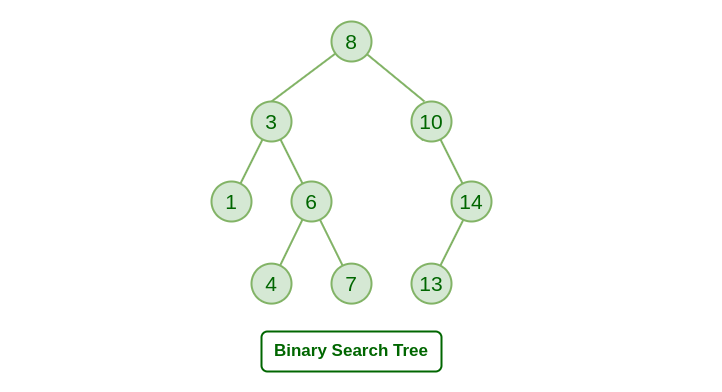
**Binary Search Tree**

Last Updated : 08 Feb, 2025

A **Binary Search Tree (or BST)** is a data structure used in computer science for organizing and storing data in a sorted manner. Each node in a **Binary Search Tree** has at most two children, a **left** child and a **right** child, with the **left** child containing values less than the parent node and the **right** child containing values greater than the parent node. This hierarchical structure allows for efficient **searching**, **insertion**, and **deletion** operations on the data stored in the tree.



**ntroduction to Binary Search Tree**

Last Updated : 12 Dec, 2024

**Binary Search Tree** is a data structure used in computer science for organizing and storing data in a sorted manner. Binary search tree follows all properties of binary tree and for every nodes, its **left** subtree contains values less than the node and the **right** subtree contains values greater than the node. This hierarchical structure allows for efficient **Searching**, **Insertion**, and **Deletion**operations on the data stored in the tree.

*Binary Search Tree*

**Properties of Binary Search Tree:**

* 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 binary search tree.
* There must be no duplicate nodes(BST may have duplicate values with different handling approaches).

**Test Your Understanding**

Given a Binary Tree, find out if it is Binary Search Tree or not.

1 / 3

**Important Points about BST**

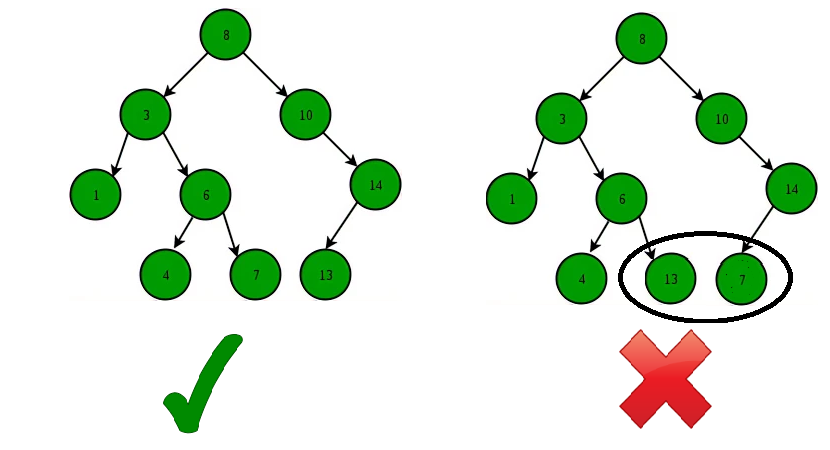
* A Binary Search Tree is useful for maintaining sorted stream of data. It allows search, insert, delete, ceiling, max and min in O(h) time. Along with these, we can always traverse the tree items in sorted order.
* With Self Balancing BSTs, we can ensure that the height of the BST is bound be Log n. Hence we achieve, the above mentioned O(h) operations in O(Log n) time.
* When we need only search, insert and delete and do not need other operations, we prefer[Hash Table](https://www.geeksforgeeks.org/hash-table-data-structure/) over BST as a Hash Table supports these operations in O(1) time on average.

**Applications of BST**

Last Updated : 30 Jul, 2024

[Binary Search Tree](https://www.geeksforgeeks.org/binary-search-tree-data-structure/) (BST) is a data structure that is commonly used to implement efficient searching, insertion, and deletion operations along with maintaining sorted sequence of data. Please remember the following properties of BSTs before moving forward.

* The left subtree of a node contains only nodes with keys lesser 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 binary search tree. There must be no duplicate nodes.



*Binary Search Tree – Structure*

A BST supports operations like search, insert, delete, maximum, minimum, floor, ceil, greater, smaller, etc in O(h) time where h is height of the BST. To keep height less, self balancing BSTs (like [AVL](https://www.geeksforgeeks.org/avl-tree-set-1-insertion/) and [Red Black Trees](https://www.geeksforgeeks.org/red-black-tree-set-1-introduction-2/)) are used in practice. These Self-Balancing BSTs maintain the height as O(Log n). Therefore all of the above mentioned operations become O(Log n). Together with these, BST also allows sorted order traversal of data in O(n) time.

1. A Self-Balancing Binary Search Tree is used to maintain sorted stream of data. For example, suppose we are getting online orders placed and we want to maintain the live data (in RAM) in sorted order of prices. For example, we wish to know number of items purchased at cost below a given cost at any moment. Or we wish to know number of items purchased at higher cost than given cost.
2. A Self-Balancing Binary Search Tree is used to implement [doubly ended priority queue](https://www.geeksforgeeks.org/double-ended-priority-queue/). With a Binary Heap, we can either implement a priority queue with support of extractMin() or with extractMax(). If we wish to support both the operations, we use a Self-Balancing Binary Search Tree to do both in O(Log n)
3. There are many more algorithm problems where a Self-Balancing BST is the best suited data structure, like [count smaller elements on right](https://www.geeksforgeeks.org/count-smaller-elements-on-right-side/), [Smallest Greater Element on Right Side](https://www.geeksforgeeks.org/smallest-greater-element-on-right-side/), etc.
4. A BST can be used to sort a large dataset. By inserting the elements of the dataset into a BST and then performing an in-order traversal, the elements will be returned in sorted order. When compared to normal sorting algorithms, the advantage here is, we can later insert / delete items in O(Log n) time.
5. Variations of BST like B Tree and B+ Tree are used in Database indexing.
6. [**TreeMap**](https://www.geeksforgeeks.org/treemap-in-java/) and **[TreeSet](https://www.geeksforgeeks.org/treeset-in-java-with-examples/" \t "_blank) in**Java, and [set](https://www.geeksforgeeks.org/set-in-cpp-stl/) and [map](https://www.geeksforgeeks.org/map-associative-containers-the-c-standard-template-library-stl/) in C++ are internally implemented using self-balancing BSTs, more formally a Red-Black Tree.