No.............. Subject DSA .... Paper ....

ation No.............. Year.............. Stream ....

Searching is used to find the location where an element is available. There are two types of search techniques. They are

1. Linear or sequential search
2. Binary search.

Linear Search.

Linear search or sequential search is a method for finding a particular value in a list that consists of checking every one of its elements, one at a time and in sequence, until the desired one is found.

Algorithm.

1. Repeat For J = 1 to N
2. IF (Item == A[J]) Then
3. Print: Item found at location J.
4. Return [End of if]

5. IF (J > N) Then
6. Print : Item does not exist.
7. Exit.

Complexity.

Linear search on a list of n elements. In the worst case, the search must visit every elements once. This happens when the value being searched for is either the last element in the list, or is not in the list. However, on the average, assuming the value searched for is in the list and each list element is equally likely to be the value searched for, the searched visits only n/2 elements. In best case the array is already sorted i.e O(1).

∴ Worst case – O(n).
Average case – O(n).

Binary Search

A binary search or half-interval search algorithm finds the position of a specified input value within an array sorted by key value. For binary search, the array should be arranged in ascending or descending order. In each step, the algorithm compares the search key value with the key value of the middle element of the array. If they keys match, then a matching element has been found and its index is returned. Otherwise, if the search key is less than the middle element's key; then the algorithm repeats its action on the sub-array to the left of the middle element. or, if the search key is greater, on the sub-array to the right. If the remaining array to be searched is empty, then the key cannot be found in the array and a special 'not found' indication is returned.

Algorithm

1. Set Beg = 1 and End = N
2. Set Mid = (Beg + End)/2
3. Repeat steps 4 to 8 while (Beg <= End) and (A[Mid] ≠ Item)
4. If (Item < A[Mid]) Then
5. Set End = Mid - 1
6. Else
7. Set Beg = Mid + 1
   [End of if]
8. Set Mid = (Beg + End)/2

9. If (A[Mid == Item) Then
10. Print : Item exists at location Mid.
11. Else
12 Print : Item does not exist.

13. Exit.

Time Complexity
The time complexity of binary search in a successful search is O(logn) and for an unsuccessful search is O(logn).

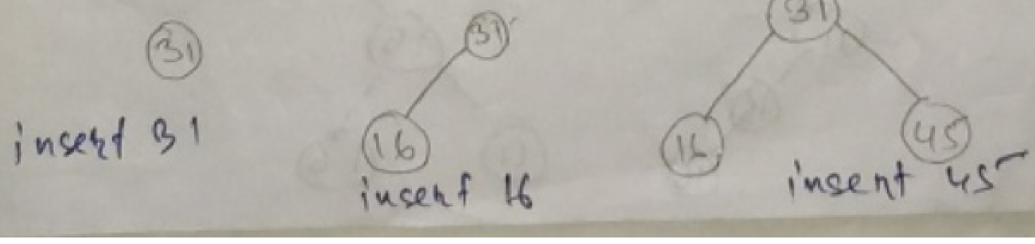
Binary Search Tree.
A binary search tree (BST), sometimes also called an ordered or sorted binary tree, is a node based binary tree data structure where each node has a comparable key and satisfies the restriction that the key in any node is larger than the keys in all nodes in that nodes left subtree and smaller than keys in all nodes in that node's right sub-tree. Each node has no more than two child nodes. Each child must either be a leaf node or the root of another binary search tree. The left sub-tree contains only nodes with keys less than the parent

node; the right sub-tree contains only nodes with keys greater than the parent node. BSTs are also dynamic datastructures, and the size of a BST is only limited by the amount of free m/y in the OS. The main advantage of BSTs is that it remains ordered, which provides quicker search times than many other data structures. The properties of BSTs are follows:

- The left subtree of a node contains only nodes with keys less than the node's key.
- The right subtree of a node contains only nodes with keys greater than the node's key.
- The left and right subtree each must also be a BST.
- Each node can have up to two successor node.
- There must be no duplicate nodes.
- A unique path exists from the root to every other node.

Insertion in BST

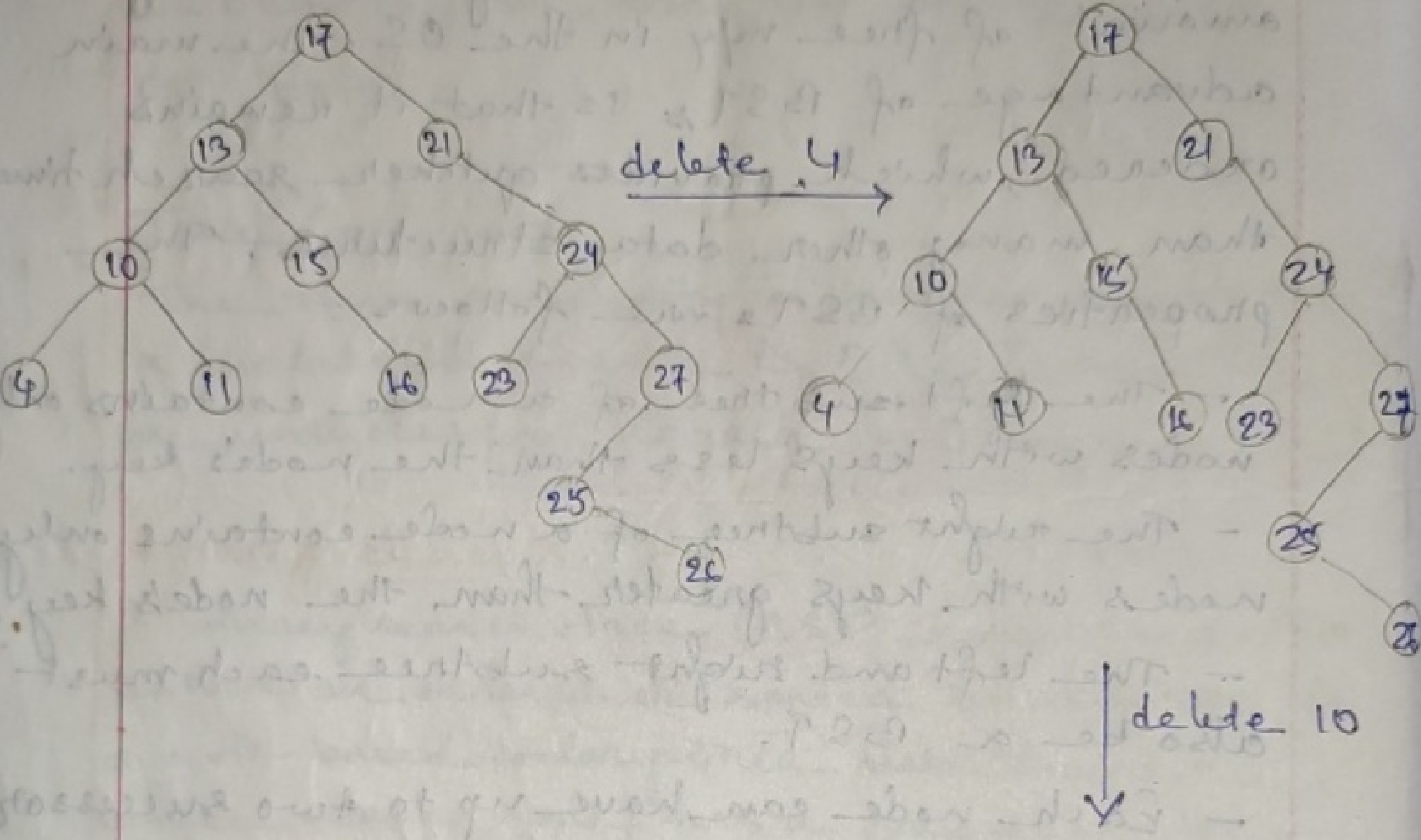
31 16 45 24 7 19 29



continue

Deletion in BST.

Consider the BST shown below first the element 4 is deleted. Then 10 is deleted and after 27 is deleted from the BST.



delete 27

Method 1   Method 2.

delete 13