DTU Compute 3/10 2016
02157 Functional Programming Michael R. Hansen

Exercise: File System

In this exercise set you are asked to revise/extend the example: *File system* on pages 140 - 142 in the textbook. The first step is to revise the declaration for `Element` (and `FileSys`) so that a file is characterized by a name and an *extension*, where both are strings. For example, `File("a2","fsx")` should be a value of type `Element` denoting a file with name `a2` having the extension `fsx`, and the following should be a valid declaration:

```
let d1 = Dir("d1",[File("a1","java");
                   Dir("d2", [File("a2","fsx");
                              Dir("d3", [File("a3","fs")])]);
                   File("a4","fsx");
                   Dir("d3", [File("a5","pdf")])]);;
```

The revised type declarations are used in each of the following exercises.

`ListOfNames`: Revise the functions `namesFileSys` and `namesElement` so that they extract the list of all file names (with extensions) and names of directories occurring in file systems and elements, respectively. For example: the name of the file `File("a2","fsx")` is the string `"a2.fsx"` and

```
namesElement d1;;
val it : string list = ["d1"; "a1.java"; "d2"; "a2.fsx"; "d3";
                        "a3.fs"; "a4.fsx"; "d3"; "a5.pdf"]
```

The order in which the strings occur in the list is of no importance.

`search`: Declare two functions `searchFileSys` *ext filesys* and `searchElement` *ext elem* in mutual recursion that can extract the *set* of all file names having extension *ext* in a file system or element, respectively. Just sets of file names *without extensions* are returned by the two functions, e.g.:

```
searchElement "fsx" d1;;
val it : Set<string> = set ["a2"; "a4"]
```

`longNames`: Declare mutually recursive functions:

```
longNamesFileSys: FileSys -> Set<string>
longNamesElement: Element -> Set<string>
```

to extract the set of so-called *long file names* of all files occurring in file systems and elements, respectively. A long file name is a string consisting of a *path* part and a file name. It has the form $dir_1\backslash dir_2\backslash \cdots dir_n\backslash name.ext$, when the file *name* with extension *ext* is in the subdirectory named dir_n and dir_{i+1} is a subdirectory of dir_i, for $1 \leq i < n$. If a file is an element of the top-most file system, then the corresponding path part is the empty string. For example, the long name for the file `a3.fs` in the above element `d1` is the string `"d1\d2\d3\a3.fs"`. (Note that the backslash character is written using the escape sequence `"\\"`.)