At this lab section, we will learn implementation of Binary Search Trees and make search, insertion, deletion operations on them.

Binary Search Trees

Asst. Prof. Dr. Feriştah DALKILIÇ
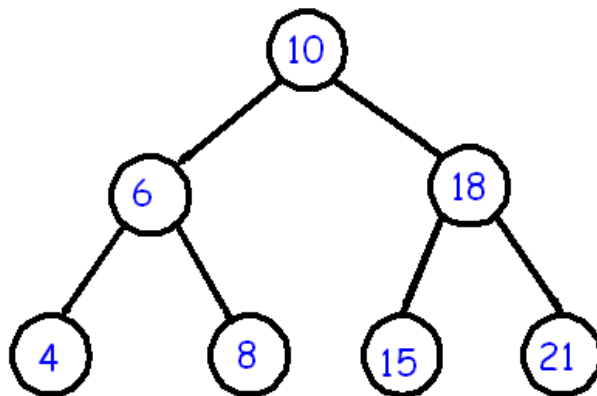Res. Asst. Fatih DİCLE

PART 1 – Binary Search Tree

A **Binary Search Tree (BST)** is a binary tree whose nodes contain Comparable objects and are organized as follows:

For each node in a binary search tree,

- The node's data is greater than all the data in the node's left subtree,
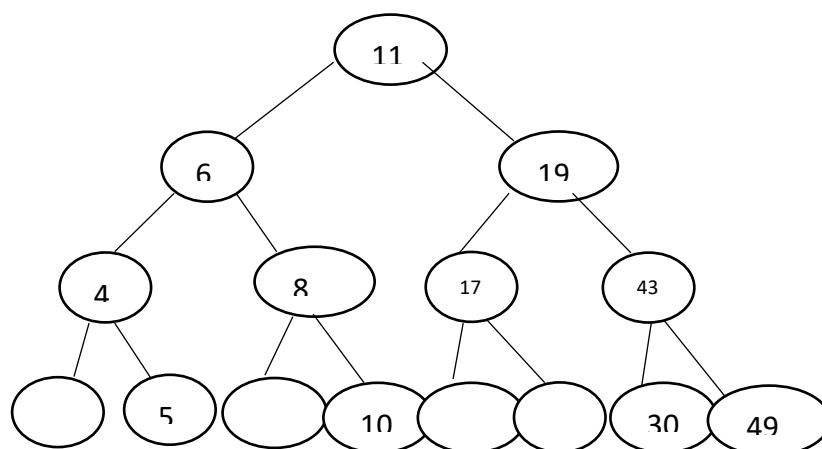- The node's data is less than all the data in the node's right subtree.

Exercise - 1
At this section, we will do some operations on a binary search tree according to BST rules.

Step – 1
Insert items to the tree and show the inserted version of the BST.

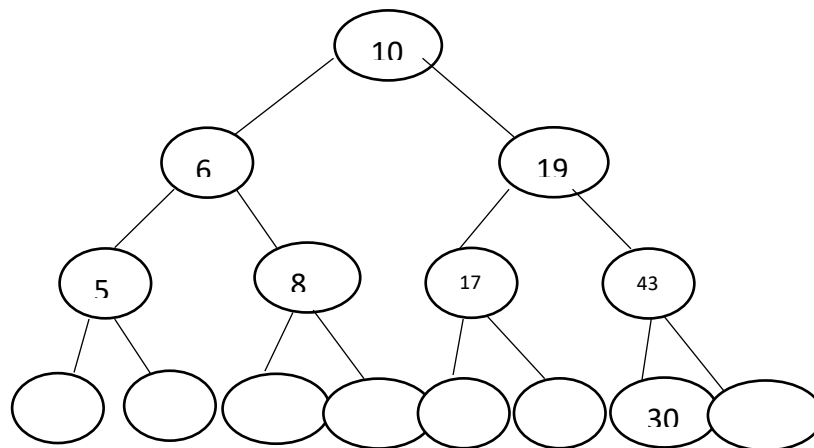Items = [11, 6, 8, 19, 4, 10, 5, 17, 43, 49, 30]

Search some items in the BST, write the visited node sequence below.

Search 5 -> 11,6,4,5

Search 30 -> 11,19,43,30

Step – 3

Show the final state of the BST after deleting 11, 4 and 49.
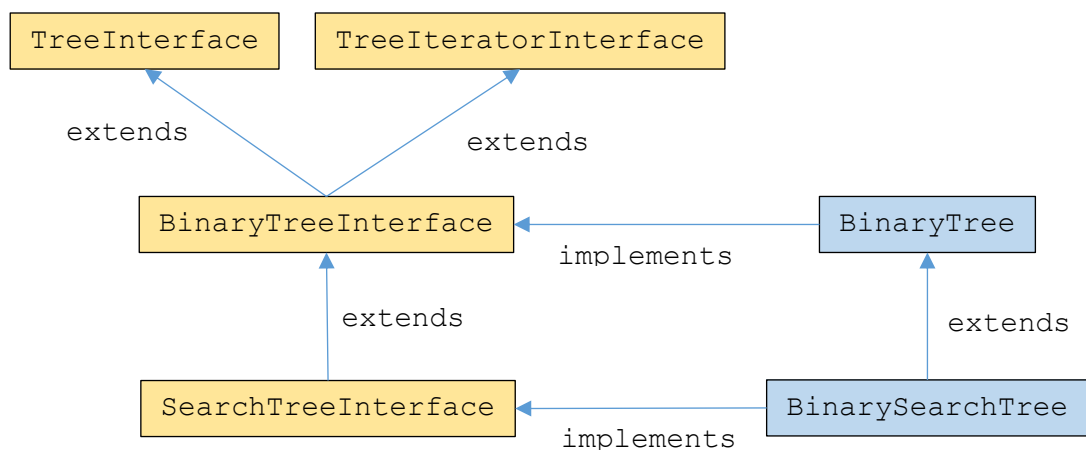

2

Exercise - 2

At this section, we will experiment insertion, deletion, and search operations on Binary Search Tree in Java.

Step – 1

You are given some ready to use classes in *src* folder. The class hierarchy is given below:

Create a new Java project and add given classes into your project.

Fill in the missing parts in findEntry, addEntry, and removeEntry methods of the *BinarySearchTree.java*.

Add a new class with the name Test.java. Create a *BinarySearchTree* instance. Read the given input file *StudentNumbers.txt* and insert all the items into BST.

Test whether the BST is empty or not.

Print the data in the root node.

Print the height of the BST.

Print the number of nodes in the BST.

Get the entry equals to your student number.

Test whether the BST contains your student number.

Remove your student number from the BTS.

Iterate and print the BTS by using in-order traversal technique.

Paste your final content of *Test.java* and your final output.

Your Test.java

Your Output