

# Übung 9

## Entwurfsmuster „Fliegengewicht“, Fazit Entwurfsmuster, Software-Tests (Modultest, Black-Box Test)

1. Implementieren Sie ein Programm, das 10000000 Bäume (10 Millionen, Sie können die angegebene Vorlage verwenden) in einer virtuellen Landschaft „zeichnet“.<sup>1</sup>

```
public class Baum {
    public String name = "Baum";
    public String beschreibung = "Eine Pflanze, die " +
        "vorzugsweise im Wald vorkommt, einen Stamm und " +
        "grüne Blätter hat.";
    public void zeichne(int nummer, int x, int y) {
        // Hier könnte man auch was malen...
        System.out.print(name + " Nummer " + nummer +
            " ist fertig.\r");
    }
}
```

- (a) Nach der „klassischen“ Methode, bei der Sie pro Baum ein Baum-Objekt mit `new` erzeugen,
  - (b) sowie in einer zweiten Implementation als „Fliegengewicht“, bei dem Sie nur eine Instanz eines Baum-Objektes tatsächlich erzeugen.
  - (c) Vergleichen Sie, falls dies in Ihrer Java-Umgebung möglich ist, den Speicherverbrauch und die Laufzeitdauer beider Implementationen.
2. Welche(s) Entwurfsmuster würden Sie möglicherweise für eine Software verwenden, die folgendes realisieren soll:
    - (a) eine Datenbank für Bücher anlegen und verwalten,
    - (b) ein interaktives Zeichenprogramm,
    - (c) eine Fertigungsstraßensteuerung,
    - (d) einen Terminkalender,
    - (e) einen Taschenrechner?

Hinweis: Die Antworten machen natürlich nur mit einer entsprechenden (kurzen) Begründung Sinn.

---

<sup>1</sup>Sie brauchen die Bäume nicht wirklich zeichnen zu lassen, auch ein Koordinatensystem ist hier überflüssig.

3. Bauen Sie in die rekursive Implementierung des einfachen `int`-basierten Fakultät-Berechnungsprogramms aus der Vorlesung ein Testmodul ein, das für Eingabewerte von -1 bis +1000 den berechneten Funktionswert auf Plausibilität überprüft, indem einmal mit dem „Vorgängerwert“ und einmal mit dem „korrekten Wert“ (unter Zuhilfenahme einer `BigInteger`-Variante) verglichen wird.
4. Versuchen Sie, anhand eines **Blackbox** Tests mit verschiedenen Eingabe-Zeichenketten herauszufinden, was die statische Klassenmethode `public static String geheimnis(String eingabe)` aus der nur als `Compilat` vorliegenden Datei **Geheim.class** eigentlich tut. Sie können hierzu, um nicht für jeden Test neu compilieren zu müssen, auch wieder die Klasse `Eingabe.java` zur Hilfe nehmen, wie in diesem Code-Fragment gezeigt:

```
public class BlackboxTest {
    public static void main(String[] args) {
        for(;;) { // Endlosschleife
            String s1 = Eingabe.readString("Eingabe: ");
            String s2 = Geheim.geheimnis(s1);
            System.out.println("Ausgabe: " + s2);
        }
    }
}
```