E.T.S.I. Informática. Universidad de Málaga
Examen Ordinario, 26 de Enero de 2006
Programación Declarativa, Prog. Funcional (3º)

APELLIDOS: **SOLUCIONES**

NOMBRE: _____

PUNTOS	1-1	1-2	1-3	1-4	2-1	2-2	2-3	3-1
	3.0	2.0	3.0	2.0	2.0	5.0	3.0	10.0

☐ si
☐ no
} publique mi calificación si fuera negativa

Consideremos la siguiente estructura para representar los números enteros

$$\textbf{data}\ E\ =\ O \mid S\ E \mid M\ E\ \textbf{deriving}\ Show$$

Por ejemplo, -1 se puede representar en la forma $(M(SO))$ o también en la forma $S(M(S(SO)))$. Sea además la siguiente función de plegado:

$$\begin{aligned}
&pliega\ ::\ (a\ \to\ a)\ \to\ (a\ \to\ a)\ \to\ a\ \to\ E\ \to\ a\\
&pliega\ s\ m\ z\ O\ \ \ \ \ \ =\ z\\
&pliega\ s\ m\ z\ (S\ e)\ \ =\ s\ (pliega\ s\ m\ z\ e)\\
&pliega\ s\ m\ z\ (M\ e)\ =\ m\ (pliega\ s\ m\ z\ e)
\end{aligned}$$

1-1 Define la función *par* (que comprueba si un entero es par) directamente a través de *pliega*:

SOL

$$\begin{aligned}
&par\ ::\ E\ \to\ Bool\\
&par\ =\ pliega\ not\ id\ True
\end{aligned}$$

1-2 Sea ahora la siguiente definición para la suma de enteros:

$$\begin{aligned}
&\textbf{instance}\ Num\ E\ \textbf{where}\ x\ +\ O\ \ \ \ \ \ =\ x\\
&\qquad\qquad\qquad\qquad\ x\ +\ (S\ y)\ \ =\ S\ (x+y)\\
&\qquad\qquad\qquad\qquad\ x\ +\ (M\ y)\ =\ M(M\ x\ +\ y)
\end{aligned}$$

Prueba que la igualdad $z + MO = z$ no es demostrable.

SOL Basta comprobar un caso. Por ejemplo, la ecuación $O+MO = O$ no es demostrable ya que la forma normal del miembro izquierdo es $M(MO)$, que no coincide con la forma normal de O.

1-3 Un dato entero (de tipo E) está normalizado si es, o bien O, o bien de la forma $S^nO(n > 0)$, o bien de la forma $MS^nO(n > 0)$. . Completa la siguiente función que permite normalizar un entero:

SOL

$$\begin{aligned}
&normaliza\ ::\ E\ \to\ E\\
&normaliza\ O\ =\ O\\
&normaliza\ (S\ x)\ =\textbf{case}\ (normaliza\ x)\ \textbf{of}\ M(S\ O)\ \to\ O\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ M(S\ y)\ \to\ M\ y\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ y\ \ \ \ \ \ \ \to\ S\ y\\
&\\
&normaliza\ (M\ x)\ =\textbf{case}\ (normaliza\ x)\ \textbf{of}\ O\ \ \ \ \to\ O\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ S\ y\ \ \to\ M\ (S\ y)\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ M\ y\ \to\ y
\end{aligned}$$

Es fácil demostrar por inducción sobre e, que *normaliza* e solamente tiene una de las formas indicadas: o bien O, o bien $S(S(...(SO)...))$, o bien $M(S(S(...(SO)...)))$. En efecto, el caso base es $e \equiv O$, y tenemos que *normaliza* $O \equiv O$. Sea ahora $e \equiv Me'$. En este caso, calculamos *normaliza* e'; si es de la forma Sy, por HI sabemos que $y \equiv S^nO(n \geq 0)$, de donde $M(Sy)$ tiene la forma $M(S(...(SO)...)))$; si *normaliza* $e' \equiv My$, entonces y es de la forma $S^nO(n > 0)$, que es precisamente el valor de *normaliza* e. Y finalmente si *normaliza* e' es O, éste es el valor de *normaliza* e'. El caso $e \equiv Se'$ es similar.

1-4 Define una instancia de la igualdad que permita demostrar por inducción $x + MO == x$:

SOL

$$\begin{array}{l}\textbf{instance}\ Eq\ E\ \textbf{where}\ x\ ==\ y\ =\ normaliza\ x\ ='\ normaliza\ y\ \textbf{where} \quad -- \ =' \text{ es la igualdad estructural}\\ \quad\quad O\ \ ='\ O\ \ =\ True\\ \quad\quad S\ x\ ='\ S\ y\ =\ x\ ='\ y\\ \quad\quad M\ x\ ='\ M\ y\ =\ x\ ='\ y\\ \quad\quad \_\ \ ='\ \_\ \ =\ False\end{array}$$

En este caso es demostrable la igualdad $z + MO == z$, ya que $z + MO$ se reduce a $M(Mz)$, de donde la igualdad $z+MO == z$ se reduce a $normaliza(M(Mz)) =' normaliza\ z$, y esta igualdad estructural se comprueba dependiendo del valor normalizado de z:

(1) $normaliza\ z \equiv O$; entonces $normaliza(M\ z)$ es O, y por lo tanto $normaliza(normaliza\ z)$ es O;

(2) $normaliza\ z \equiv S\ y'$, entonces $normaliza(M\ z) \equiv M(Sy')$, de donde $normaliza(M(M\ z) \equiv Sy'$;

(3) $normaliza\ z \equiv M\ y'$, entonces $normaliza(M\ z) \equiv y'$, pero como tanto y' como My' están normalizados, entonces, y' es necesariamente de la forma $S\ u$, de donde $normaliza(M(M\ z)) \equiv M(S\ u) \equiv M\ y'$.

Consideremos las siguientes definiciones

$$\begin{array}{ll}bmap\ f\ g\ [\,] & = [\,]\\ bmap\ f\ g\ (x : xs) & = f\ x\ :\ g\ x\ :\ bmap\ f\ g\ xs\end{array}$$

$$\begin{array}{ll}length\ [\,] & = 0\\ length\ (x : xs) & = 1\ +\ length\ xs\end{array}$$

2-1 Infiere el tipo más general de la función *bmap*.

SOL *bmap* tiene tres argumentos; las expresiones $f\ x$, $g\ x$, indican que tanto f como g son funciones cuyos tipos son de la forma $a \to b$, siendo a el tipo de x, y b el tipo base asociado a la lista resultado. En este caso, el tercer argumento de *bmap* es de tipo $[a]$, y el resultado también; luego $bmap :: (a \to b) \to (a \to b) \to [a] \to [b]$.

2-2 Demuestra por inducción sobre listas que *bmap* duplica la longitud de una lista:

$\forall\ xs\ \centerdot\ xs :: [a]\ \centerdot\ length\ (bmap\ f\ g\ xs)\ =\ 2 * length\ \ xs$.

SOL Caso base ($xs \equiv []$):

$length(bmap\ f\ g\ []) = 2 * length[]$

$\equiv_{\cdot:\cdot}$ 1ª) *bmap*, 1ª *length*

$length[] = 2 * 0$

$\equiv_{\cdot:\cdot}$ 1ª) *length*, aritmética

$0 = 0$

Paso inductivo:

$length(bmap\ f\ g\ (x : xs)) = 2 * length(x : xs)$

$\equiv_{\cdot:\cdot}$ 2ª) *bmap*, 2ª *length*

$length(f\ x : g\ x : bmap\ f\ g\ xs) = 2 * (1 + length\ xs)$

$\equiv_{\cdot:\cdot}$ 2ª) *length* aplicada dos veces, aritmética

$1 + 1 + length(bmap\ f\ g\ xs) = 2 + 2 * length\ xs$

$\Leftarrow$

HI

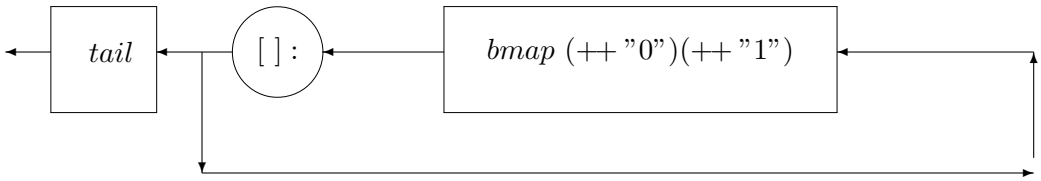
2-3 Describe la función bmap utilizando solamente la función estándar de plegado de listas *foldr*:

$$bmap'\ f\ g\ =\ foldr\ (\lambda\ x\ u\ \to\ f\ x\ :\ g\ x\ :\ u)\ []$$

3-1 Un número binario podemos representarlo como una lista no vacía de elementos del conjunto de caracteres $\{'0',\ '1'\}$. Describe una red de procesos para generar la lista infinita de binarios ordenada por número de bits y por valor: ["0", "1","00", "01", "10", "11", "000", "001", "010", "011", "100", "101", "110", "111", ...
(Ayuda: Usa la función *bmap* del apartado 2-1).

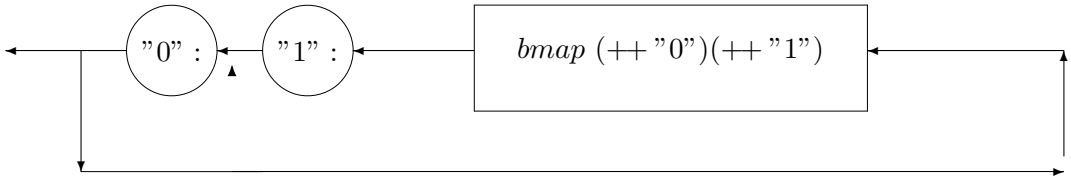
SOL Una primera solución es la siguiente:

$$bins\ =\ tail\ bs\ where\ bs\ =\ [\,]\ :\ bmap\ (++\ "0")\ (++\ "1")bs$$



Otra solución ligeramente distinta la proporciona el siguiente código y gráfico:

$bins \;=\; \text{"0"} : \text{"1"} \;:\; bmap\;(++\,\text{"0"})\;(++\,\text{"1"})bins$



3-2 Completa la siguiente función para el cálculo del mínimo común múltiplo de los elementos de una lista de naturales positivos:

SOL

$$
\begin{aligned}
&mcm \;::\; [Integer] \;\rightarrow\; Integer\\
&mcm\;(x : xs) \;=\; head\;[\; m \mid m \;\leftarrow\; [x,\; 2 * x\;..\;],\\
&\qquad\qquad\qquad\qquad and\;[\; m\;`mod`\;y == 0 \mid y \;\leftarrow\; xs\;]\;]
\end{aligned}
$$