Exercise sheet 3
for September 22, 2004

You should try to solve at least the exercises 5.1 – 5.3, 5.8 – 5.10, 5.12, and 5.13 from H&R Chapter 5, the exercises 6.1 and 6.3 from Chapter 6 and exercise E3.1 and E3.2.

You are strongly encouraged to hand in solutions to exercise 6.3 and E3.2.

Exercise E3.1 (Merging lists of integers)

(a) Declare an SML function

    merge : int list * int list -> int list

    that takes two sorted lists of integers and merges them into a sorted list of integers.

(b) Implement a similar Java method

    static int[] merge(int[] xs, int[] ys)

    that takes two sorted arrays of integers and merges them into a (new) sorted array of integers.

(c) The goal of this exercise is to use Lists of Integers instead of arrays as in b. But this appears to be rather cumbersome—at least if you try to use iterators to traverse the lists—so I recommend that you skip to the next exercise.

    Implement a Java method

    static LinkedList merge(List xs, List ys)
that takes two sorted lists of Integer objects and merges them into a sorted List of Integer objects. The method should build a new LinkedList, and should not modify the given Lists. The interface List and the class LinkedList are from the java.util package.

Two Lists xs and ys of Integer objects may be built like this:

```java
List xs = new LinkedList();
xs.addLast(new Integer(3));
xs.addLast(new Integer(5));
xs.addLast(new Integer(12));
List ys = new LinkedList();
ys.addLast(new Integer(2));
ys.addLast(new Integer(3));
ys.addLast(new Integer(4));
ys.addLast(new Integer(7));
```

**Exercise E3.2 (Flatten lists)** Declare a function

```haskell
flatten : 'a list list -> 'a list
```

that takes a list of lists and gives a list of all the elements in the lists. For example:

```haskell
flatten [[1, 2], [3], [], [4, 42, 23]] = [1, 2, 3, 4, 42, 23]
```