Yusuf Fawzy Elnady

Binary Search Tree (additional functionalities)

Summary:

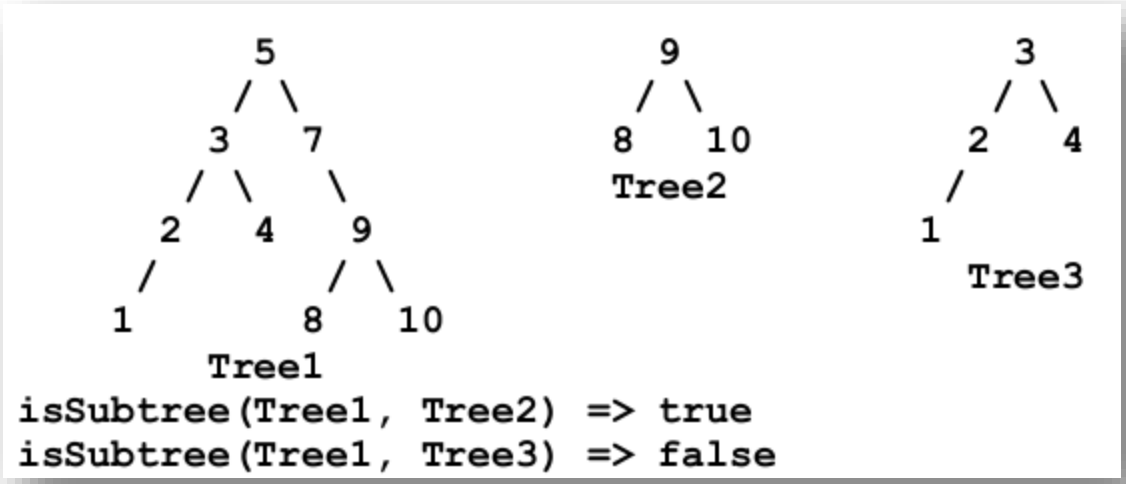
This program entails two classes, a **template binary search tree class** with this name **BSTFCI** and a **node class** with name **BSTNode.** The following functionalities are provided (Check Tree Balance, Tree Comparison, Print Range and T).

1. Checking Tree Balance:

A **Balanced Binary Tree** is a tree where the **heights** of the two child sub-trees of any node differ by at most one and the left subtree is balanced, and the right subtree is balanced. The method "**isBalance**" checks if the BST is balanced or not.

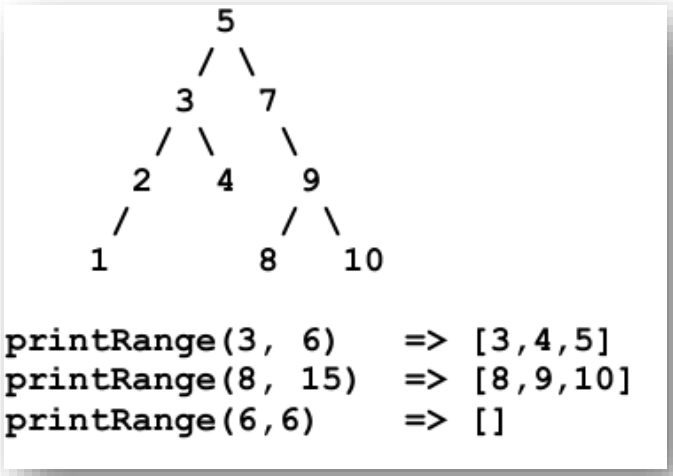
2. Tree Comparison:

A function that decides if a **BSTFCI T2** is a sub-tree of another **BSTFCI T1**.
Prototype: **bool isSubTree(BSTFCI* t1, BSTFCI* t2);**



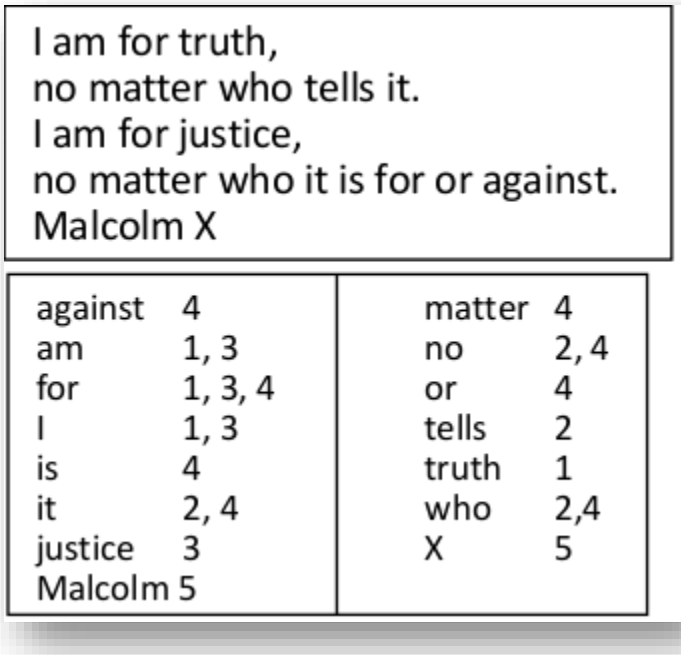
3. Print Range:

A recursive function named "**printRange**" that stores integers given a low key value and a high key value, then prints in sorted order all records whose key values fall between the two given keys by visiting as few nodes in the BST as possible.



4. Number of Lines of each Word:

This is an app that makes use of our class **BSTFCI,** the application takes text consisting of lines and prints a list of the words of the text and the lines they appear on are printed next to them.
The application works by building a binary search tree and each node contains a word and a vector of that contains the list of lines where this word exists. The punctuation marks like "." and "," are removed before processing the text.



The project is built using **C++, Visual Studio**.