CMSC 22610	Implementation of	Lab 1
Winter 2009	Computer Languages - I	January 14, 2009

Lab 1: Records, Datatypes, and Pattern Matching

Due: ∅

Welcome again! Datatypes and pattern matching are the champagne and caviar of ML programming. If you don't yet believe that, do this lab, and you will find your beliefs altered.

1 Files and Directories

Here is a datatype for a description of files and directories. It's recursive and shapely.

```
datatype file
  = F of {base: string,
          ext : string,
          size: int} (* size in bytes *)

datatype file_or_dir
  = File of file
  | Dir  of {name: string,
             contents: file_or_dir list}
```

Note that `base` and `ext` (for "extension") are standard names for the parts of a filename before and after the dot, respectively. Here is an example `file_or_dir` value.

```
val mm =
  Dir {name="home",
       contents=
        [Dir {name="mork",
              contents=
               [File (F {base="strawberry-fields",
                         ext="mp3",
                         size=14144}),
                File (F {base="mindy",
                         ext="jpg",
                         size=2000})]},
         Dir {name="mindy",
              contents=
               [File (F {base="mindy",
                         ext="vcf",
                         size=128}),
                Dir {name="pics",
                     contents=
                      [File (F {base="cat",
                                ext="gif",
                                size=1029}),
                       File (F {base="dog",
                                ext="png",
                                size=555})]}]}]}
```

Write the functions corresponding to the following `val` specifications. Use pattern matching as liberally as possible but no more liberally. In particular, use underscores and record elisions (three dots) for values that need not be named. You should be able to infer what the following functions should compute by their names, types and comments.

```
val totalSize  : file_or_dir -> int (* size in bytes *)
val numFiles   : file_or_dir -> int
val dirNames   : file_or_dir -> string list

val images     :           file_or_dir -> file list
val biggerThan : int    -> file_or_dir -> file list
val filesNamed : string -> file_or_dir -> file list

val fullPaths  : file_or_dir -> string list
```

The last function should evaluate on `mm` (from above) to the following:

```
["/home/mork/strawberry-fields.mp3",
 "/home/mork/mindy.jpg",
 "/home/mindy/mindy.vcf",
 "/home/mindy/pics/cat.gif",
 "/home/mindy/pics/dog.png"]
```

2 Errata

Revision 0 (Jan 14)

- None so far!