

## Übungsblatt 5

*Besprechung in der Vorlesung am 31.3.2003*

### 1 Aufteilen von Dateien

1. Schreiben Sie sich eine Textdatei, in die Sie positive und negative ganze Zahlen eingeben. Dazu eignet sich jeder Texteditor.
2. Schreiben Sie dann ein Java-Programm, das diese Textdatei einliest und alle positiven Zahlen in einer Textdatei `positiv.txt` und alle negativen Zahlen in einer Datei `negativ.txt` speichert.
3. Erweitern Sie Ihr Programm so, dass nach dem Aufteilen der Daten die Dateien `positiv.txt` und `negativ.txt` an der Konsole ausgegeben werden.

### 2 Directory-Lister

1. Schreiben Sie eine Klasse `DirectoryLister`, die eine Methode enthält, die alle Dateinamen aus einem Verzeichnis auf der Konsole auflistet., z.B.:

```
public class DirectoryLister {  
    ...  
    public void doList(String directory) { ... }  
}
```

2. Geben Sie das Interface `ListerCallback` ein und speichern Sie es ab, so dass Sie es in Ihrem Programm verwenden können.

```
interface ListerCallback {  
    public void perform(File theFile);  
}
```

3. Modifizieren Sie die Klasse `DirectoryLister` so, dass bei ihrem Konstruktor eine Referenz auf ein Objekt, das `ListerCallback` implementiert, übergeben werden kann. Also wird Ihr neuer Konstruktor der Klasse `DirectoryLister` so aussehen:

```
public DirectoryLister (ListerCallback callback) { ... }
```

Ändern Sie den Rest der Klasse so, dass zum Ausgeben der Dateien die `perform()`-Methode von `ListCallback` verwendet wird. Also sollte irgendwo in Ihrem Programm für jede Datei, die Sie ausgeben wollen, einmal `ListCallback.perform(...)` aufgerufen werden.

**Achtung:** Sie können Ihr Programm im Moment zwar kompilieren, das Ausführen geht aber erst nach dem nächsten Aufgabenteil!

4. Wenn Sie Ihr Programm nun ausführen wollen, brauchen Sie ja ein Objekt, das `ListCallback` implementiert. Nehmen Sie dazu die folgende Klasse.

```
public class CallbackTest implements ListCallback {
    public void perform(File theFile) {
        System.out.println(theFile.getName());
    }
}
```

Ändern Sie Ihr Programm so, dass beim Erzeugen der Instanz von `DirectoryLister` eine Referenz auf ein Objekt vom Typ `CallbackTest` übergeben wird. Nun sollten Sie Ihr Programm ausführen können und es sollten alle Dateinamen auf der Konsole erscheinen.

### 3 Lines of Code

1. Schreiben Sie eine Klasse `InspectCode`, die das Interface `ListCallback` aus der zweiten Aufgabe implementiert. Der `perform()`-Methode sollen aber nun nur Textdateien vom Typ „.java“ übergeben werden. Nehmen Sie die dazu nötigen Änderungen in der Klasse `DirectoryLister` aus der zweiten Aufgabe vor. Als Ergebnis soll die `InspectCode.perform()`-Methode die Anzahl der Zeilen in dieser Datei ausgeben.

2. **Für Hartgesottene:**

Modifizieren Sie die Klasse `InspectCode`, so dass bei der Zählung der Zeilen Kommentare in Java-Programmen nicht berücksichtigt werden.

**Hinweis:** Einzeilige Kommentare beginnen mit „//“, dann ist der Rest der Zeile Kommentar. Mehrzeilige Kommentare beginnen mit „/\*“. Dann wird das Kommentarende durch „\*/“ markiert.

3. **Für ganz Hartgesottene:**

Modifizieren Sie das gesamte Programm so (Sie dürfen also alle Klassen verändern), so dass ausgehend von einem Startverzeichnis rekursiv alle Java-Dateien (Endung „.java“) auch in allen Unterverzeichnissen untersucht und die Lines-of-Code als Gesamtsumme ermittelt werden.

**Tipp:** Wenn Sie in `DirectoryLister` auf ein Verzeichnis stoßen, können Sie den Pfad zu den Dateien in diesem Verzeichnis erzeugen, indem Sie an den bisherigen Pfad einen Slash („/\“) anhängen und dann den Namen des

aktuellen Verzeichnisses. Mit dieser Information können Sie nun sofort wieder den `DirectoryLister` aufrufen.

**2. Tipp:** Ergänzen Sie die `ListCallback.perform()`- Methode und `DirectoryLister.doList()` um geeignete Rückgabewerte, so dass das Aufsummieren der Lines of Code möglich wird.