Exercise sheet 4
for September 29, 2004

2004-10-20

You should try to solve at least the exercises 7.1–7.3 and 7.6 from H&R Chapter 7, exercise 8.5 and 8.4 from Chapter 8, and the exercises E4.1–E4.3. After that, try exercise 7.7 and 7.8 from H&R Chapter 7.

We urge you to hand in the solution to exercise E4.3.

Exercise E4.1 (List of Fibonacci numbers)  Declare an SML function

\[
\text{fibs : int -> int list}
\]

that for a given \(n\) gives the list of Fibonacci numbers \([F_n, F_{n-1}, F_{n-2}, \ldots, F_1, F_0]\). Example:

\[
\text{fibs 4} = [3,2,1,1,0]
\]

Arrange the computation so that each Fibonacci numbers is only computed once. See exercise 1.5 in H&R for information about Fibonacci numbers.

Exercise E4.2 (Modelling fonts)  In Cascading Style Sheets (CSS) a font is described by the attributes: \textit{family}, \textit{style}, \textit{variant}, \textit{weight}, and \textit{size}.

A font family is the name of a specific font or it is one of the generic families: \textit{serif}, \textit{sans-serif}, \textit{cursive}, \textit{fantasy}, or \textit{monospace}.

A font style is one of \textit{normal}, \textit{italic}, or \textit{oblique}.

A font variant is either \textit{normal} or \textit{small-caps}.

A font weight is one of \textit{normal}, \textit{bold}, \textit{bolder}, \textit{lighter}, or one of the numbers 100, 200, 300, 400, 500, 600, 700, 800, 900.
A font size is one of: an absolute size: xx-small, x-small, small, medium, large, x-large, xx-large; a relative size: larger or smaller; a length size in points or pixels, for example 12pt; a percentage size, for example 90%.

(a) Declare SML types to model a CSS font.

(b) Write down the value for (at least) a bold small italic sans-serif small-caps font.

(c) Declare a predicate that is true if a font is larger than medium. The length size 12pt (or 17px) corresponds to medium.

(d) Declare a function that makes a font one step larger. Only fonts with an absolute or length size is affected (an xx-large font that is made one step larger is still xx-large).

(e) Declare a function that takes a list of fonts and make all the fonts in the list one step larger.

(f) Declare a function that takes a list of font attributes and returns a font. Not all attributes need to be present in the list, if an attribute is not present the function should chose a suitable default attribute. If the same kind of attribute is specified more than once in the list (e.g., a font is specified to be both serif and monospace) then the resulting attribute is undefined (that is, you chose).

(g) (Optional) Declare a function that takes a list of font attributes and returns a list of possible fonts. Again, like in (f), if an attribute is missing from the list the function should chose a suitable default attribute.

**Exercise E4.3 (Playing with trains)**

Given the datatype declarations:

```sml
datatype carriage =
  Passenger of int
| Goods of real

infix 5 :--:
datatype train =
  Engine of string * int
| :--: of train * carriage
```

Declare functions to:

(a) Add up the total passenger capacity of a train.

(b) Add up the total goods capacity of a train.
(c) Do both the above simultaneously returning a pair.

(d) Checks that a train’s weight does not exceed the engine’s capacity. A passenger carriage weights three and a half tonnes plus the weight of the passengers and a goods carriage weights five tonnes. An average passenger (with luggage) weights 120 kg.

(e) Take two trains and sort them so that one engine gets all the passenger carriages and the other gets all the goods carriages.