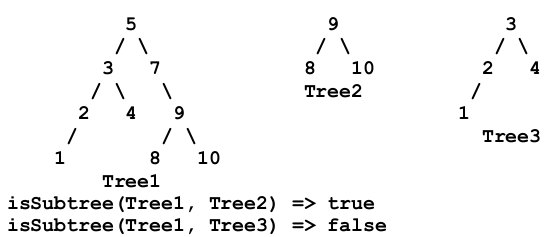
**Create template binary search tree class with this name BSTFCI and create node class with name BSTNode**

**1. Add 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 balanaced and the right subtree is balanced.

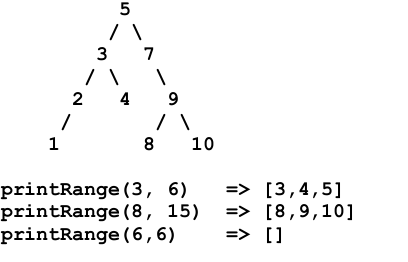
Add method called **‘isBalance’** to BSTFCI this method will check in the BST is balance or not.

**2. Tree Comparison**-Write a function that decides if a BSTFCI T2 is a sub-tree of another BSTFCI T1.  
**Prototype: bool isSubTree(BSTFCI\* t1, BSTFCI\* t2);**  
Note: You may need to write another function that takes 2 BSTNodes and compares their sub-trees.



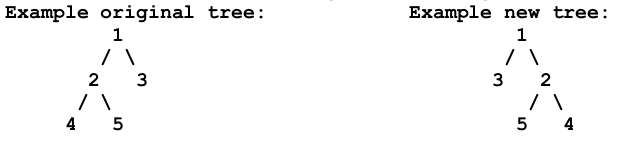
**3. Print Range**

Add a recursive function named **printRange** in the class BSTFCI that stores integers and given a low key value and a high key value, it prints in sorted order all records whose key values fall between the two given keys.   
Function **printRange** should visit as few nodes in the BST as possible.   
**You should NOT** traverse all the tree in-order and print the ones in the range. This will not be considered a correct solution. You should do smart traversal to only traverse the related parts.

****

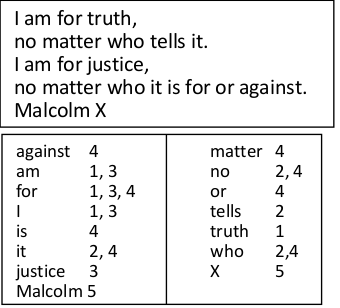
**4. Tree Flipping**

- Write a **flip** method that takes the node which the mirror image of the tree will start from if no parameter send to the function the default value will be the root node. **void flip(Node\* node = root)**



1. **Tree Application**

* Write an index builder application that 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 using BSTFCI and each node contains a word and a vector of that contains the list of lines where this word exists. For each new word, the program finds it and adds the line number to the vector. If word is not found, it is added to the tree. Then traverse the tree in-order to print the nodes.
* You need to remove punctuation marks like **.** and **,** from the text before processing it.
* For example, the text below produces the given index, Test Your code on the given text and 1 more examples.

****