CS119 – Activity 9

Consider the abstract data type Set:

Function	Explanation
`empty :: Set a`	gives an empty set
`isEmpty :: Set a -> Bool`	determines if a set is empty
`member :: Set a -> a -> Bool`	determines whether a value is contained in a set
`insert :: a -> Set a -> Set a`	adds a value to a set
`delete :: a -> Set a -> Set a`	deletes a value to a set

Suppose that we decide to implement this ADT with a list of elements as provided in the notes . If we have a set with n elements, in the worst case how many elements would have to be examined in the `member` operation? How many elements would have to be examined in the `insert` operation?

Suppose that we decide instead to implement this ADT with a binary search tree as provided in the notes . Remember that in a binary search tree all elements in the left subtree will be less than the root, all elements in the right subtree will be greater than the root, and the subtrees will also be binary search trees.

Draw two different BSTs for the set {5,10,22,26,30}.

What are the sequences of nodes examined for each of your trees when using the `member` operation for seeing if the value 15 is contained in the set?

What happens for each of your trees when we insert the value 15? Draw the new trees with the inserted value.

What happens if we try to delete the value at the root of your tree? Draw the new tree.