# 5 Structural Recursion vs. Accumulators; Abstractions

## Portfolio Problems

### 5.1 Problem

Exercise 15.10 on page 175 in the text deals with shopping lists.

1. Design the method that produces a shopping list sorted by the brand names. The class *String* defines the following method:

   ```
   // this String is lexicographically before that String: positive result
   // this String is same as that String: result is 0
   // this String is lexicographically after that String: negative result
   int compareTo(String that);
   ```

2. Design the methods *howMany*, *brandList*, and *highestPrice* as described in the Exercise 15.10.

3. Rewrite the methods *howMany*, *brandList*, and *highestPrice* using accumulator style.

### 5.2 Problem

Problem 4.5 asked you to design data definitions for files of images, text, and sounds.

Problem 14.6 asked you to add methods *timeToDownload*, *smallerThan*, and *sameName*. Do it, if you have not done so before.

Then design the appropriate abstract class that eliminates as much repetition in the code as possible. ∎

## Main Assignment

### 5.3 Problem

Complete Part 1 of the Lab 5, dealing with bank accounts.

Hand in (as two separate files) both the original solution and the solution after you have designed all abstractions. ∎

## 5.4 Problem

Complete Part 2 of the Lab 5, converting the loops to the accumulator style.

Hand in (as two separate files) both the original solution and the solution after you have designed the loops with accumulators. ∎