- Don't forget to set your Eclipse workspace and working set.
- You must submit the JAR file, exported (with source code), from your Eclipse project.
- You must check your JAR file to make sure all the source files (.java files) are present. It can be opened with file compression programs such as 7-zip or Winrar.
- Failure to export properly will result in your work not getting marked.

1) To submit:
    - Export your project to a JAR file, with source code.
    - Name your JAR file ID_Week10_Q2.jar. For example, 6623110021_Week10_Q2.jar
    - Submit the JAR file on MyCourseville.

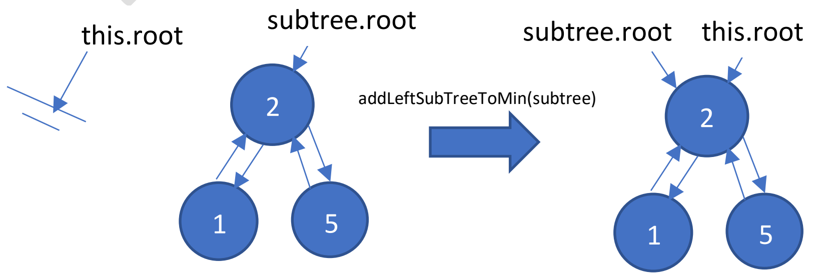
You are given all classes for coding a binary search tree.

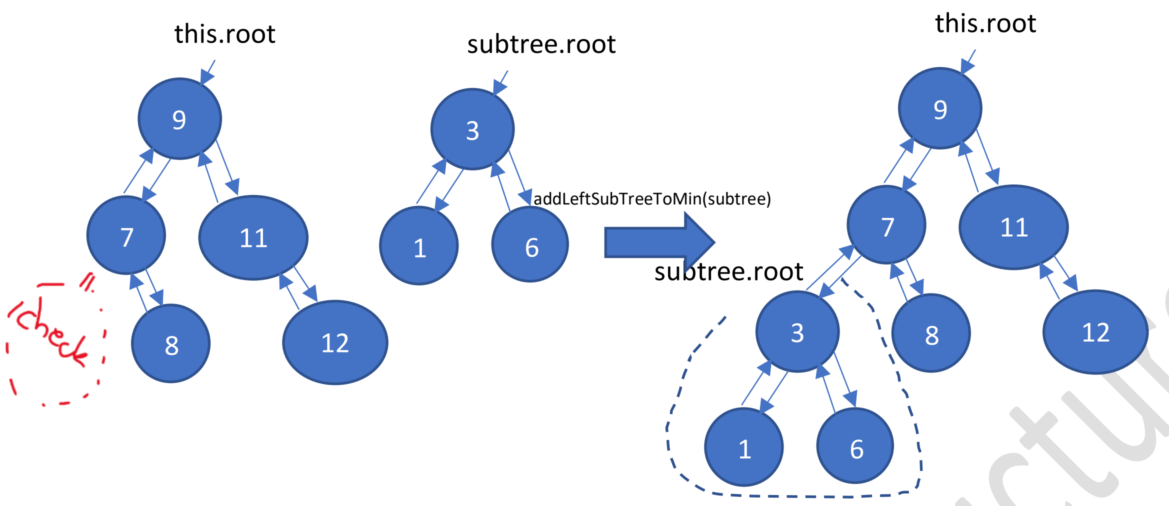
In class BST, write method

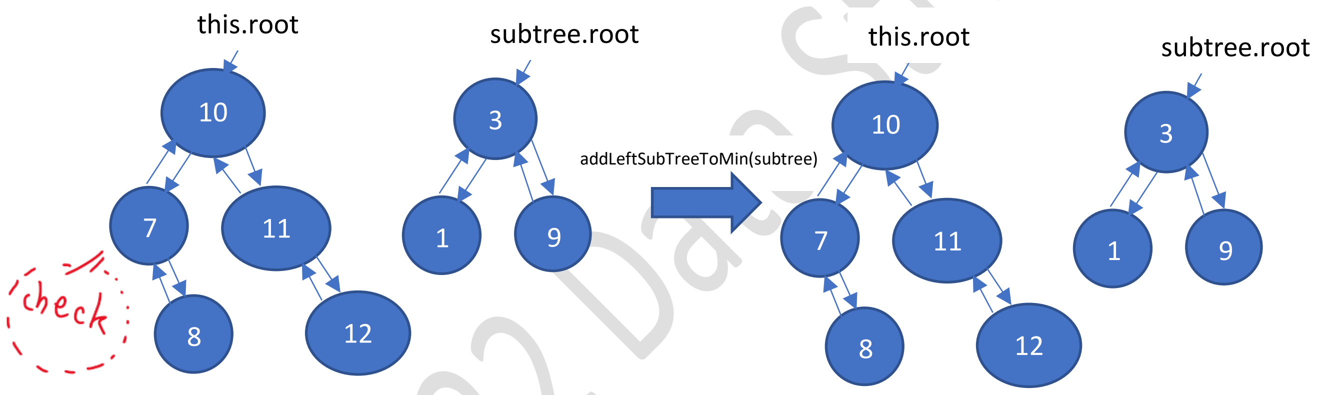
```
public void addLeftSubTreeToMin(BST subtree)
```

- This method tries to add an entire "subtree" as a left subtree of the left most node in our tree, changing our tree.
- Assume there will be no direct access, from anywhere else apart from our BST, to "subtree" in the future.
- If the "subtree" is an empty tree, this method does nothing.
- If our tree is empty, then our tree becomes the subtree.
- Before doing change, check whether after the addition of "subtree", the tree will still be a binary search tree:
    - If so, link the entire subtree to our tree.
    - If not, do nothing.
- The method must not have any loop (but you can call existing methods that have loop).
- Only BST.java is allowed to be modified.
    - Only modify `addLeftSubTreeToMin` method. You are not allowed to create new method(s).
- This `addLeftSubTreeToMin` method must be the last method in class BST (not counting main method).

Example:







Nothing changes because we won’t get a binary search tree.

The JUnit tests are in BSTTest.java (If you don’t do any proper coding you won’t get any mark)

• `testAddEmptySubTree()`	1 marks
• `testAddToEmptyTree()`	2 marks
• `testAddSuccess()`	3 marks
• `testAddFail()`	2 marks
• `testNoLoop()`	2 marks