Programmieraufgaben Binäre Suchbäume Lösungen

Aufgabe 4.3 – search()

Rekursive Lösung:

```
public boolean search(BinSearchTree node, int key) {
 if (node == null) return false;
 else if (key < node.getKey()) return search(node.getLeft(), key);
 else if (key > node.getKey()) return search(node.getRight(), key);
 else return true;
}
```

Iterative Lösung (geht für einfache Suche noch gut und spart den Aufwand für die Tail-Rekursion):

```
public boolean search(BinSearchTree node, int key) {
 while (node != null && node.key != key) {
   if (key < node.getKey()) node = node.getLeft();
   else node = node.getRight();
 }
 return node != null;
}
```

Aufgabe 4.5 – search() alle Elemente

```
public boolean search(BinSearchTree node, int key) {
 if (node == null) return false;
 else if (key < node.getKey()) return search(node.getLeft(), key);
 else if (key > node.getKey()) return search(node.getRight(), key);
 else {
   System.out.print("|");
   search(node.getRight(), key);
   return true;
 }
}
```

Aufgabe 4.8 – insert()

```
public BinSearchTree insert(BinSearchTree node, int key) {
 if (node == null) node = new BinSearchTree(key);
 else {
   System.out.print(node.getKey() + " ");
   if (key < node.key) node.setLeft(insert(node.getLeft(), key));
   else node.setRight(insert(node.getRight(), key));
 }
 return node;
}
```

Aufgabe 4.13 – delete()

Lösung bei der *ein* Element mit dem entsprechenden Schlüssel gelöscht wird:

```
public BinSearchTree delete(BinSearchTree node, int key) {
 if (node != null) {
  if (key < node.getKey()) node.setLeft(delete(node.getLeft(), key));
  else if (key > node.getKey()) node.setRight(delete(node.getRight(), key));
  else if (node.getRight() == null) node = node.getLeft();
  else if (node.getLeft() == null) node = node.getRight();
  else {
   BinSearchTree n = node.getRight(), p = null;
   while (n.getLeft() != null) {
    p = n;
    n = n.getLeft();
   }
   if (p != null) {
    p.setLeft(n.getRight());
    n.setLeft(node.getLeft());
    n.setRight(node.getRight());
   } else n.setLeft(node.getLeft());
   node = n;
  }
 }
 return node;
}
```

Die Musterlösung aus dem Leitprogramm löscht jeweils gerade *alle* Elemente mit dem entsprechenden Schlüssel. Das kann man so machen:

```
public BinSearchTree delete(BinSearchTree node, int key) {
 if (node != null) {
  if (key < node.getKey()) node.setLeft(delete(node.getLeft(), key));
  else if (key > node.getKey()) node.setRight(delete(node.getRight(), key));
  else if (node.getRight() == null) node = node.getLeft();
  else {
   if (node.getLeft() == null) node = node.getRight();
   else {
    BinSearchTree n = node.getRight(), p = null;
    while (n.getLeft() != null) {
     p = n;
     n = n.getLeft();
    }
    if (p != null) {
     p.setLeft(n.getRight());
     n.setLeft(node.getLeft());
     n.setRight(node.getRight());
    } else n.setLeft(node.getLeft());
    node = n;
   }
   node = delete(node, key);
  }
 }
 return node;
}
```