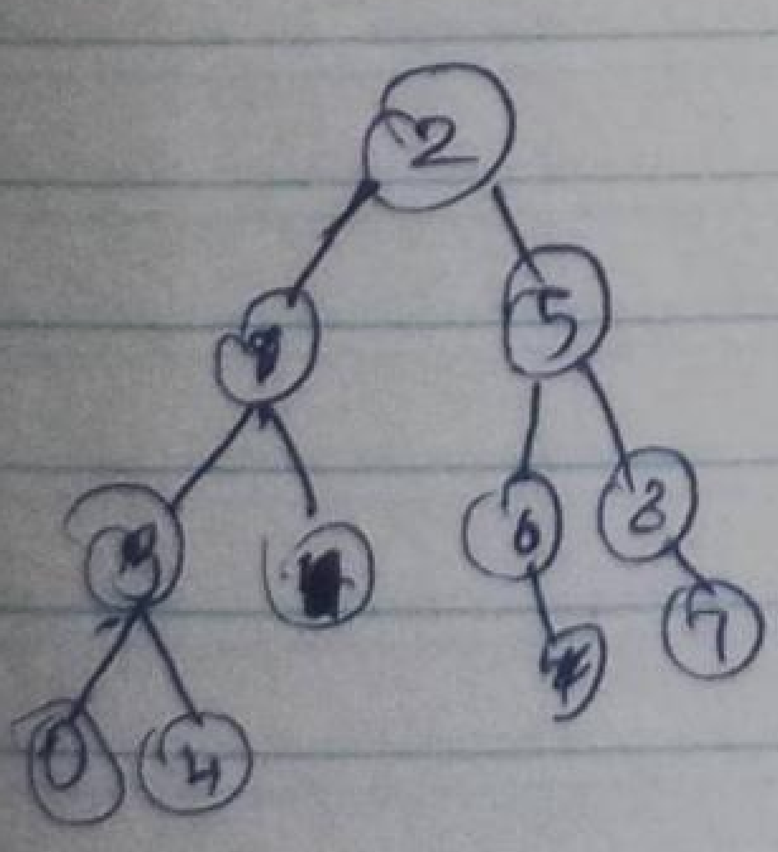
2101⁷, 130 = 2374130.

2. Merge sort.



=> C, E, E, L, M, O, W

4.



~~L, R, Ro~~

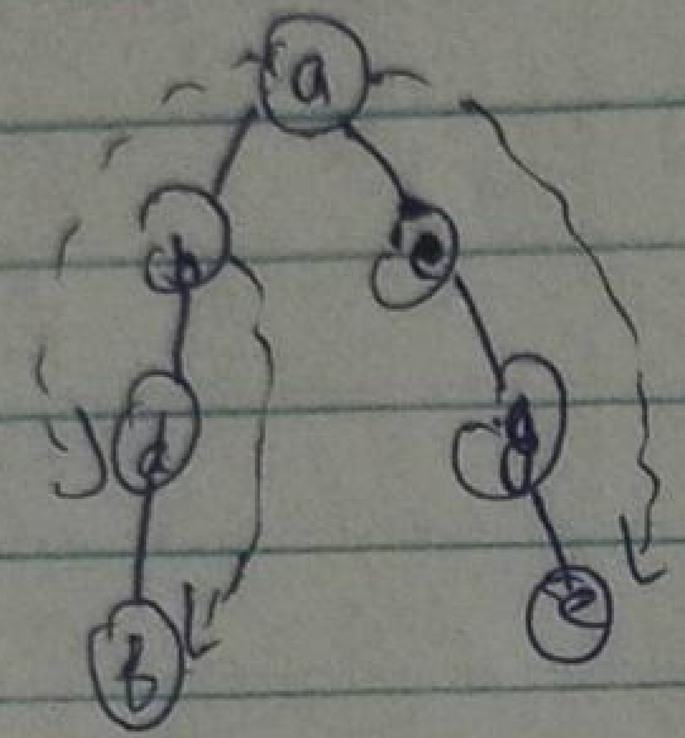
Pre Ro, L, R

In L, Ro, R

Po L, R, Ro

1. Master Theorem.

6. DFS.



7. ~~Foto~~ Topological order is , 4, 3, 2, 1.

8. Algorith Uniqueness (A.(1 ... n)

{

for (i=1 to n-1)