Mandatory assignment two.

In this assignment you will implement a binary search tree.

Before you start coding consider the ADT and make sure you have all interfaces in place. You do not have to make a generic tree; it is fine with a tree that can take integers. The traversals may also return a string, you don't need to return an iterator.

You should build the tree it in three phases:

1. A binary tree
2. A binary search tree with add
3. A binary search tree with remove

Test driven development is well suite for the project since it can be broken into small simple bits that can be test on their own. Besides, it is a requirement that all parts of the code are well tested. When you encounter errors, it can be useful to see a graphical representation of the tree.

To that end I have uploaded the source code for a simple ascii code printout of a tree, but you are welcome to make a nicer graphical presentation (it is not a requirement).

Remember that in phase one there are no add or delete methods. This is because a general binary tree does not have a specific ordering.

In order to create a tree you must manually do it by creating and combining Nodes.

The assignment will be handed in 1/11-2020. According to the course description mandatory assignments must be handed in and approved in order to register for the exam.

Binary Tree ADT:

Operation	Description
getRoot	Returns a reference to the root
isEmpty	Determines whether the tree is empty
Size	Returns the number of elements in the tree
Contains	Determines if an element is present in the tree
inOrder	Returns an inOrder representation of the tree
Preorder	Returns an preOrder representation of the tree
postOrder	Returns an postOrder representation of the tree
levelOrder	Returns an level Order representation of the tree
Height	Returns the height of the tree

Binary Tree Node ADT:

Operation	Description
setElement	Store the element in the Node
getElement	Returns the element from the Node
addLeftChild	Add a left child to the Node
addRightChild	Add a right child to the Node
getLeftChild	Returns a reference to the left child
getRightChild	Returns a reference to the right child

Binary Search Tree ADT (extends Binary Tree):

Operation	Description
addElement	Add an element to the tree.
removeElement	Remove an element from the three
removeAllOccurrences	Remove all occurrences of an element from the tree
removeMin	Remove the minimum element from the three
removeMax	Remove the maximum element from the three
findMin	Returns a reference to the minimum element of the tree
findMax	Returns a reference to the maximum element of the tree

Binary Search Tree Node ADT (extends Binary Tree Node):

Operation	Description
setParent	Set the parent of a Node
getPatent	Returns the parent of a Node