

## Selbsttest 4

### 1) Information hiding

Ein Student soll einen Suchbaum für Strings implementieren und soll dabei die Prinzipien des Information hiding anwenden. Der Student schlägt folgende Schnittstelle vor:

```
public class Tree {
    Node head;
    public Tree() { ... }
    public void insert(String name) { ... }
    public boolean search(String name) { ... }
    public boolean remove(String name) { ... }
}
public class Node {
    String name;
    Node left, right;
    Node(String name) { ... }
}
```

Hat der Student seine Arbeit gut gemacht, oder sind ihm Fehler unterlaufen?

Gibt es Fehler? Welche? Korrigieren Sie eventuelle Fehler. Wieviele Zeilen müssen Sie minimal ändern?

Gibt es Möglichkeiten die Sichtbarkeit von Variablen/Methoden weniger restriktiv zu gestalten und trotzdem die Aufgabenstellung zu erfüllen? Wieviele Zeilen können Sie so ändern?

### 2) Bäume

Ein balancierter Baum ist nicht unbedingt vollständig

In manchen Anwendungsfällen können lineare Listen bei der Suche effizienter sein als balancierte Suchbäume

In einem AVL-Baum unterscheidet sich die Anzahl der Knoten im linken Unterbaum maximal um 1 vom rechten Unterbaum

Die Auswertung eines arithmetischen Ausdrucks (dargestellt als Syntaxbaum) erfolgt "in order"

Gibt man die Knoten beliebiger Suchbäume "in order" aus, so ist die Ausgabe automatisch sortiert

Bei der Umwandlung von Mehrwegbäumen in Bruderbäume bleibt die logarithmische Suchzeit erhalten

Ja	Nein

### 3) Bruderbaum

Versuchen Sie einen binären Suchbaum (linker Unterbaum < Wurzel <= rechter Unterbaum) für Strings möglichst gut auf einen Bruderbaum abzubilden.

Gibt es wesentliche Unterschiede zwischen Binär- und Bruderbaum, sodass eine Abbildung schwierig werden könnte?

Geht bei Ihrer Lösung die logarithmische Suchzeit verloren?