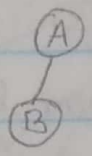
Name: Gaurav Taneja
ID: 1001955801
Assignment -02

Assigment 2.

Let us consider a binary tree of level 3 or 0
So the root node will be the leaf node.
where as if the level of binary tree is one there is a possibility it can have one or more leaf node
Thus we can conclude that least number of leaf nodes in a binary tree is one.

Eg:-

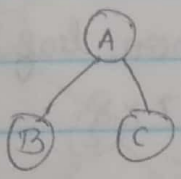


No of nodes = 2.
leaf node = 1

Formula for greatest number of leaf nodes:

For even $= \frac{n}{2}$ For odd $= \frac{n}{2} + 1$

Eg:-



number of nodes = 3

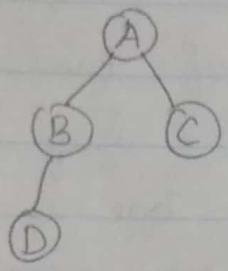
As the number of nodes are odd

$\therefore$ leaf nodes $= \frac{n+1}{2}$ $\therefore \frac{3+1}{2}$

$= 2.$

Two leaf nodes which is correct as above tree has. two leaf nodes B and C

let us consider



number of nodes are four

leaf nodes $= \frac{n}{2}$

$= \frac{4}{2} = 2$

Two leaf nodes which are D and C for above example.

2) In full binary tree, every non-leaf nodes has two child nodes; The total number of nodes in a binary tree is calculated by taking sum of non leaf nodes and leaf nodes.

Eg:- height = 3

Total nos of nodes = non leaf node + leaf node

$= 7 + 8$

$= 15$

Let us consider n as number of leaf node thus number of non leaf node are $(n-1)$

Total nodes = non leaf + leaf node

$= (n-1) + n$

$= 2n - 1$

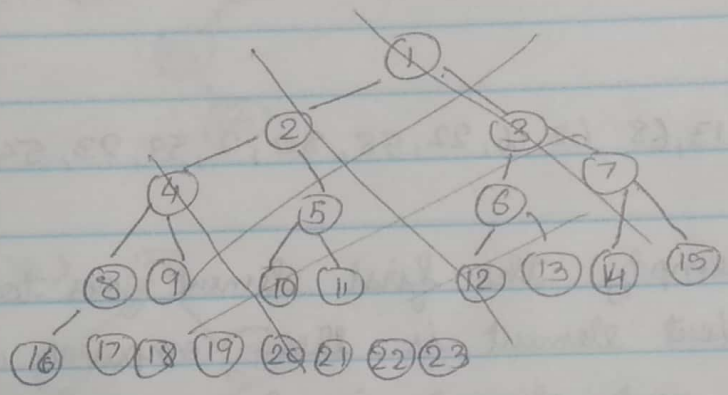
If height of full binary tree is 4 then number of leaf nodes will be 16. Therefore

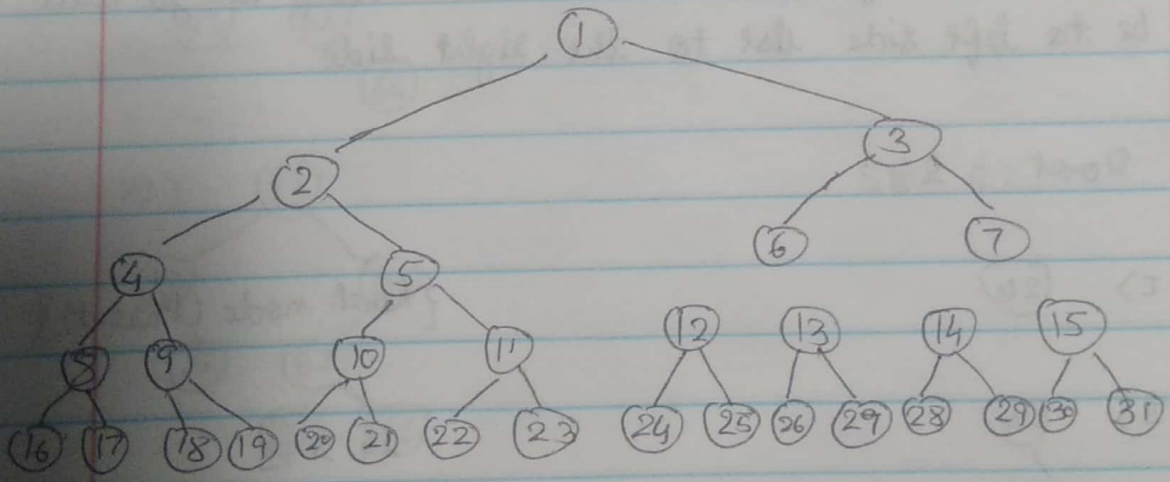
Total nodes = $2n - 1$

$= 2(16) - 1$

$= 31$

Tree with height = 4 will be.

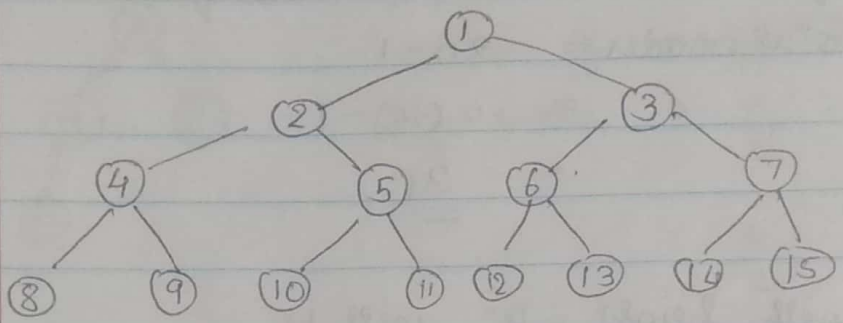




We can calculate leaf nodes from number of nodes.

leaf node = $\frac{n+1}{2}$ where n = number of nodes

Now n = 15 ∴ number of leaf node = $\frac{15+1}{2} = 8$



Q2.)

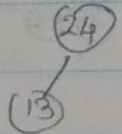
1.) nodes = [24, 13, 68, 62, 66, 22, 58, 80, 2, 59, 23, 54, 1, 15, 95]

If tree is empty the first element goes to the root. Next element is then compared with root node, if next element is less, it go will be to left side else to the right side.

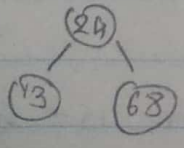
Root = 24

I.) 

{Root node (Parent)}

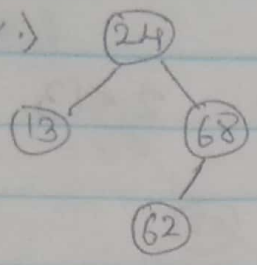
II.) 

24 > 13

III.) 

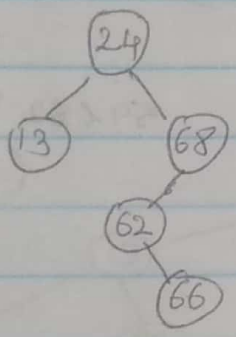
24 < 68

IV.)



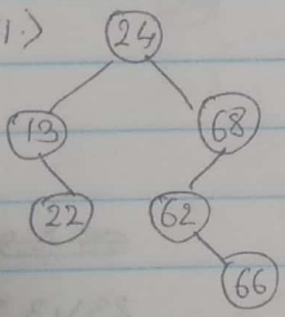
$24 < 62$ & $62 < 68$

V.)



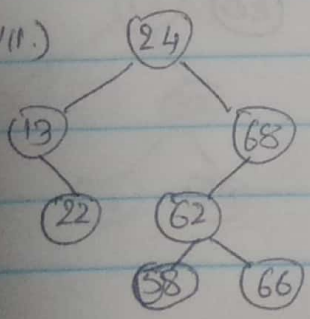
$66 < 68$ & $66 > 62$.

VI.)



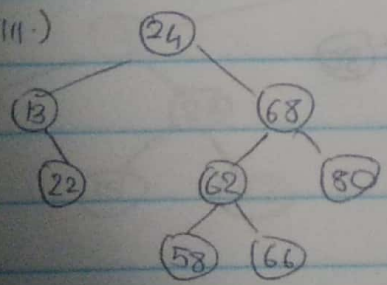
$24 > 22$ & $13 > 22$.

VII.)



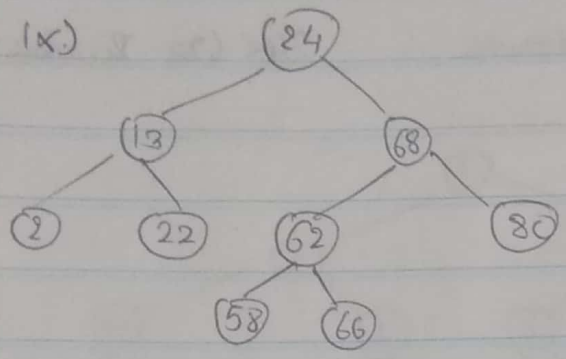
$58 < 62$.

VIII.)



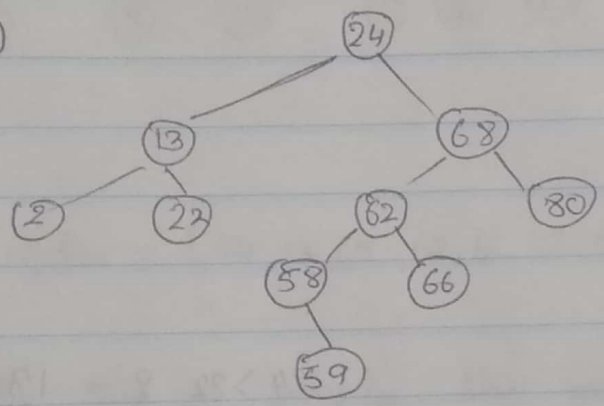
$68 < 80$

IX)



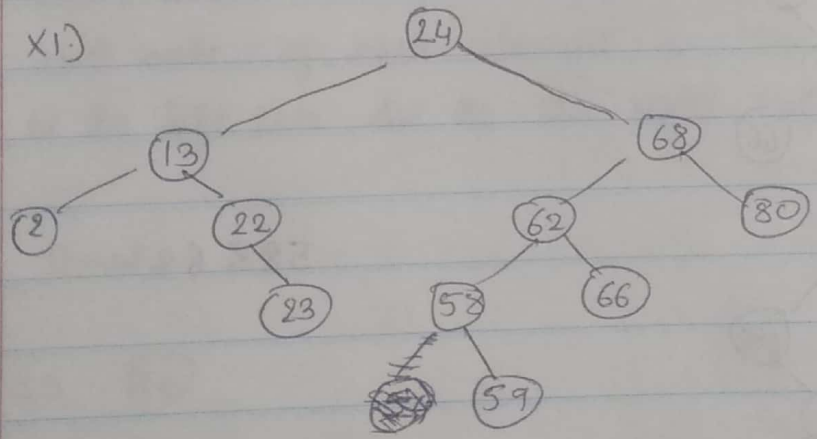
2 < 13

X.)



59 < 62 & 59 > 58

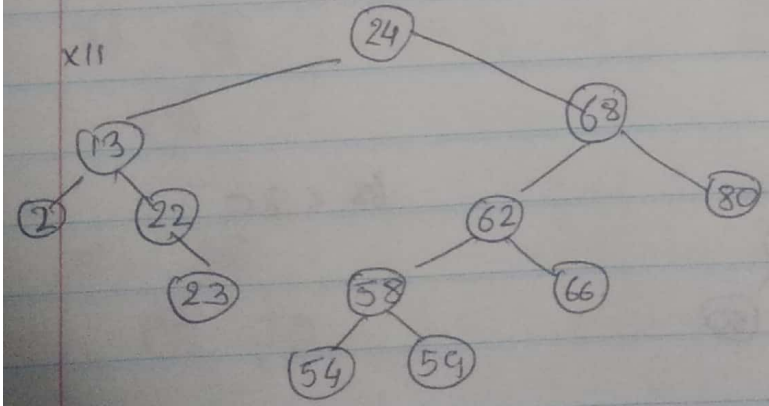
XI.)



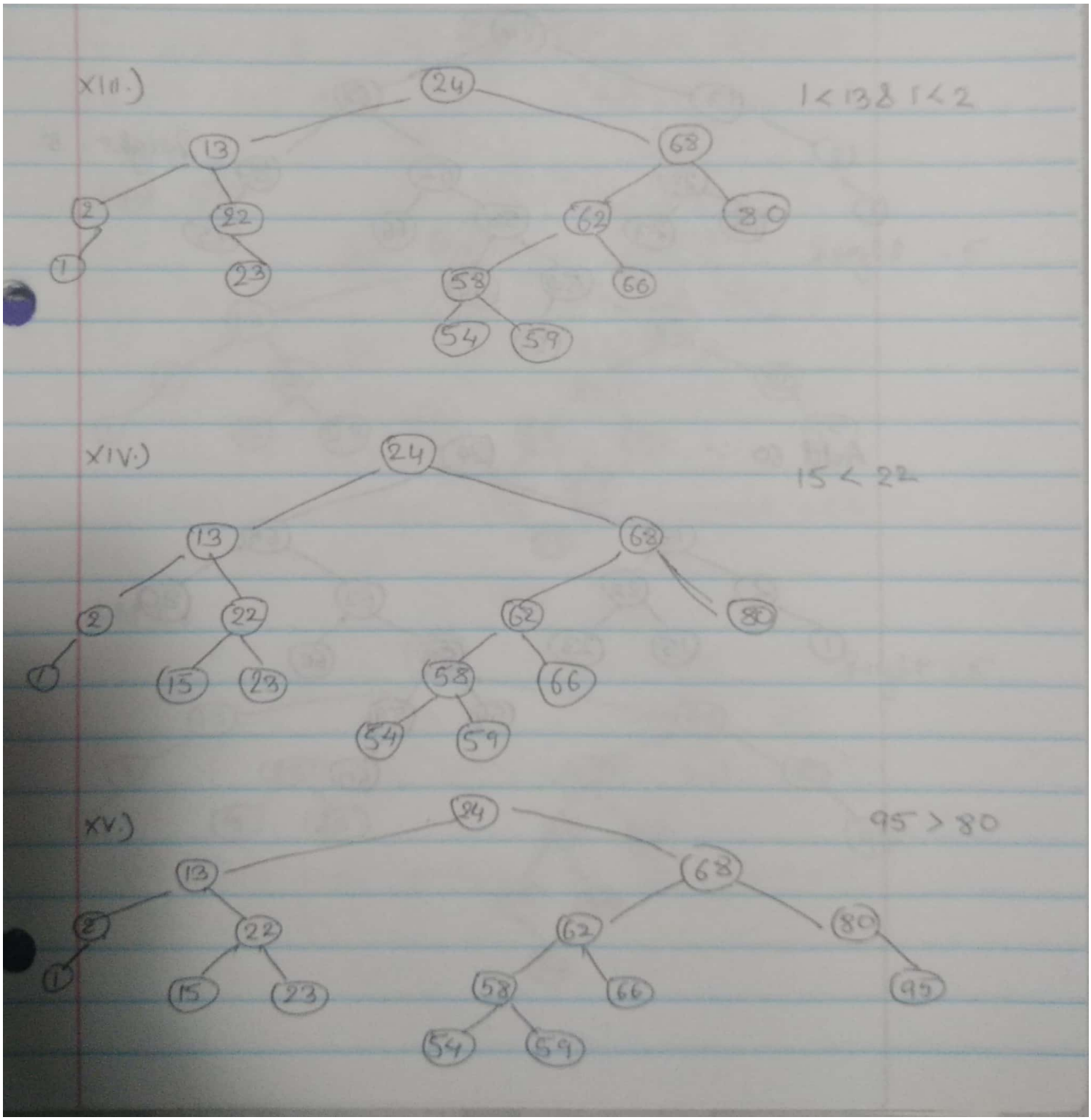
~~58 > 54~~

23 > 13 & 23 > 22

XII



58 > 54

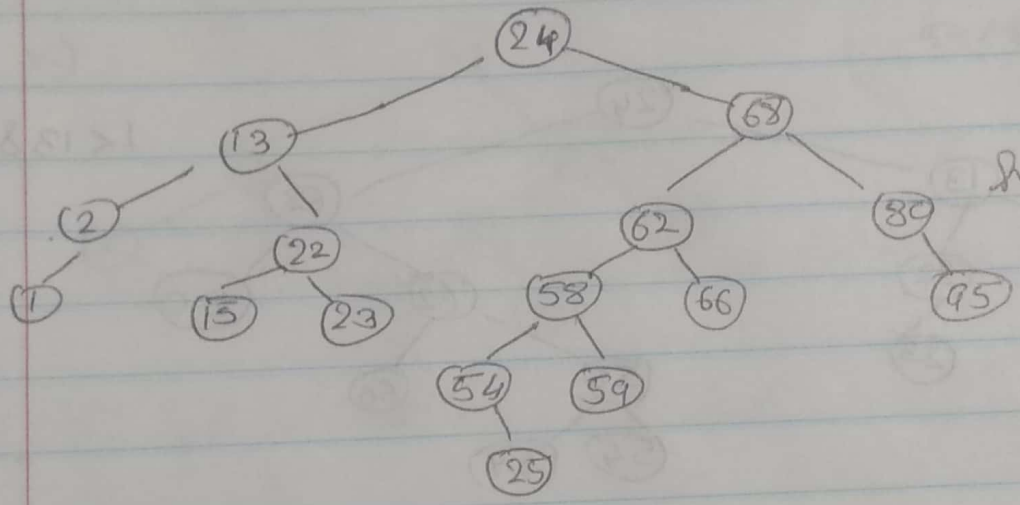


2.) Height = 4

We can increase height of the tree by adding nodes to leaf node.

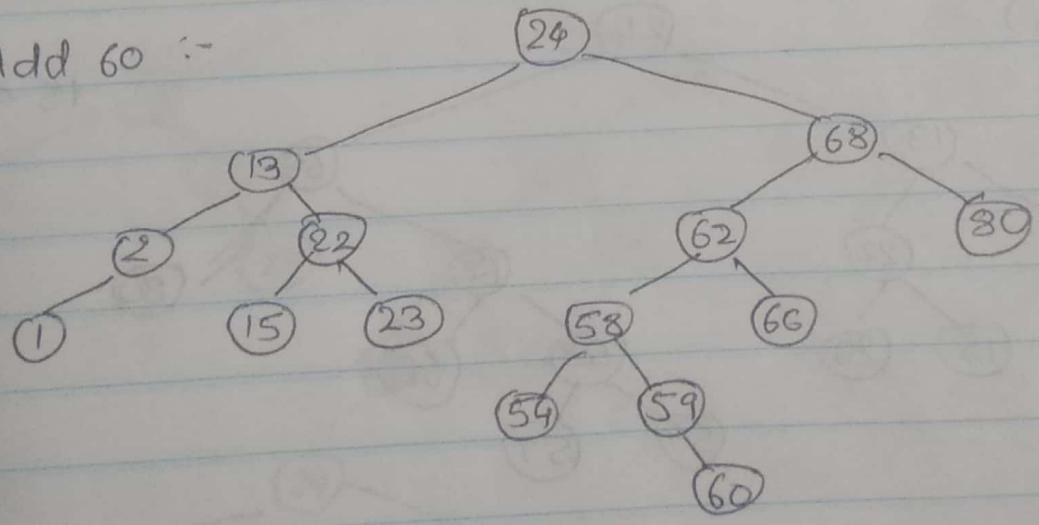
We can add leaf nodes between 24 & 62.

Such as 25, 60, 61, 55 & 56

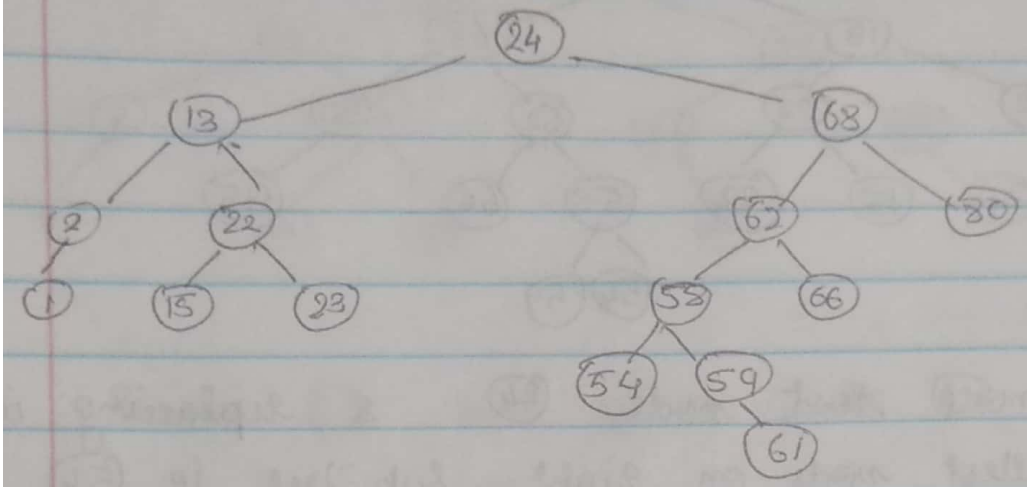


Height = 5

Add 60 :-

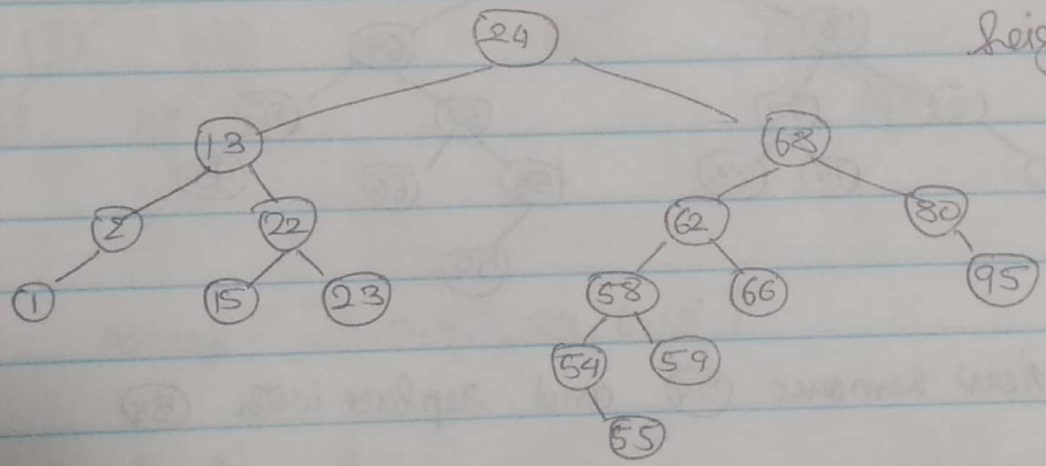


Add 61



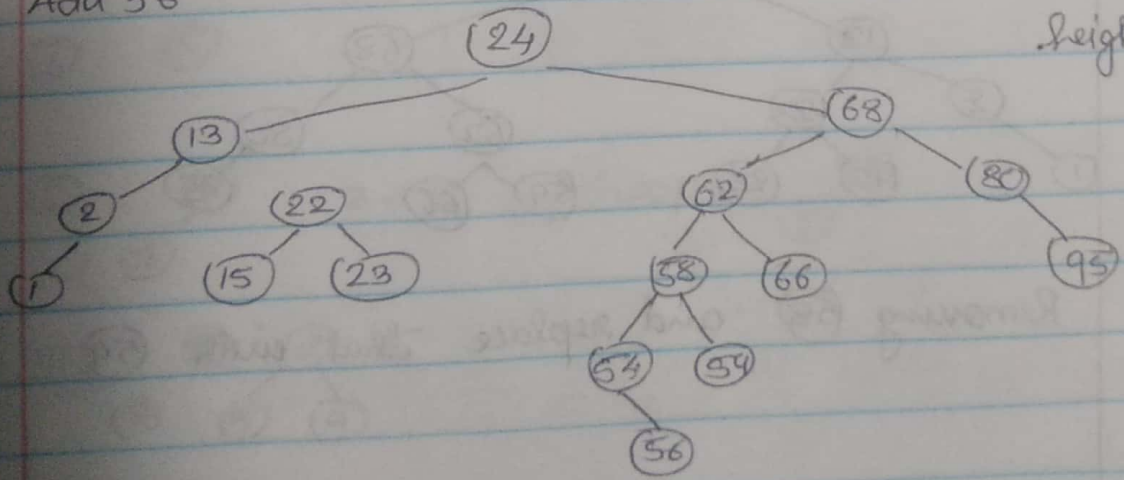
height = 5

Add 55



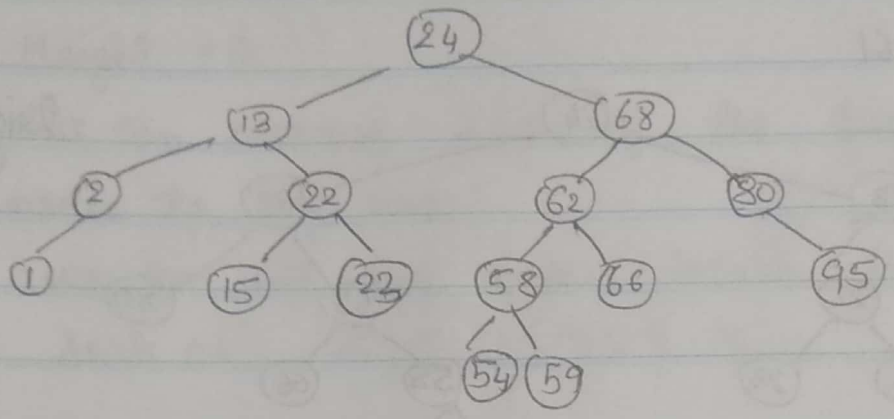
height = 5

Add 56

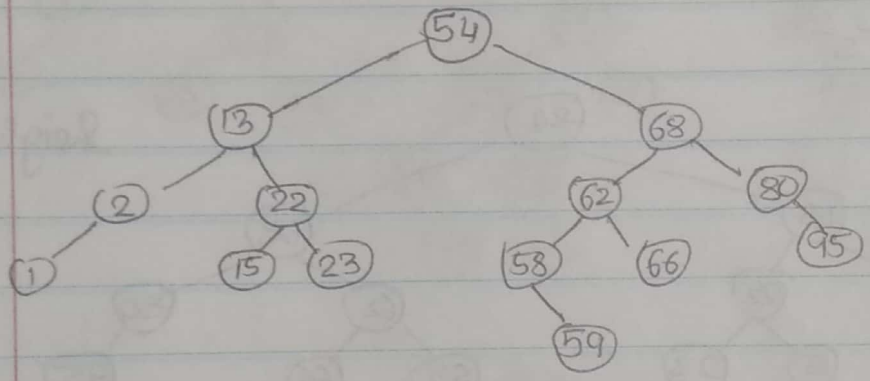


height = 5

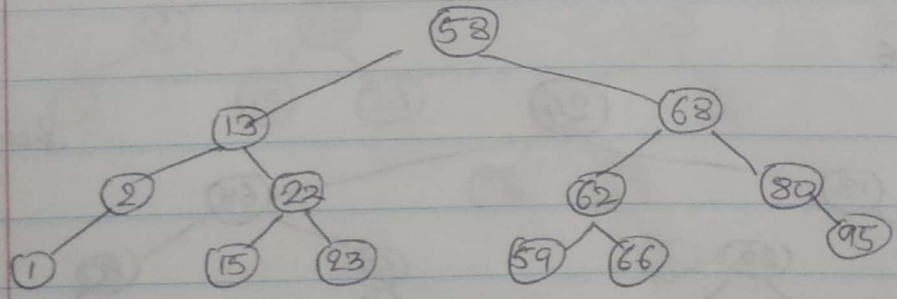
3)



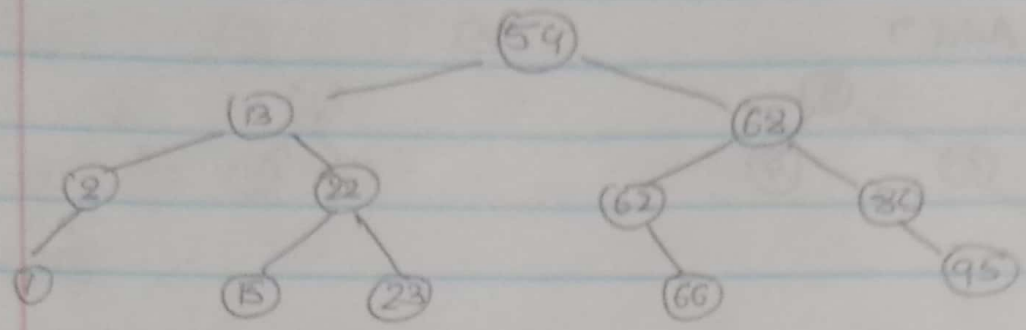
Remove root node (24) & replacing with smallest node on right sub-tree ie (54)



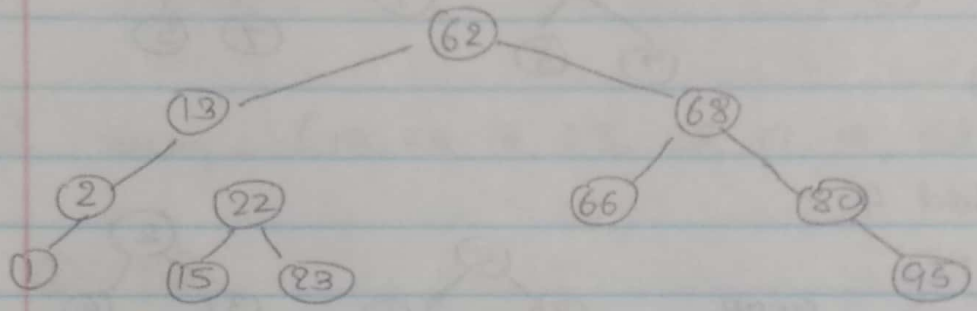
Now remove (54) and replace with (58)



Removing (58) and replace that with (59)



Remove (59) and replace with (62)



Q3.) array = [5, 3, 9, 7, 2, 4, 6, 1, 8]

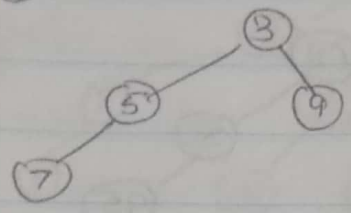
Constructing tree :-

I) 5

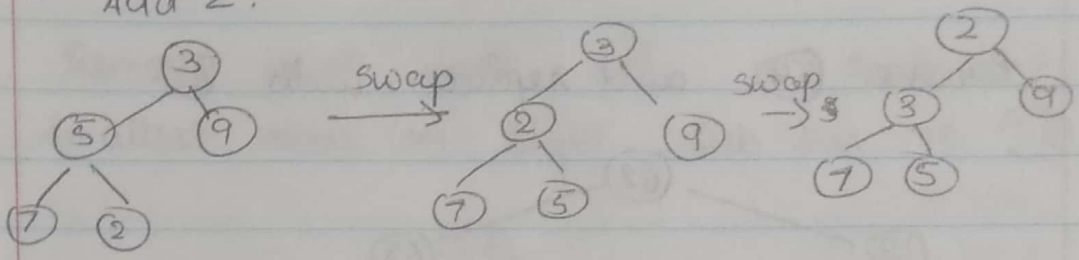
II) 5 — 3 — 3 < 5 swap. 3 — 5

III) 5 — 3 — 3 — 5 — 9

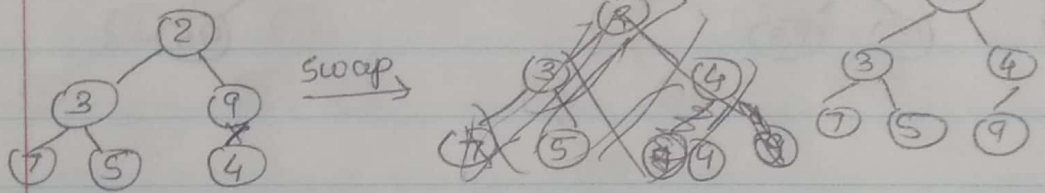
(iv) Add 7



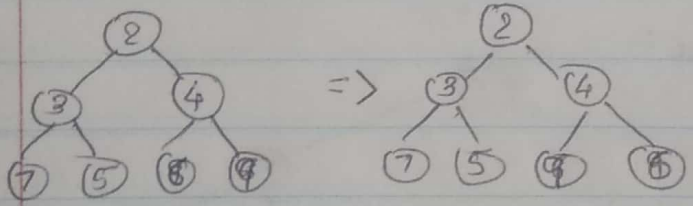
(v) Add 2.



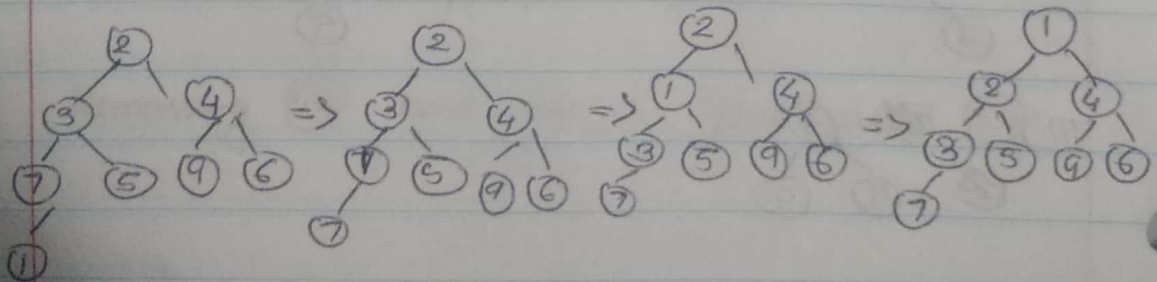
(vi) Add 4

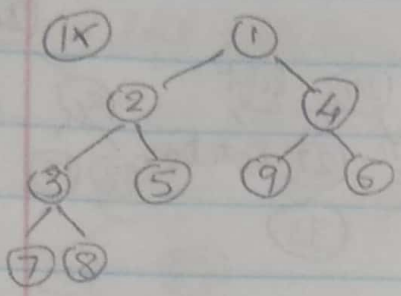


(vii) Add 6.



(viii) Add 1

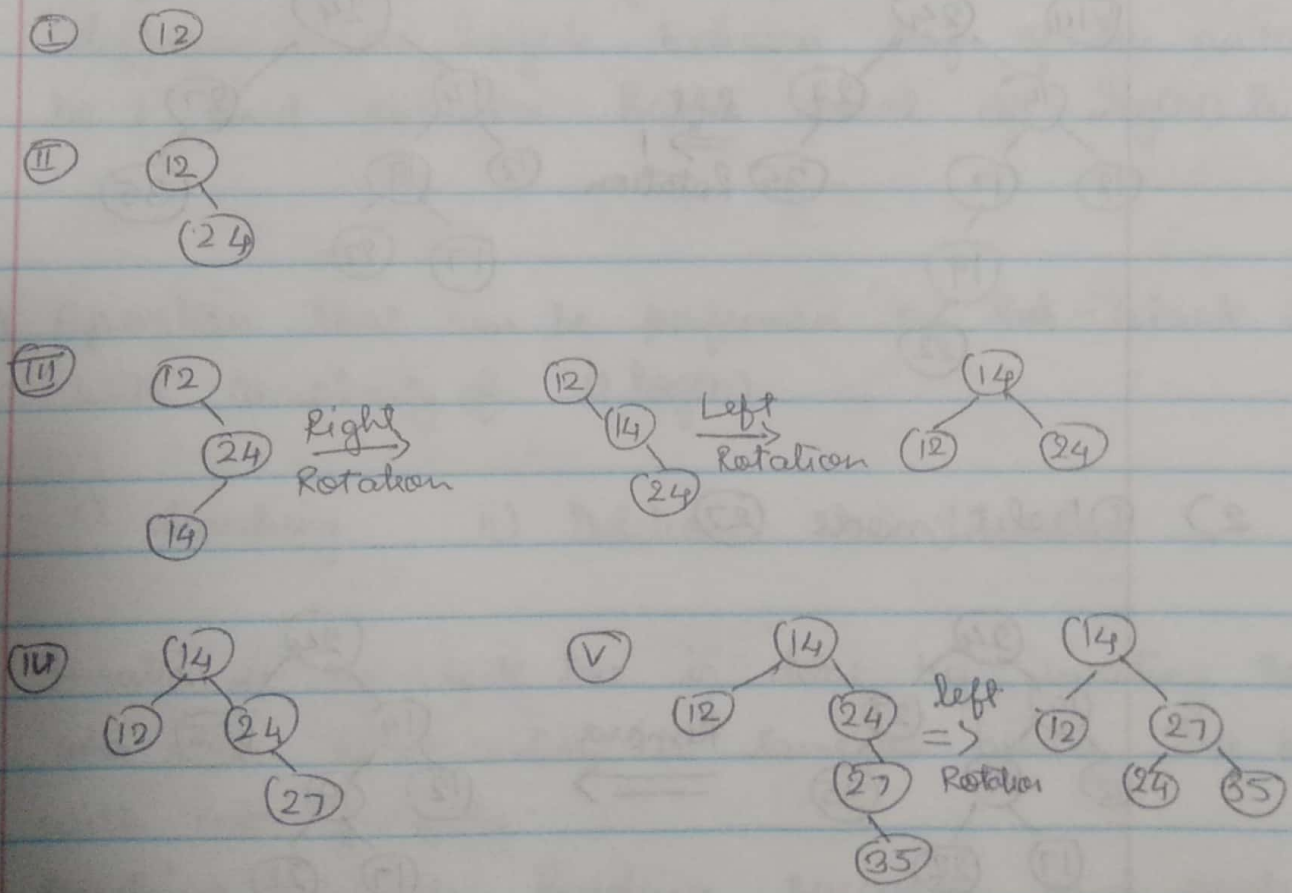


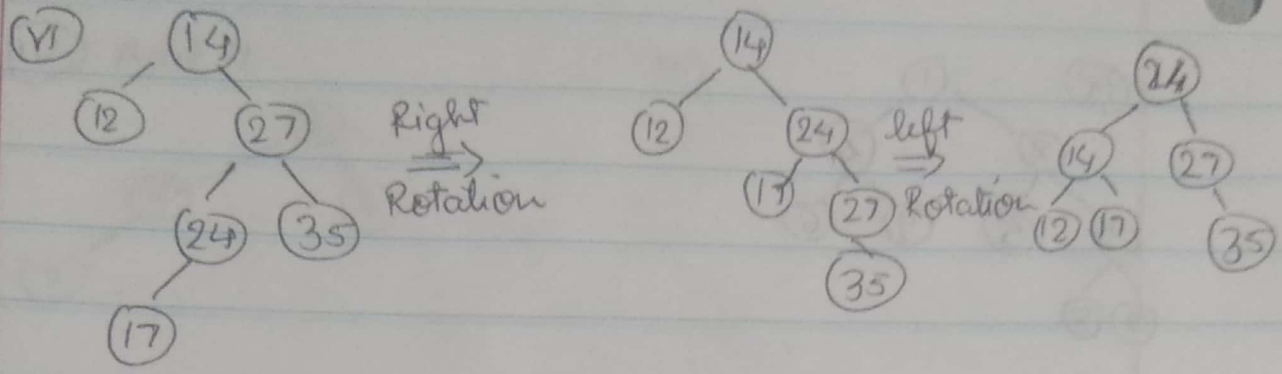


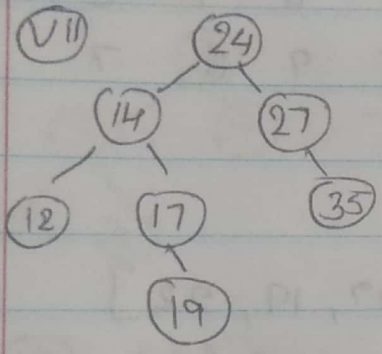
Index	0	1	2	3	4	5	6	7	8	9
Value		1	2	84	3	5	9	6	7	8

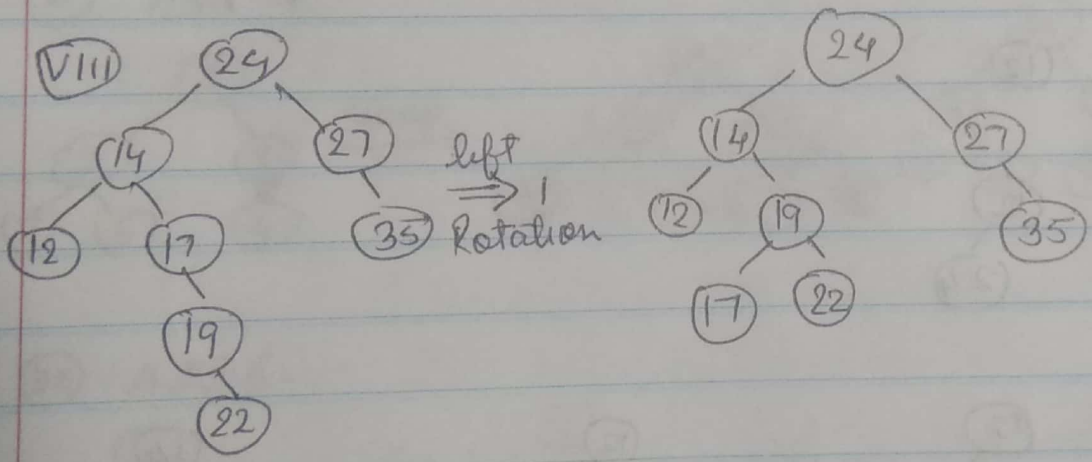
Q4.)

1.) array = [12, 24, 14, 27, 35, 17, 19, 22]

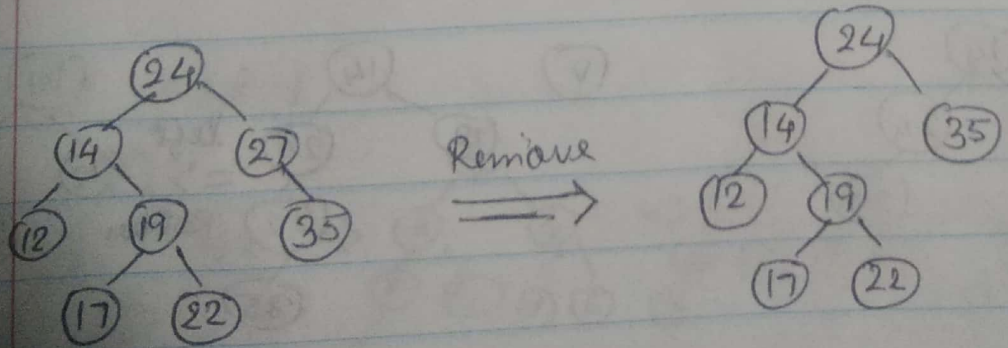




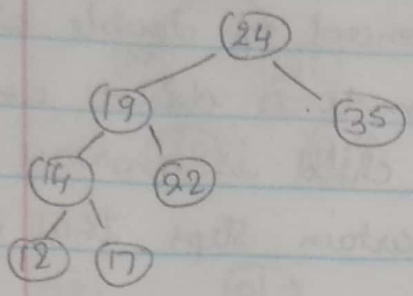




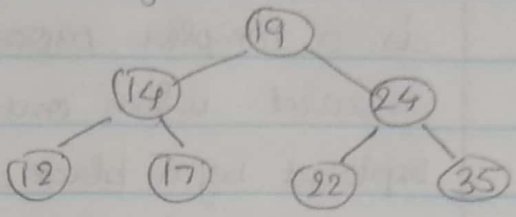
2.) (I) Delet node (27)



left rotation



Right Rotation



3) The balance factor to generate an AVL tree is -1, 0 or 1 for each node. This condition that there should be a difference of -1, 0 or 1 of left sub tree and right subtree. The maximum defference in heights between leaf of tree an must be 1 and AVL tree has a total of log(N) levels.

Q5.)

1) Operation that can be performed on Red-black tree with complexity of O log(n)

i) Searching ii.) Deletion iii.) Insertion.

Insertion in red-black tree is done by inserting red color node. The operation is same as binary tree but with color property.

Searching includes finding successor and predecessor of node.

Deletion process is same as binary tree we end up delete

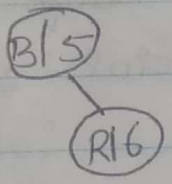
node that has one child or is a leaf node - Deletion is a complex process. The concept of double black is used. when one black node is deleted and replaced by a black child, the child is marked as a double black. There are certain steps that convert double black to single black.

Q5.)

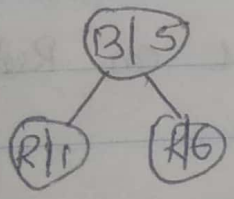
2.) data: 5, 6, 1, 9, 2, 4, 3, 8, 7

I) B|5

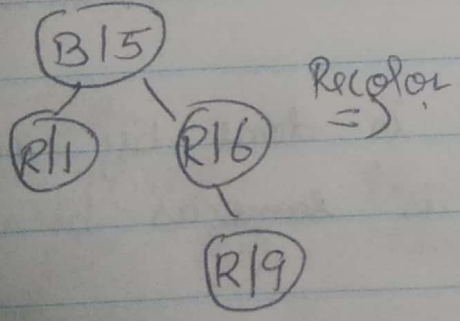
II)



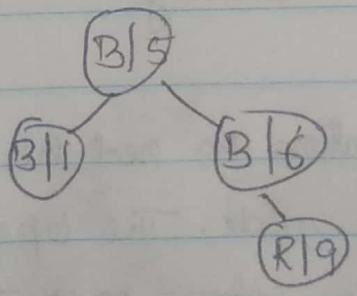
III)



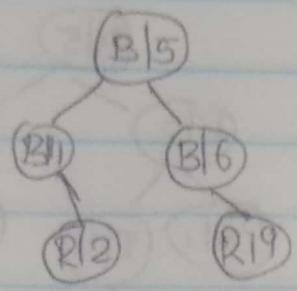
IV)



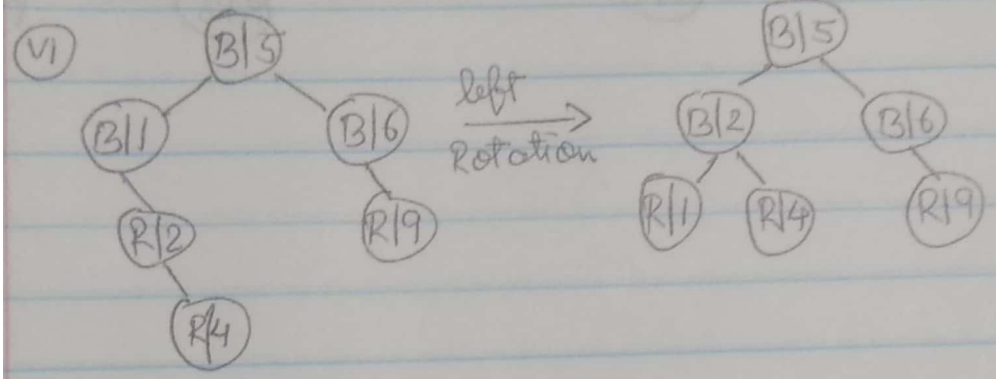
Recolor =>



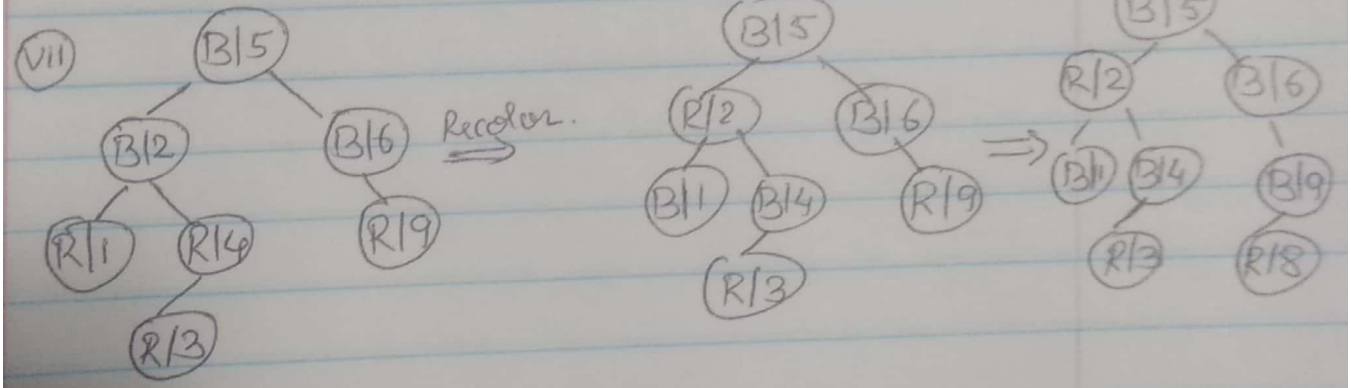
(V)



(VI)



(VII)



(IX) Right Rotate & Recolor.