TECHNO INDIA UNIVERSITY
WESTBENGAL

Data Structures & Algorithms
(PCC-CS 301)

Dr. Debashis Das
Associate Professor
Department of CSE
Techno India University, Kolkata

Topics Covered

1. Binary tree formation from traversal sequences
2. BST searching
3. Data deletion from BST

Binary Tree

- Binary tree formation from the traversal

  - A binary tree can be formed if two traversal sequences are known
    - Pre-order and in-order can form the tree
    - Post-order and in-order can form the tree
    - But pre-order and post-order cannot form the tree

If there is only pre-order or post-order traversal is provided, you can form a BST as in-order is known as the sorted sequence of the BST

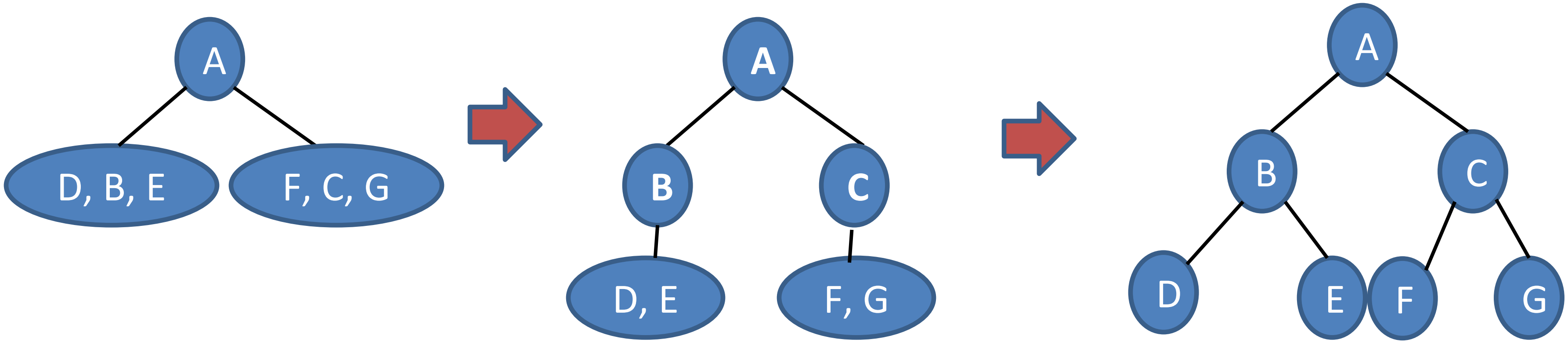
Binary Tree

- Binary tree formation from the traversal

Pre-order: A -> B -> D -> E -> C -> F -> G

In-order: D -> B -> E -> A -> F -> C -> G

- Left most node in the pre-order will be parent
- Follow the in-order traversal
- The set of all nodes that reside in the left of root, are left sub-tree
- The set of all nodes that reside in the right of root, are right sub-tree.



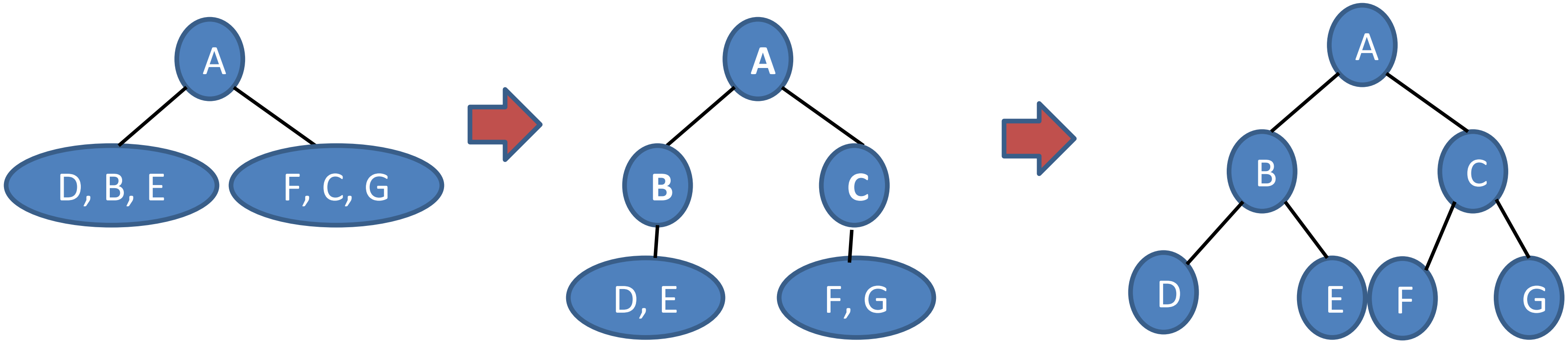
Binary Tree

- Binary tree formation from the traversal

Post-order: D -> E -> B -> F -> G -> C -> A

In-order: D -> B -> E -> A -> F -> C -> G

- Right most node in the post-order will be parent
- Follow the in-order traversal
- The set of all nodes that reside in the left of root, are left sub-tree
- The set of all nodes that reside in the right of root, are right sub-tree.

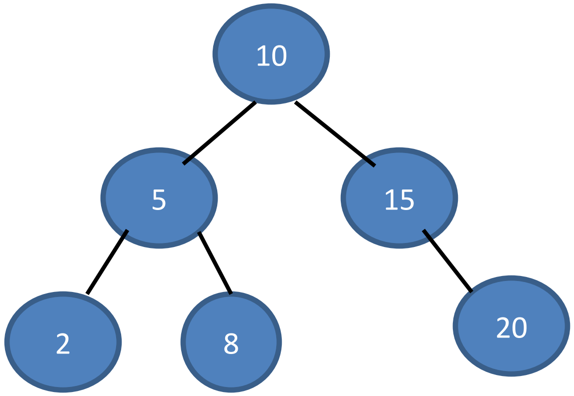


Binary Search Tree

- BST searching
  - Searching of a user given data into a BST
    - Start the searching from the root node
    - If the searching element is less than the current node, search in left sub-tree
    - If it is larger than the current searching node, search the right sub-tree next

Search for 8 =>

8 < 10 :: search the left sub-tree, next searching node is 5
8 > 5 :: search the right sub-tree, next searching node is 8
8 = 8 :: data found



Binary Search Tree

- BST searching (algorithm)

```
BST_search(root, key)  // key is the searching element
{
  set ptr := root
  while ptr != NULL
     if ptr -> data = key
         Print "search successful" and return
     else
         if  ptr -> data < key
              ptr := ptr -> right
         else
              ptr := ptr -> left
   Print "Search unsuccessful"
   return
}
```

Time Complexity:

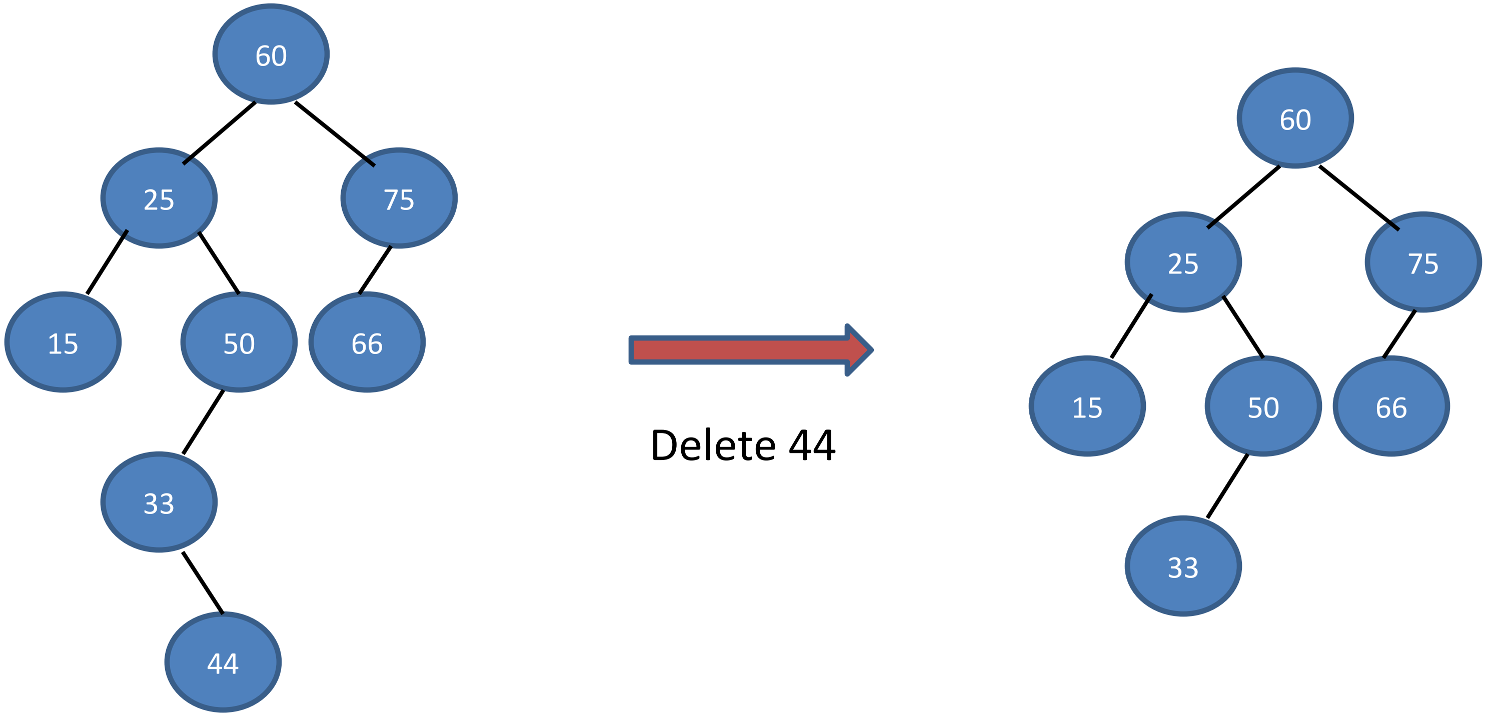
As we are searching a binary tree and data searching process will maximally be executed as the same number of iteration as the height of the tree.
Hence, the time complexity of BST search is $O(\log_2 n)$.

Binary Search Tree

- Data deletion from BST
  - Data deletion from a BST depends on the position of deleting element
    - Deleting data has no children, it is a leaf node
      - ✔ Simply remove the node
    - Deleting data has only one child
      - ✔ Replace the deleting node with its only child node
    - Deleting data has two children
      - ✔ Replace the deleting node with its in-order successor
        - Delete the in-order successor by following case1 or case2 (in-order successor will not have any left child)
        - Replace it with actual deleting node

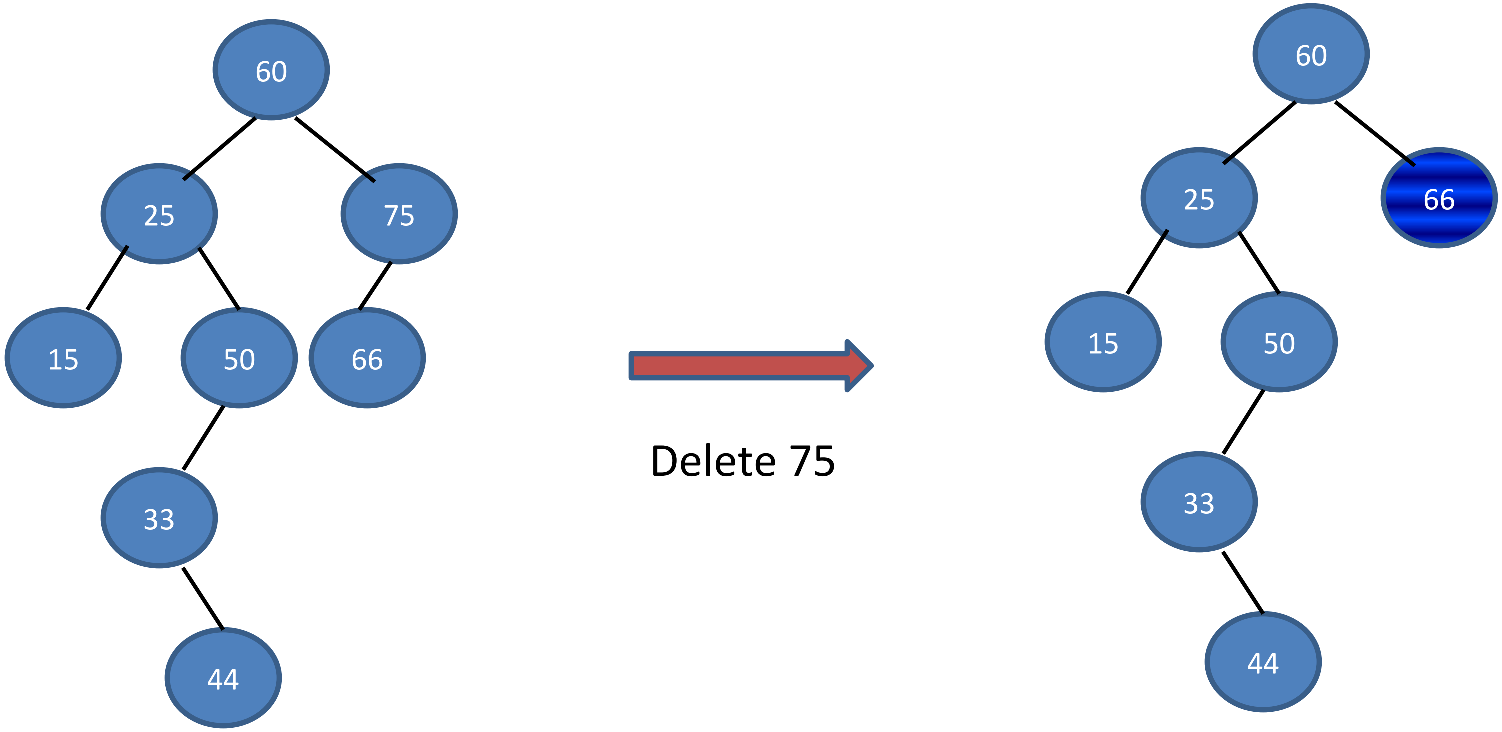
Binary Search Tree

- Data deletion from BST
  - Deleting data has no children



Binary Search Tree

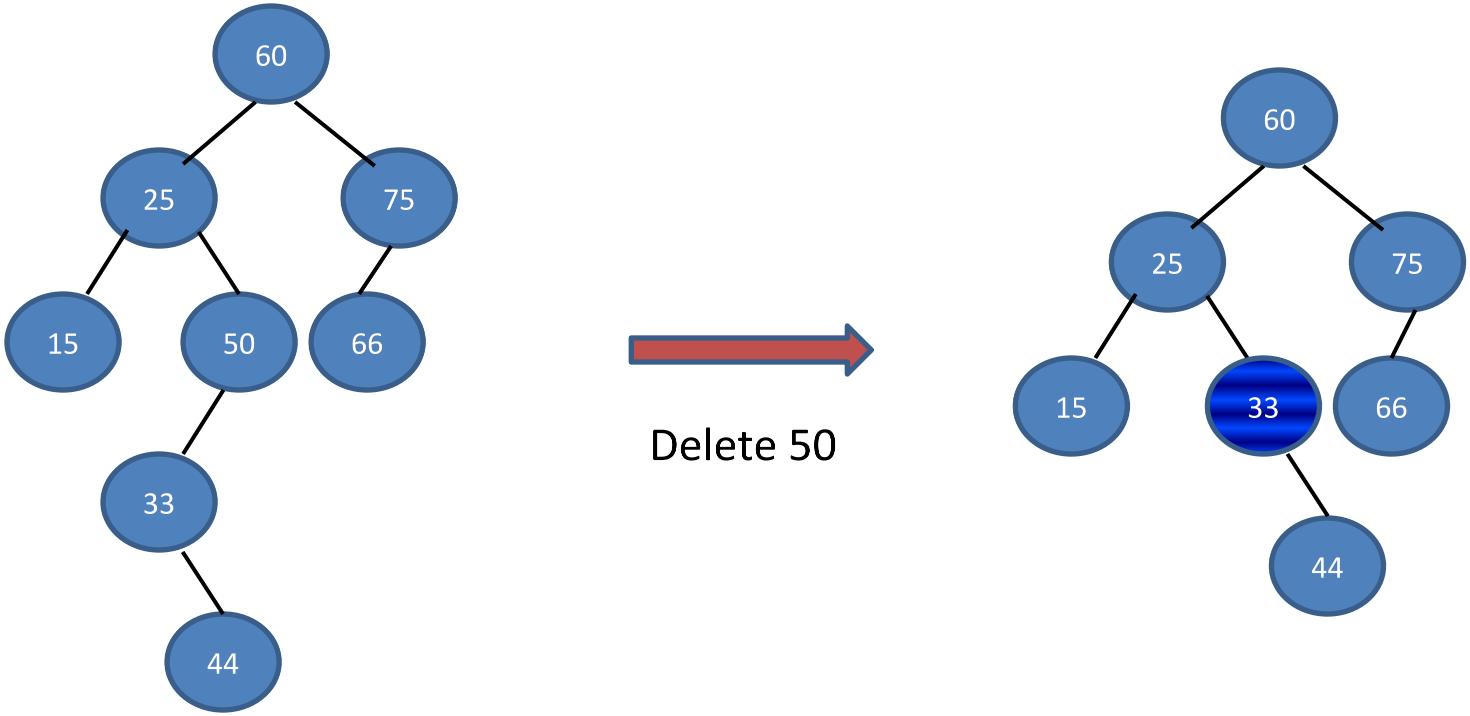
- Data deletion from BST
  - Deleting data has only one child



Delete 75

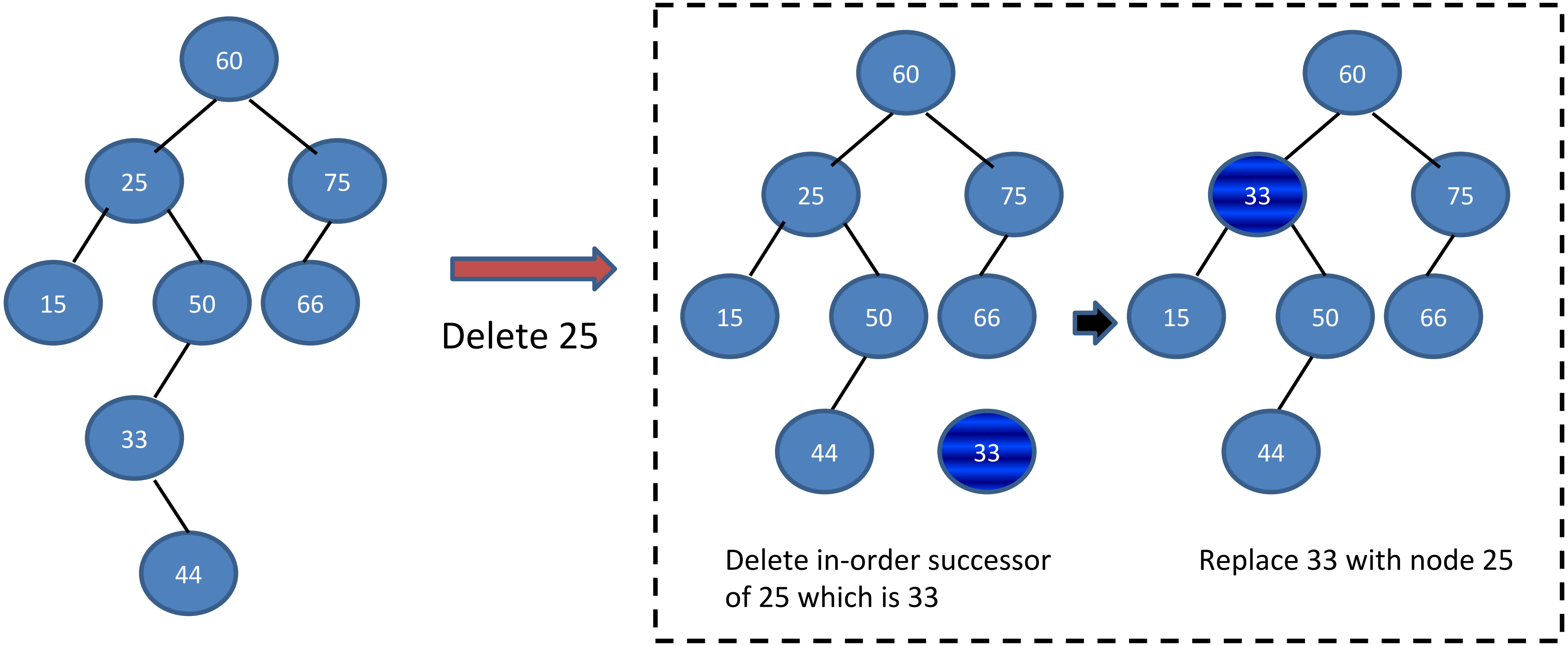
Binary Search Tree

- Data deletion from BST
  - Deleting data has only one child (another example)



Binary Search Tree

- Data deletion from BST
  - Deleting data has two children



Delete 25

Delete in-order successor of 25 which is 33

Replace 33 with node 25

Binary Search Tree

- Data deletion from BST (complexity)

  □ Time complexity

    - ○ Deletion of any data from BST executes in two steps
      - ✔ searching the deleting data position which requires O($\log_2$n) time
      - ✔ Removing of selected data requires simple changing of few links, requires O(1) time
    - ○ Total time complexity O($\log_2$n)

Queries?

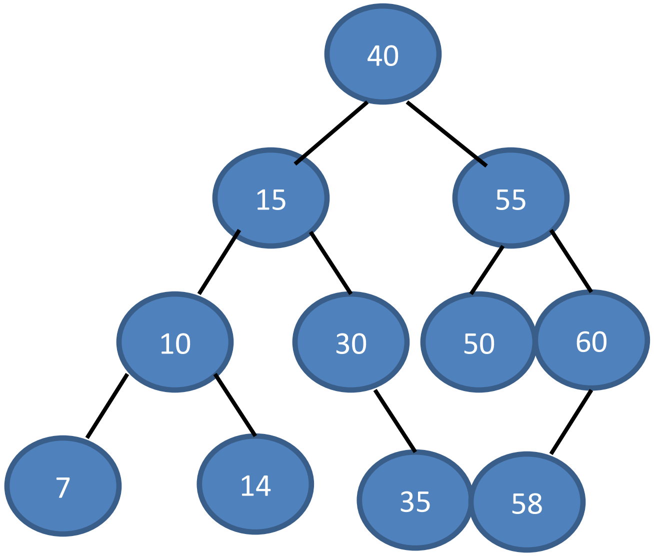
Problem

Q1. Form the BST from the provided traversal

Post-order: 2, 8, 5, 3, 14, 12, 18, 25, 30, 20, 15, 10

Q2. Delete following data from the BST shown below

a) delete 10
b) delete 60
c) delete 40


40
15
55
10
30
50
60
7
14
35
58