Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ciencias Exactas y Naturales
Departamento de Ciencia de la Computación
Estructura de Datos y Algoritmos II

Nombre y Apellido: Legajo:

Examen Final

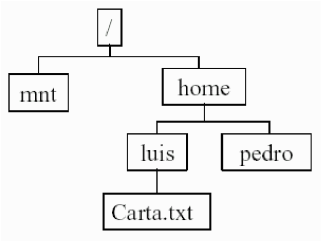
1. Dada una secuencia con pares de la forma (Artículo, Precio) ordenada cronológicamente, se desea saber cuál fue el artículo que más cantidad de aumentos tuvo a lo largo del tiempo. Suponiendo que se definió un tipo de datos *Art* para representar los distintos artículos , los cuales pueden compararse, definir una función $aumentos : Seq\ (Art, Int) \to (Art, Int)$, que resuelva el problema calculando el artículo que más cantidad de aumentos tuvo y la cantidad de aumentos.
Definir la función con profundidad de orden $O(\lg n)$, donde n es el largo de la secuencia.
Por ejemplo,

$$| \ aumentos\langle(a,5),(b,4),(a,4),(a,30),(b,7),(b,10)\rangle = (b,2) \ |$$
$$| \ aumentos\langle(a,5),(a,6),(b,1),(a,3),(a,4),(b,2)\rangle = (a,2) \ |$$

2. Se utilizará el siguiente tipos de datos para representar un árbol de directorios en un sistema de archivos.

```
type Name = String
data DirTree a = Dir Name [DirTree a]   -- nombre de un directorio y su contenido
               | File Name a            -- nombre de un archivo y contenido
```

donde la raíz de un árbol de directorios debe ser `"/"` y no puede haber dos directorios o archivos con el mismo nombre en el mismo path.
Por ejemplo, para representar el árbol de directorio:



definimos dirtree:

```
dirtree = Dir "/" [Dir "home" [Dir "luis" [File "Carta.txt" "xxx"], Dir "Pedro" []], Dir "mnt" []]
```

Además se utilizará el tipo Path para representar una ruta absoluta:

```
type Path = [String]
```

Por ejemplo el path `"/home/pepe"` se reprerenta con la lista `["home","pepe"]`, mientras que la raíz `"/"` con `[]`.
Definir en Haskell las siguientes funciones para manejar esta estructura de directorios.

a) `names :: [DirTree a] → [Name]`, dada una lista de árboles de directorios devuelve una lista con los nombres correspondientes a archivos o directorios que están en la raíces de los árboles.

Por ejemplo,

`names [Dir "home" [Dir "luis" [File "Carta.txt" "xxx"], Dir "Pedro" []], Dir "mnt" []] = ["home","mnt"]`

b) Para crear un directorio vacío en un path, siempre que exista el path y no haya un archivo o directorio con el mismo nombre, se definió la siguiente función:

```
mkDir :: Path → Name → DirTree a → DirTree a
mkDir p n d = mkdir ("/" : p) n d
```

Completar con la definición de mkdir.

c) Para listar los archivos y directorios dentro de una ruta dada se definió:

$$\mathsf{ls} :: \mathsf{Path} \to \mathsf{DirTree}\ a \to [\mathsf{Name}]$$
$$\mathsf{ls}\ p\ t = \mathsf{ls}\ (\texttt{"/"} : p)\ t$$

Completar la definición de ls definiendo ls.

Por ejemplo,

$$\mathsf{ls}\ [\,]\ \mathsf{dirtree} = [\texttt{"home"}, \texttt{"mnt"}]$$
$$\mathsf{ls}\ [\texttt{"home"}]\ \mathsf{dirtree} = [\texttt{"luis"}, \texttt{"pedro"}]$$

3. (Sólo libres) Dadas las siguientes definiciones:

$$\begin{aligned}
&\mathsf{elem}\ x\ [\,] = \mathit{False}\\
&\mathsf{elem}\ x\ (y : ys) = x \equiv y \vee \mathsf{elem}\ x\ ys\\
&\mathsf{concatMap}\ f = \mathsf{concat} \circ \mathsf{map}\ f\\
&\mathsf{allNames}\ (\mathsf{Dir}\ n\ xs) = n : \mathsf{concatMap}\ \mathsf{allNames}\ xs\\
&\mathsf{allNames}\ (\mathsf{File}\ n\ x) = [n]\\
&\mathsf{isName}\ y\ (\mathsf{Dir}\ n\ xs) = y \equiv n \vee \mathsf{or}\ (\mathsf{map}\ (\mathsf{isName}\ y)\ xs)\\
&\mathsf{isName}\ y\ (\mathsf{File}\ n\ x) = y \equiv n
\end{aligned}$$

Probar por inducción estructural sobre t que:

$$\mathsf{elem}\ n\ (\mathsf{allNames}\ t) = \mathsf{isName}\ t$$

para cualquier $n :: \mathsf{String}$.
Ayuda: Puede usar la siguiente propiedad $\mathsf{elem}\ x\ (\mathsf{concat}\ xss) = \mathsf{or}\ (\mathsf{map}\ (\mathsf{elem}\ x)\ xss)$, sin demostrarla.