TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
A. Berlea, F. Forster, T. Gawlitza

SoS 2007
Lösungsvorschläge zu Blatt 1
Ausgegeben am: 20. April 2007
Abgabe bis: **27. April 2007**

Übungen zu Einführung in die Informatik II

Aufgabe 1 Asserts (Lösungsvorschlag)

Die Klasse `SearchTree`:

```
class SearchTree <E extends Comparable<E>>{
    private E key;
    private SearchTree<E> left , right;

    public SearchTree(E k){
                key = k;
        }

    public String toString(){
        return ""+key+"("+((left==null) ? "" : left+",")
                        +((right==null) ? "" : right)
                    +")";
    }

    public void insert(E e){
    if (key.compareTo(e) < 0)
            if (right != null) right.insert(e);
            else right = new SearchTree<E>(e);
        else if (key.compareTo(e) > 0)
            if (left != null) left.insert(e);
            else left = new SearchTree<E>(e);

        assert isSearchTree();
    }


    public SearchTree<E> search(E e){
        if (key.compareTo(e)<0) return (right == null) ? null : right.search(e);
        if (key.compareTo(e)>0) return (left == null) ? null : left.search(e);
        return this;
    }

    public static<E extends Comparable<E>> E min(E e1,E e2){
        return (e1 == null) || ((e2!=null) && e1.compareTo(e2) <= 0) ? e1 : e2;
    }
    public static<E extends Comparable<E>> E max(E e1,E e2){
        return (e1 == null) || ((e2!=null) && e1.compareTo(e2) <= 0) ? e2 : e1;
    }
    public E max(){
```

```
        E temp = (left==null) ? key : max(key,left.max());
        return (right==null) ? temp : max(temp,right.max());
    }
    public E min(){
        E temp = (left==null) ? key : min(key,left.min());
        return (right==null) ? temp : min(temp,right.min());
    }


    public boolean isSearchTree(){
        return
            ((left==null) ? true : (key.compareTo(left.max())>=0) && left.isSearchTree())
            &&
            ((right==null) ? true :
                             (key.compareTo(right.min())<=0) && right.isSearchTree());
    }

    public int height(){
        return 1 + Math.max(left==null ? 0 : left.height(),
                            right==null ? 0 : right.height());
    }

    public boolean isBalanced(){
        return
            Math.abs((left==null ? 0 : left.height()) -
                     (right==null ? 0 : right.height())) <= 1
            &&(left==null ? true : left.isBalanced())
            &&(right==null ? true : right.isBalanced());
    }

    public static SearchTree<Integer> random(int n){
        if (n<0) return null;
        java.util.Random generator = new java.util.Random();
        SearchTree<Integer> t = new SearchTree<Integer>(generator.nextInt());
        for (int i=1;i<n;i++)
            t.insert(generator.nextInt());
        return t;
    }
}
```

Eine Test-Klasse

```
class Umgebung{
    public static void main(String[] a){
        SearchTree<String> t = new SearchTree<String>("Muenchen");
        System.out.println(t);
        t.insert("Trier"); System.out.println(t);
        t.insert("Koeln"); System.out.println(t);
        t.insert("Berlin"); System.out.println(t);
        t.insert("Hamburg"); System.out.println(t);
        t.insert("Osnabrueck"); System.out.println(t);
        t.insert("Stuttgart"); System.out.println(t);
        System.out.println("Min␣in␣t:␣"+t.min());
        System.out.println("Max␣in␣t:␣"+t.max());
        SearchTree<String> t1 = t.search("Berlin");
        System.out.println("Berlin␣tree␣=␣"+t1);
        SearchTree<String> t2 = t.search("Bremen");
        System.out.println("Bremen␣tree␣=␣"+t2);

        System.out.println(t2.min(null,"Alfa"));
        System.out.println(t2.min("Alfa",null));
```

```
        SearchTree<Integer> t5;
        for(int i=10;i<1000;i=i+10){
            t5 = SearchTree.random(i);
/* Uncommenting this raises an AssertionError if assertions are enabled
            assert (i<100 || t5.isBalanced()) :
                            "Random tree with "+i+" elements is not balanced";
*/

        }
    }
}
```