E.T.S.I. Informática. Universidad de Málaga
19 de Febrero de 2007
Programación Declarativa (Prog. Funcional)

APELLIDOS: _____

NOMBRE: _____

PUNTUACIONES:

1	2	3	4	total
2.0	*4.0*	*2.0*	*2.0*	*10.0*

☐ sí
☐ no
} deseo que se publique mi calificación si fuera negativa

---

1 Completa la siguiente función sabiendo que debe calcular la mediana de una tupla de 5 valores

$$
\begin{array}{l}
mediana :: (Integer, Integer, Integer, Integer, Integer) \rightarrow Integer \\
mediana\ (x,y,z,t,u) \mid x > z \quad = mediana\ (z,y,x,t,u) \\
\qquad\qquad\qquad\quad \mid y > z \quad = \ldots \\
\qquad\qquad\qquad\quad \mid z > t \quad = mediana\ (x,y,t,z,u) \\
\qquad\qquad\qquad\quad \mid \ldots \quad = \ldots \\
\qquad\qquad\qquad\quad \mid otherwise\ = z
\end{array}
$$

y da un razonamiento basado en conjuntos bien construidos para probar su corrección.

---

2 Consideremos las siguientes declaraciones para representar y manipular los números naturales:

data $N = O \mid S\ N$ **deriving** *Show*

$\_1 = S\ O;\ \_2 = S\ \_1;\ \_3 = S\ \_2;\ \_4 = S\ \_3$

instance *Num* N **where**

$$
\begin{array}{ll}
y + O = y & itera\ g\ z\ O = z \\
y + S\ x = S\ (y + x) & itera\ g\ z\ (S\ n) = g\ n\ (itera\ g\ z\ n) \\
y * O = O & \\
y * S\ x = y * x + y &
\end{array}
$$

A Demuestra que el tipo de *itera* es $itera :: \ldots.$
En efecto:

B Sabiendo que n está en forma normal (FN), calcula las FNs de las expresiones

$n + \_1$ $\qquad\qquad$ $itera\ (const\ S)\ n\ \_1$

C Demuestra por inducción que se verifica $\forall n \,.\, n :: N \,.\, \_1 * n = n$

D Define la función $par :: N \rightarrow Bool$ y la suma $(+)$ directamente a través de *itera*:

$par = itera \ldots$ $\qquad\qquad$ $(y+) = itera \ldots$

E.T.S.I. Informática. Universidad de Málaga
Parcial 4º, 19 de Febrero de 2007
Programación Declarativa (Prog. Funcional)

APELLIDOS: _____

NOMBRE: _____

PUNTUACIONES:

3	4	total
5.0	*5.0*	*10.0*

☐ sí
☐ no

} deseo que se publique mi calificación si fuera negativa

---

3 Pretendemos escribir el algoritmo *quicksort con dos pivotes*:
1.- tomar los dos primeros elementos de la lista en el orden $x \leq y$.
2.- *repartir* el resto de la lista en tres segmentos $m1$, $m2$ y $m3$, donde $m1$ contenga los menores que x, $m2$ los comprendidos entre x e y (excluido y), y finalmente $m3$ los restantes.
3.- ordenar por separado cada segmento y "juntarlos"de forma que resulte la lista original pero ordenada.

A Describa una función para repartir una lista según dos valores x y y, con $x \leq y$:

Main > *repartir* 3 7 [1, 5, 3, 9, 2, 7]
([1, 2], [5, 3], [9, 7])

```
repartir :: Ord a => a → a → [a] → ([a], [a], [a])
repartir x y [ ] = ([ ], [ ], [ ])
  — si x ≤ y
repartir x y (u : us) ...
```

B Describa el algoritmo *quicksort* que ordena según el esquema anterior:

```
quicksort :: Ord a => [a] → [a]
...
```

---

4 Se considera la sucesión $\{f_n\}_{n\geq 0}$ definida en forma inductiva en la forma siguiente:

$$f_0 = 1, \qquad f_1 = -1, \qquad \text{y para } n \geq 0, f_{n+2} = (n+1)f_n + nf_{n+1}$$

A Describa una red de procesos que tenga como salida la lista infinita $[f_0, f_1, f_2, ...]$. Escriba así mismo la ecuación HASKELL correspondiente

B Use la red anterior para comprobar que la salida es la lista cíclica $[1, -1, 1, -1, 1, -1, ...]$