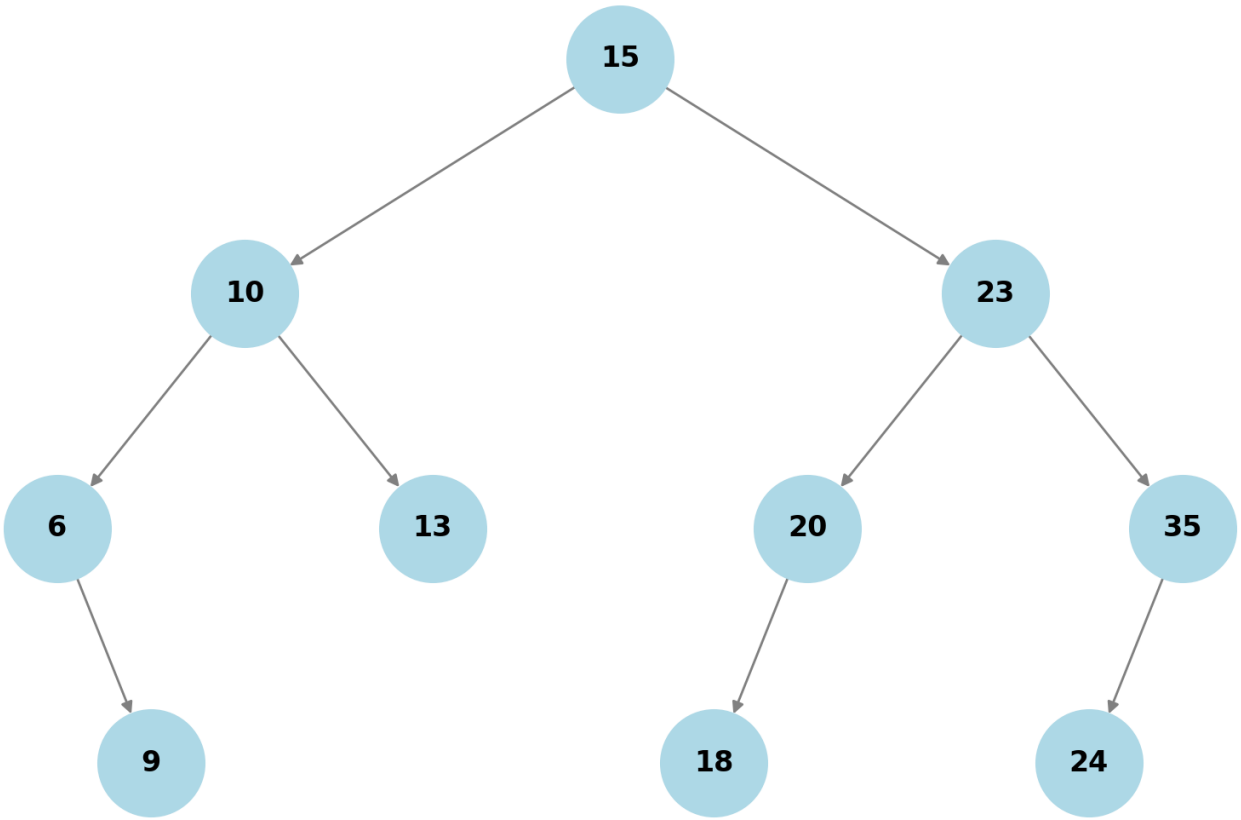
Binary Search Tree (BST) - Lab 6

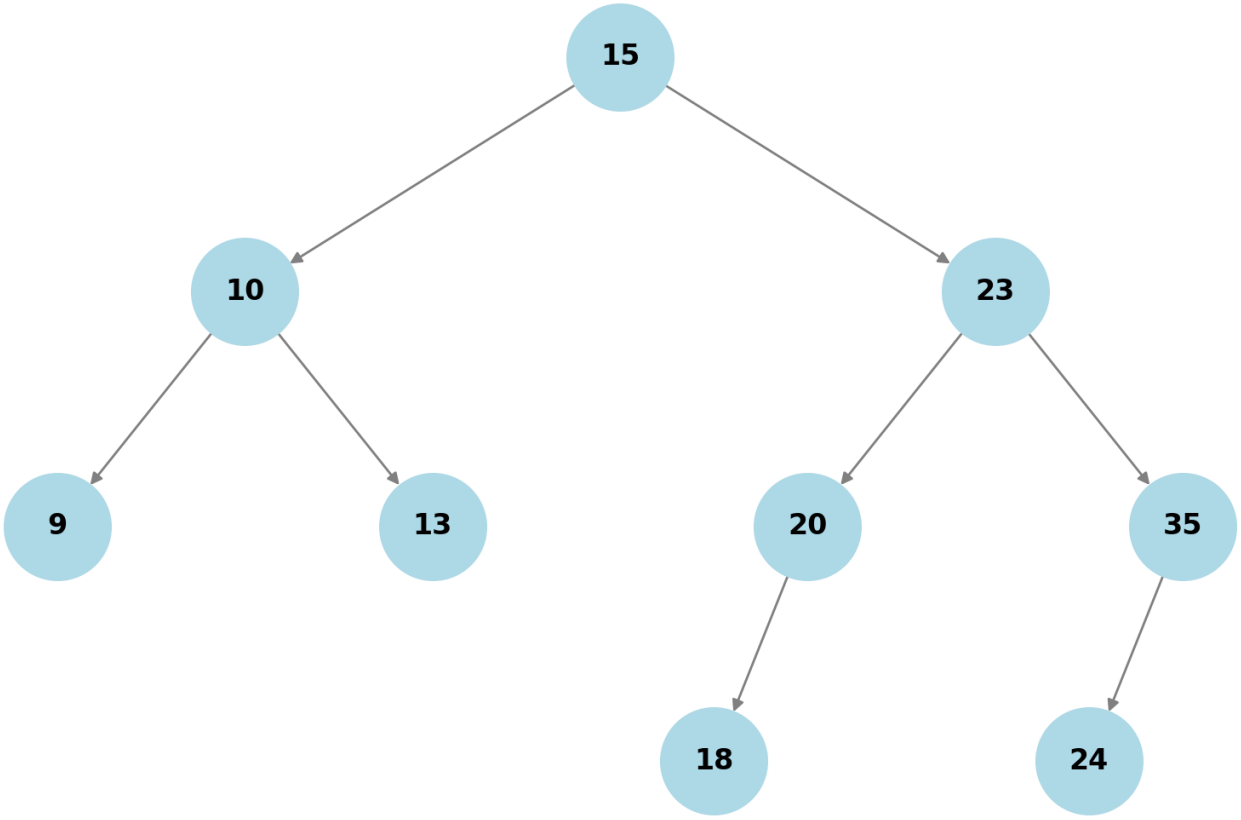
Task Description

This report presents the implementation of a Binary Search Tree (BST) in Java, demonstrating insertion, search, and deletion operations. The BST is visualized using Python. We inserted specific elements into the tree, performed deletions, and analyzed the impact on structure.

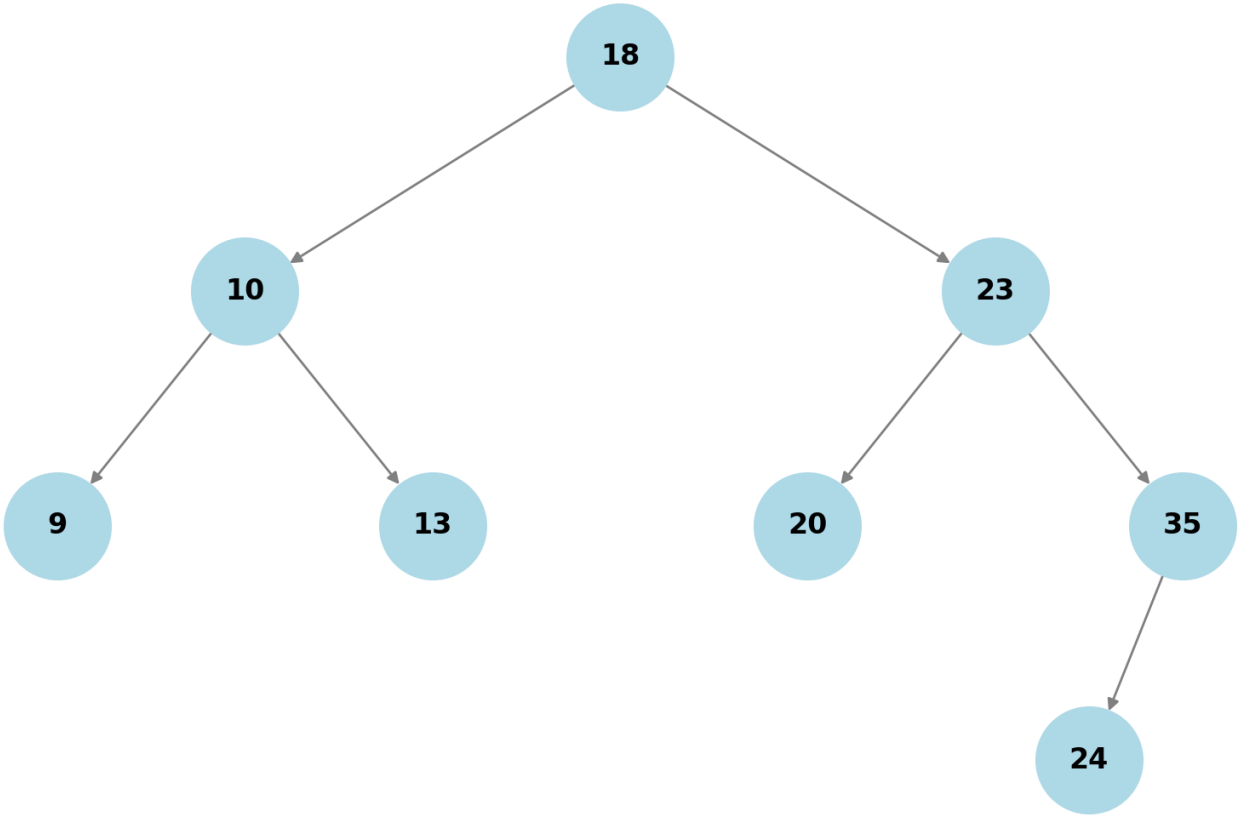
BST Before Deletion



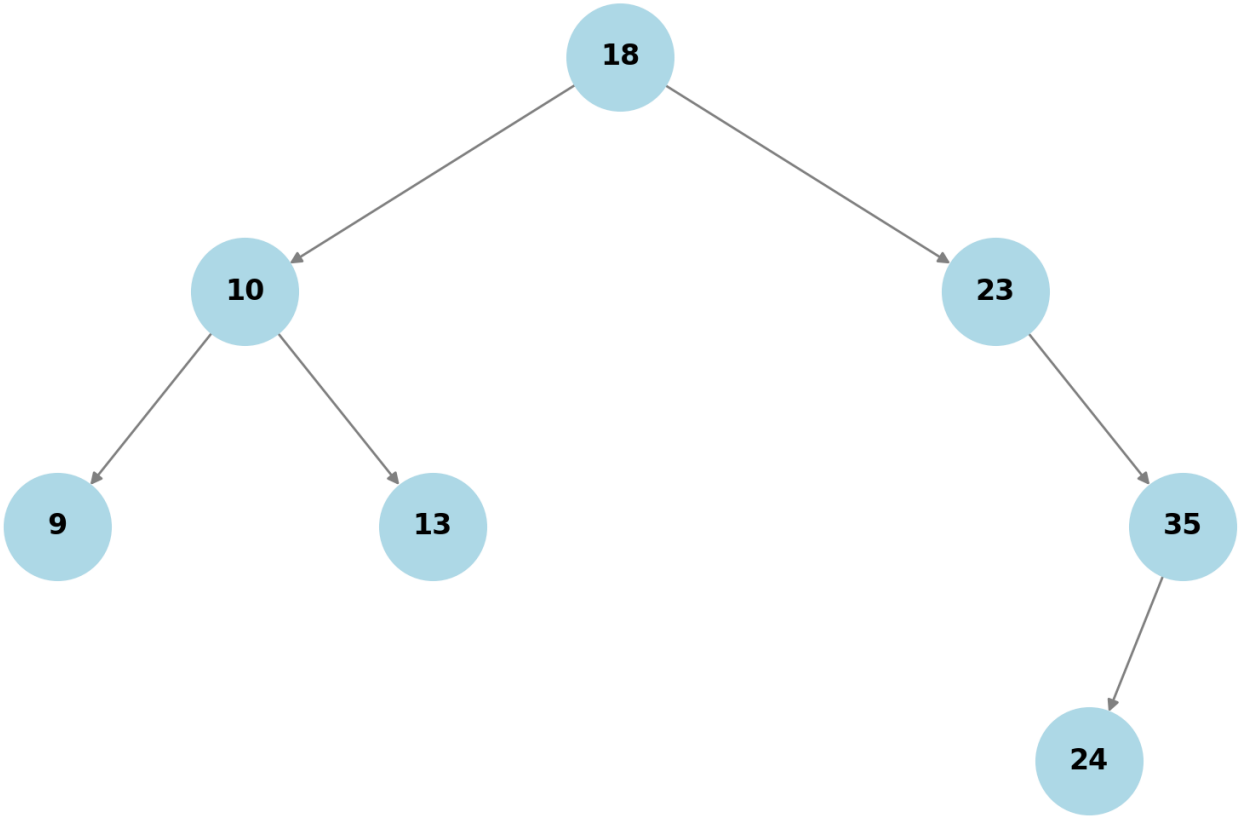
BST After Deleting 6



BST After Deleting 15



BST After Deleting 20



Justification of BST Changes

1. **Deleting 6**: Since 6 was a leaf node, it was removed directly without affecting the structure.
2. **Deleting 15**: The node had children (10, 23), so it was replaced by its in-order successor (18).
3. **Deleting 20**: The node was replaced by its smallest right subtree element (24).

This sequence of operations maintains the BST properties.

Java Implementation of BST

```
class Node {
    int key;
    Node leftChild, rightChild;
    public Node(int key) {
        this.key = key;
        leftChild = rightChild = null;
    }
}

class BinarySearchTree {
    Node root;
    public BinarySearchTree() { root = null; }
    void insert(int key) { root = insertRec(root, key); }
    Node insertRec(Node root, int key) {
        if (root == null) { root = new Node(key); return root; }
```

```
        if (key < root.key) root.leftChild = insertRec(root.leftChild, key);
        else if (key > root.key) root.rightChild = insertRec(root.rightChild, key);
        return root;
    }
    void inorderTraversal(Node root) {
        if (root != null) {
            inorderTraversal(root.leftChild);
            System.out.print(root.key + " ");
            inorderTraversal(root.rightChild);
        }
    }
    public static void main(String[] args) {
        BinarySearchTree bst = new BinarySearchTree();
        int[] elements = {23, 41, 75, 43, 81, 12, 34, 99, 101, 106, 27};
        for (int elem : elements) { bst.insert(elem); }
        System.out.println("BST In-order:");
        bst.inorderTraversal(bst.root);
    }
}
```