Alexandria University
Faculty of Engineering

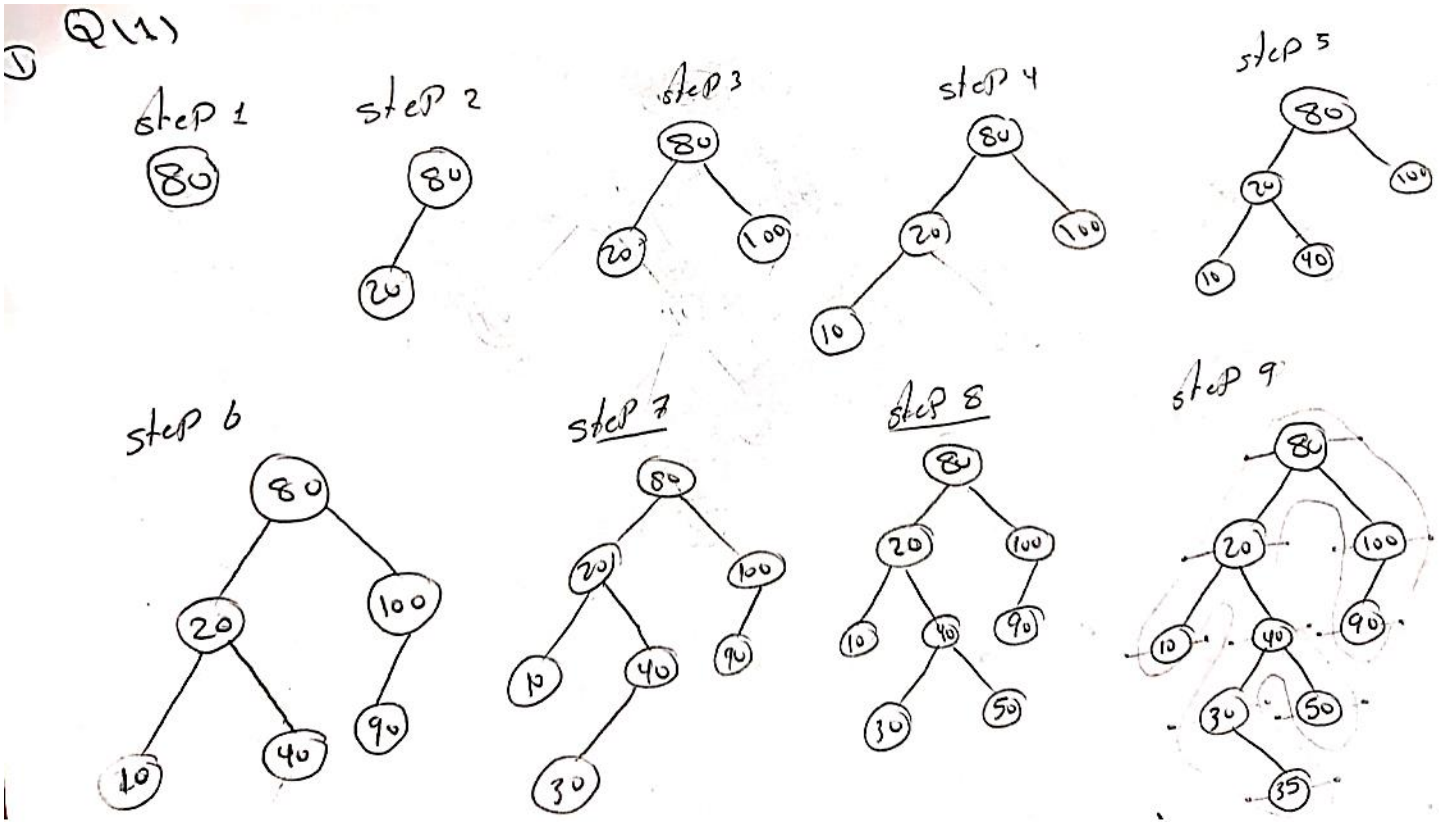
Assignment NO. 3

By:

Ali Mohamed Taima Hummus

Question 1

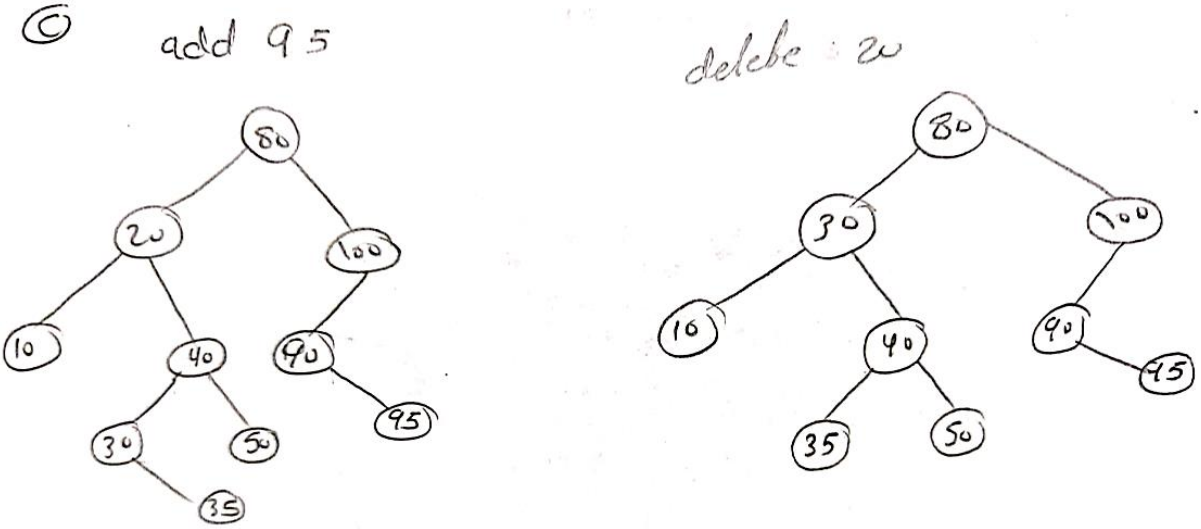
a)



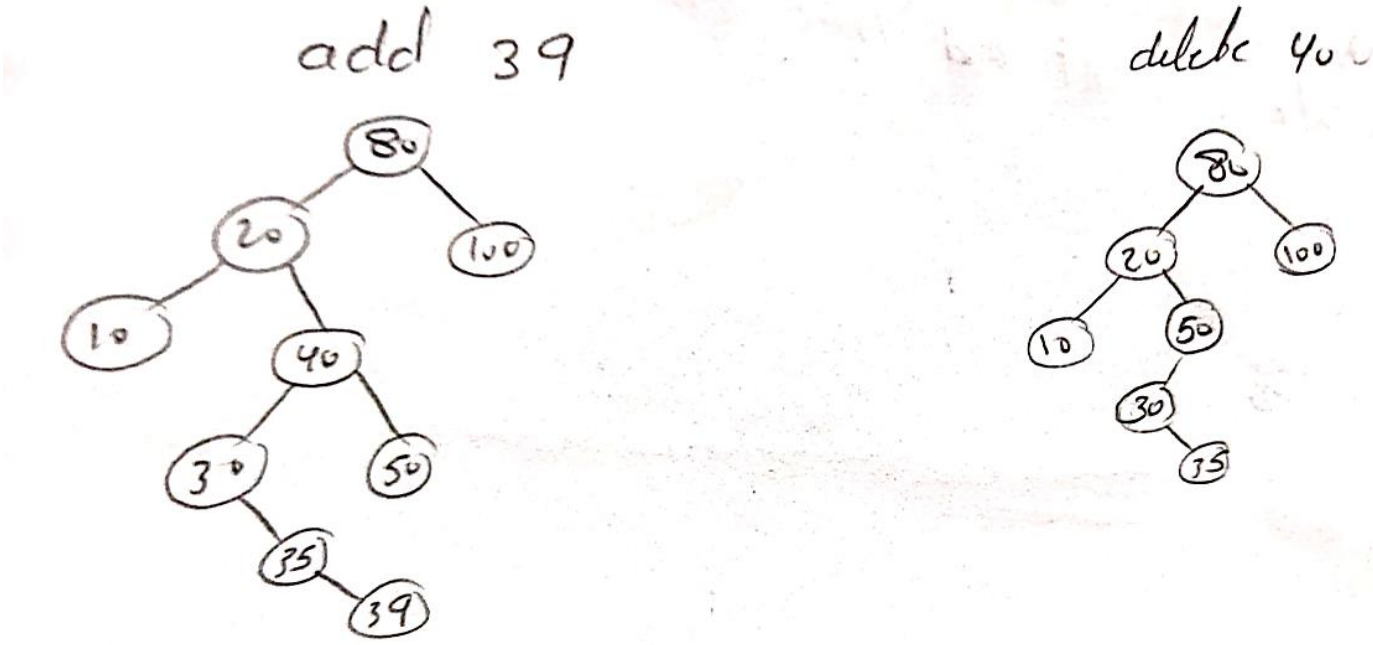
b)

- Pre-order:
  - Visit the root.
  - Visit the left subtree, using preorder.
  - Visit the right subtree, using preorder.
  - (80, 20, 10, 40, 30, 35, 50, 100, 80)
- Postorder:
  - Visit the left subtree, using postorder.
  - Visit the right subtree, using postorder.
  - Visit the root.
  - (10, 35, 30, 50, 40, 20, 90, 100, 80)

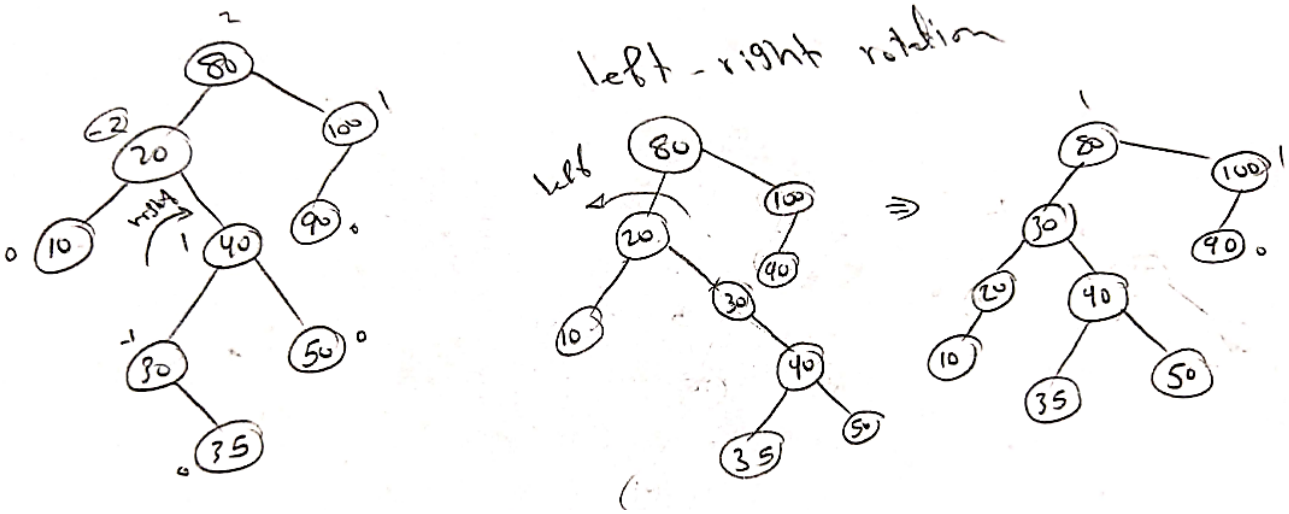
c)



d)



e)



Question 2

a) Preorder: (4, 3, 2, 1, 8, 6, 5, 7, 9)

b)Inorder: (1, 2, 3, 4, 5, 6, 7, 8, 9)

c)Postorder: (1, 2, 3, 5, 7, 6, 9, 8, 4)

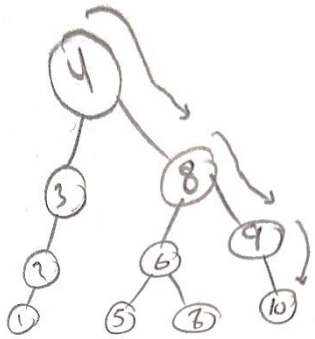
d)

1- check binary tree: **The tree is binary tree** as each node has at most 2 children

2- check BST: **The tree is BST** as all nodes of left subtree are less than the root(or parent) and all nodes of right subtree are greater than the root(or parent)

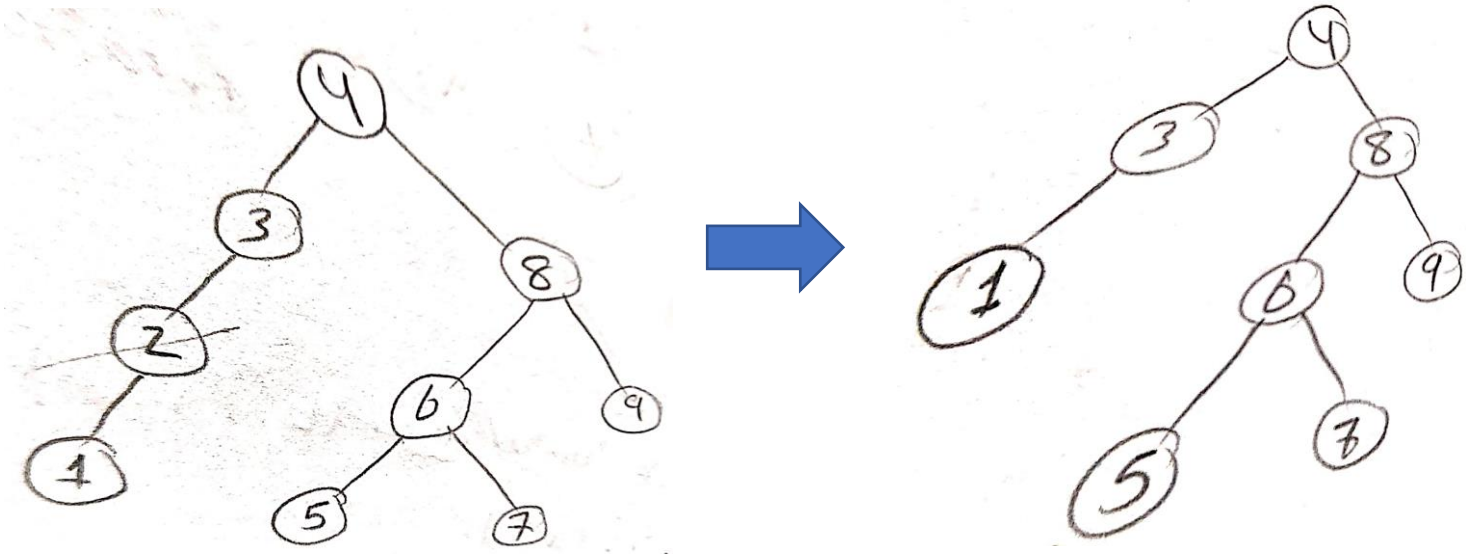
3- check AVL: **The tree is not an AVL tree**, as the **balance factor of node 3 is 2** and the condition of AVL tree that balance factor **must be (-1, 0, 1)** that means not greater than 1 or less than -1

e) **10** is less than **4**, so negate to right subtree until **8**, find that **10** is greater than **8**, so navigate to right subtree until **9**, again wer find 10 is greater than **9**, so navigate to right and there is no more nodes, so put **10** in the right of **9**



---

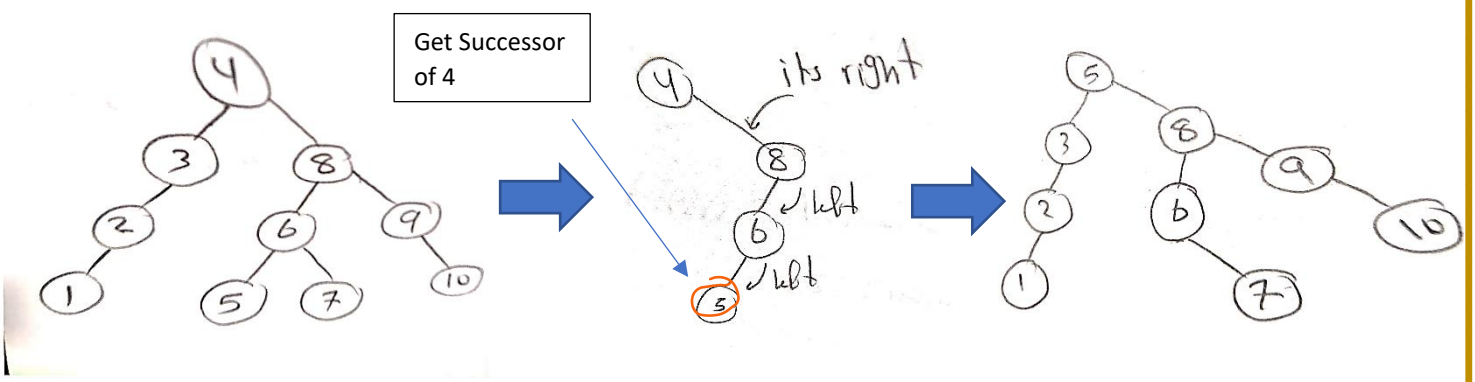
f) We want to delete **2**, so we should search for its successor, but **2** doesn't have right subtree so doesn't have successor, so we will delete **2** and place the nodes smaller than it in its place, so we put **1** and delete **2**



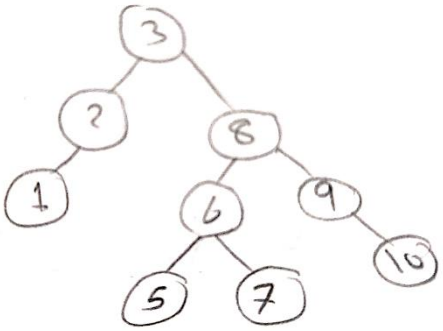
---

g) There are two methods to delete 4

1- To delete **4** we should put its successor in its position and delete the successor from its original position, in our tree the successor of **4** is **5**, so we delete **4** and put **5** instead of it and delete **5** from down

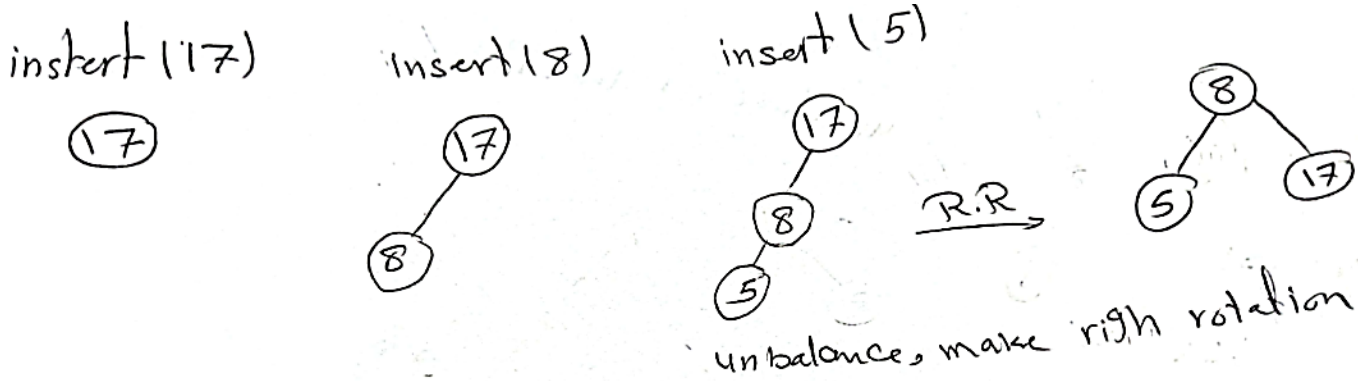


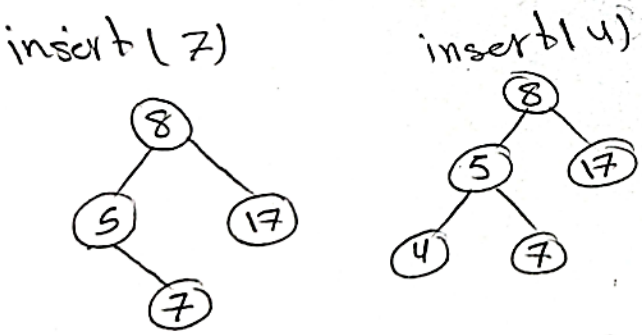
2- To delete **4** we put the node tha less than the node that we want to delete in our tree the node is **3**, so we delete **4** and put **3** instead of it and its subtrees

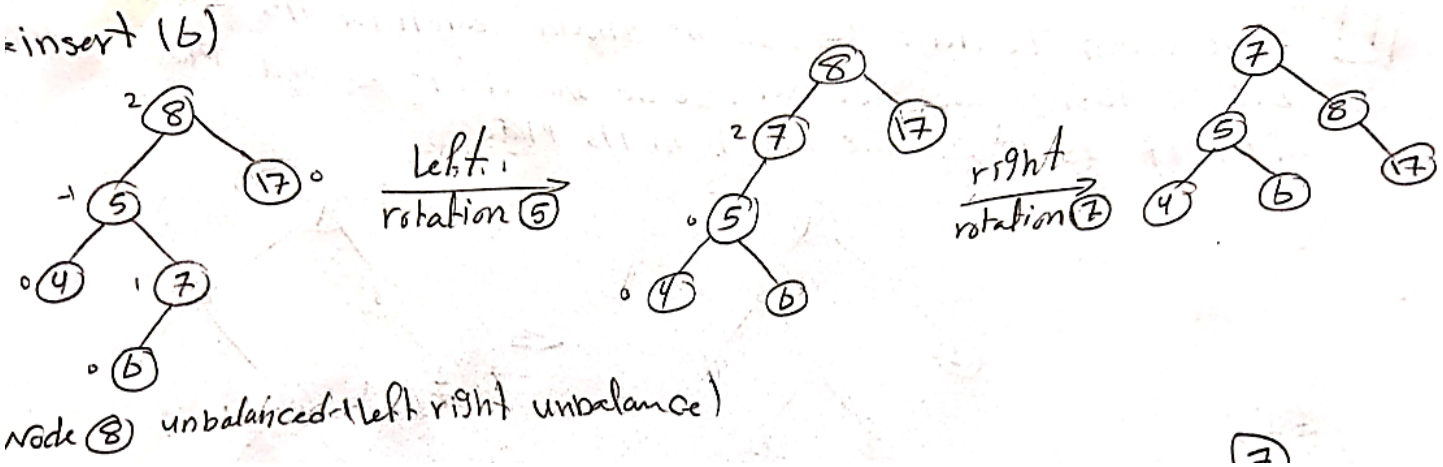


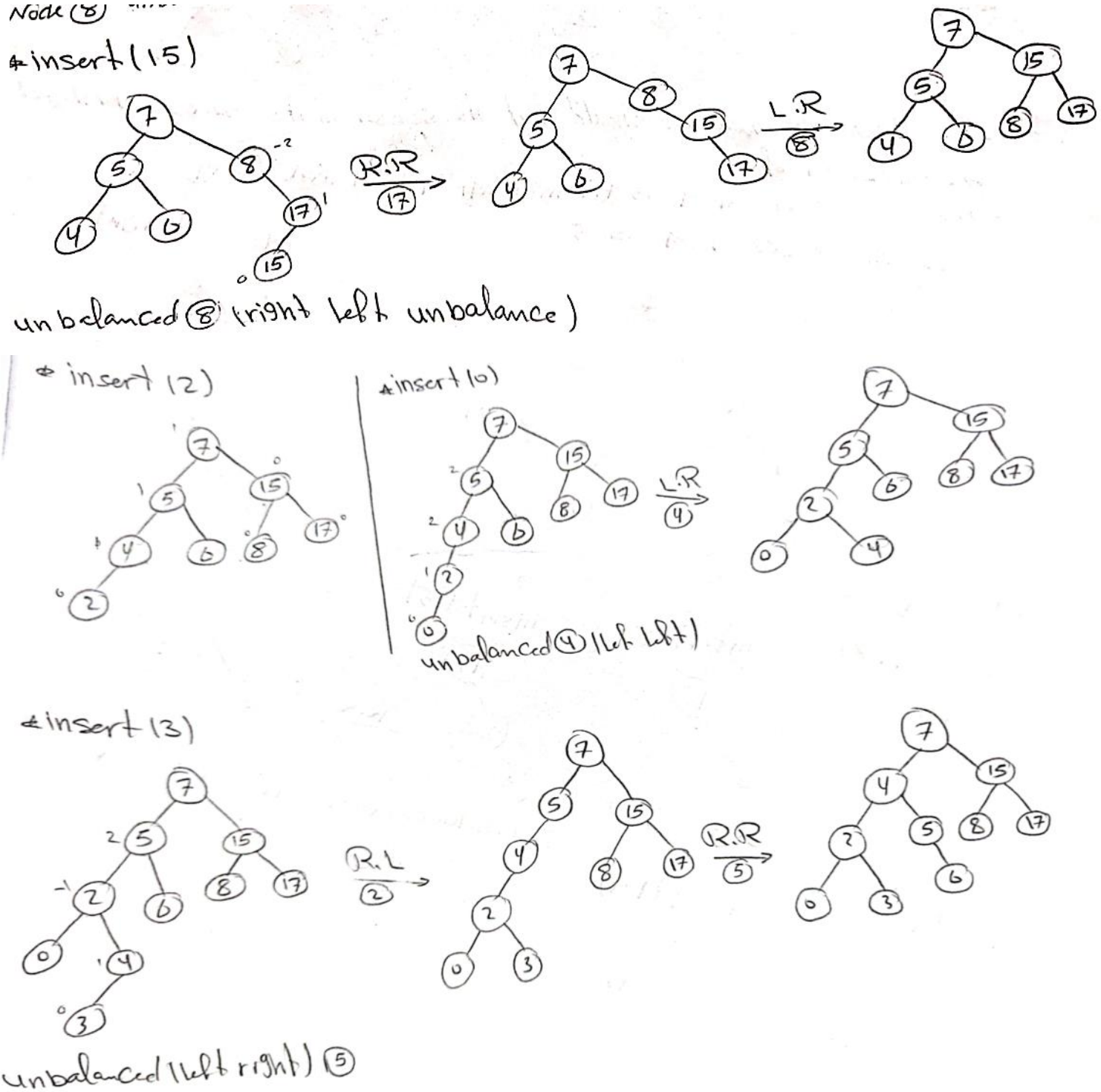
Question 3

a)





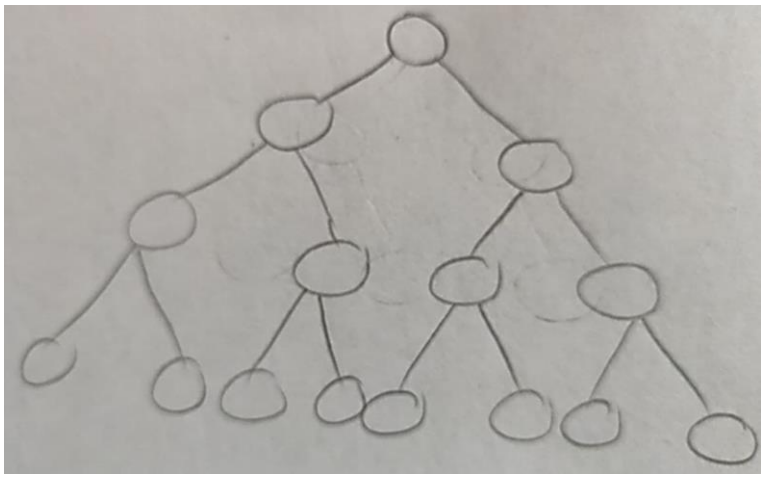




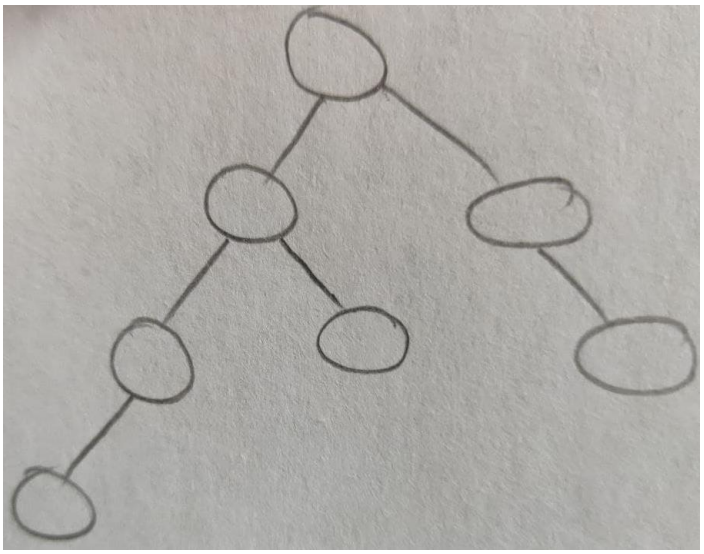
b)

1- If height of AVL tree is h, maximum number of nodes can't exceed $\mathbf{2^{h+1} - 1}$

- As we show in the picture the height is 3 and the number of nodes is 15 and we can't add any other nodes and have the same height (3)



2- Minimum number of nodes in a tree with height h can be represented as:
N(h) = N(h-1) + N(h-2) + 1 for n>2 where N(0) = 1 and N(1) = 2.



- The difference between height of left sub-tree and right sub-tree of any node can't be more than 1.This property gives balance factor which whose value should be from (-1, 0, 1) in case of AVL tree
- In this example I suppose that height is 3 and I want to know the minimum nodes
  - N(3) = N(2) + N(1) + 1
    - N(2) = N(1) + N(0) + 1 = 2 + 1 + 1 = 4
  - N(3) = 4 + 2 + 1 = 7
  - so we can't delete any node because if we delete the balance factor will be more than 1 or less than -1

c)

1- If there are n nodes in AVL tree, minimum height of AVL tree is **Floor($\log_2$(n+1) - 1)**

- Suppose we have 8 nodes **Floor($\log_2$(7+1) - 1)** = 2
- For 2 nodes **Floor($\log_2$(2+1) - 1)** = 1
- For 15 nodes **Floor($\log_2$(15+1) - 1)** = 3
- For 16 nodes **Floor($\log_2$(16+1) - 1)** = 4

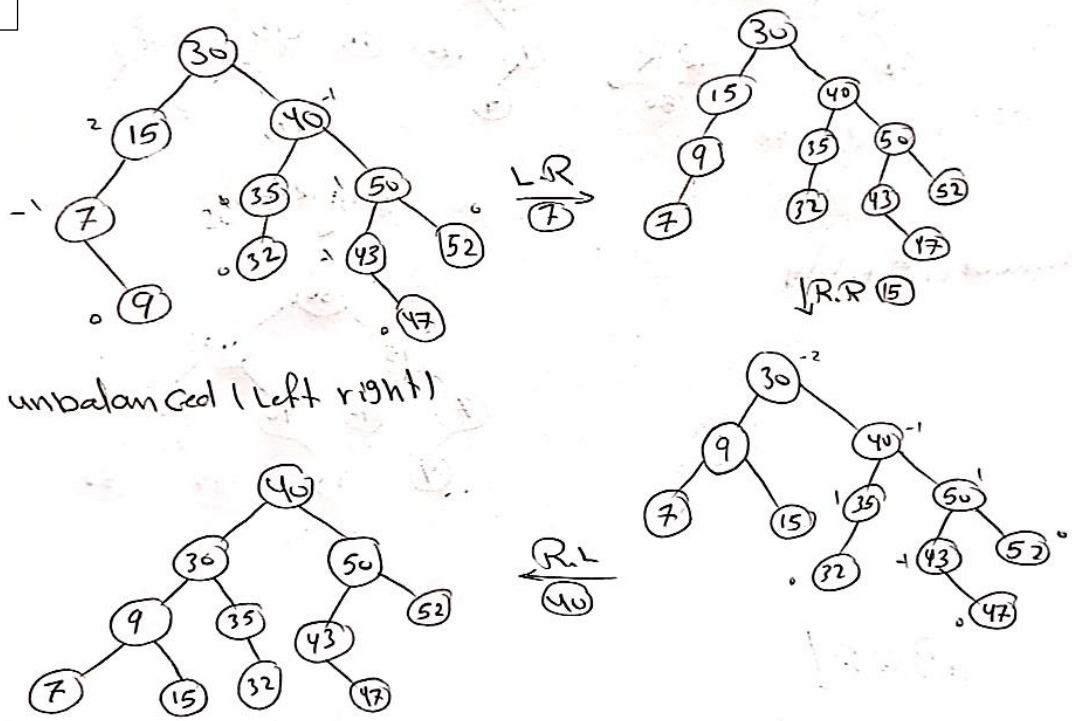
2- If there are n nodes in AVL tree, maximum height can't exceed **1.44*$\log_2$n**

- We can get Maximum height of tree is we put the minimum number of nodes in the tree
- So if I want to calculate the height for 2 nodes the result will be **1.44*$\log_2$2 = 1**

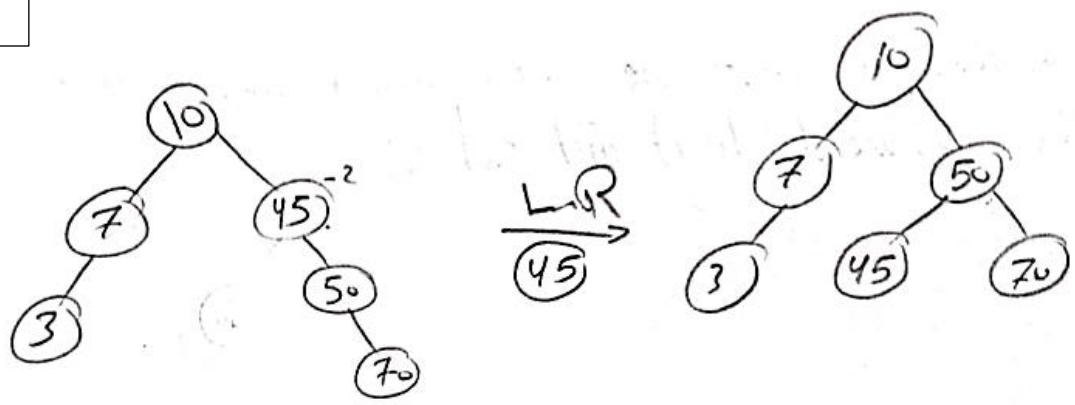
Question 4

Q(4)

a

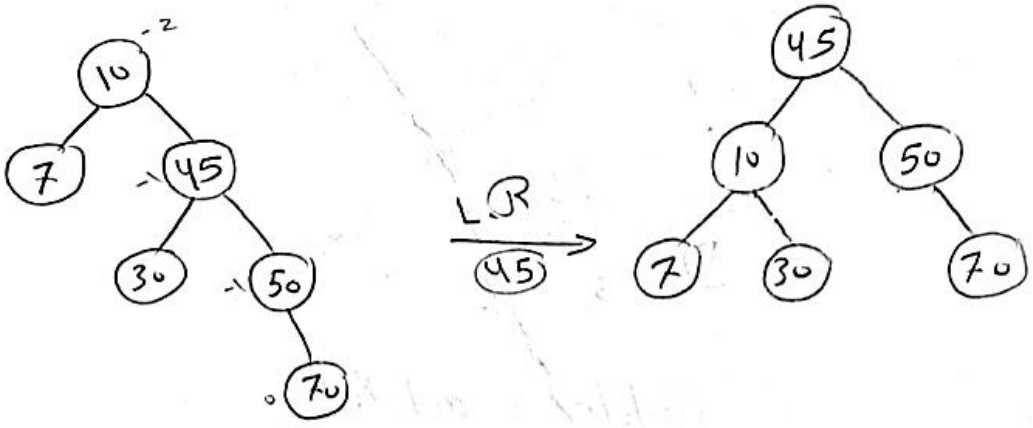


b



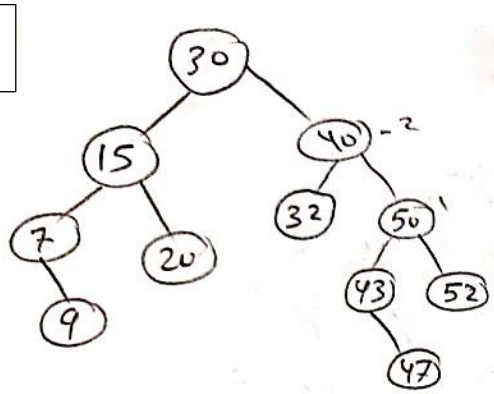
unbalanced (right-right)

c

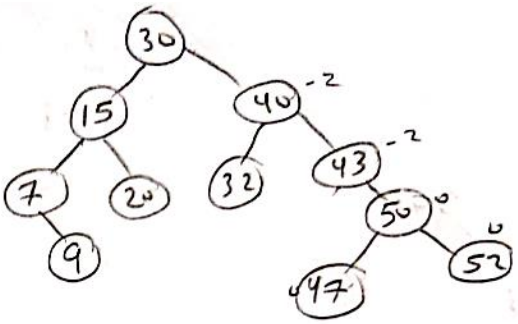


unbalanced (right-right)

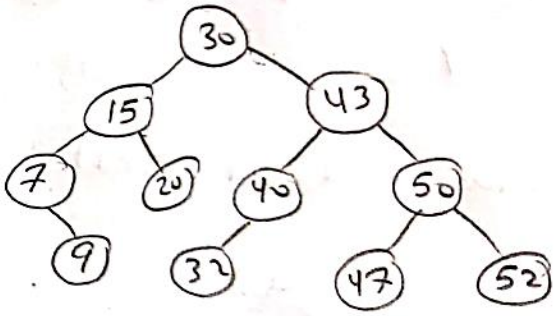
d



R.R
50



40 R.L



unbalanced (right-Left)