```java
package applicationLevelMethods;

import bst_Package.BSTInterface;
import bst_Package.BSTNode;
import bst_Package.BinarySearchTree;
import queues.ArrayBndQueue;
import queues.LinkedUnbndQueue;

public class Methods {

    // If tree is not empty then skip first node that is in order
    // Return the element
    public static <T> Comparable<?> min(BinarySearchTree<?> tree) {
        if (!tree.isEmpty())
            {tree.reset(BSTInterface.INORDER);
            return tree.getNext(BSTInterface.INORDER);}
        else
            throw new NullPointerException();
    }

    // If tree is not empty skip to the last node that is in order
    // Return the element
    public static <T> Comparable<?> max(BinarySearchTree<?> tree) {
        if (!tree.isEmpty())
            {
                int count = tree.reset(BSTInterface.INORDER);

                int max = 0;
                for (int i = 1; i <= count; i++) {
                    max = (Integer)tree.getNext(BSTInterface.INORDER);}
                return max;
            }
        else
            throw new NullPointerException();

    }
}

```