E.T.S.I. Informática. Universidad de Málaga — Apellidos: ______________________

18 de Febrero de 2008

Programación Declarativa (Prog. Funcional) — Nombre: ______________________

Puntuaciones:

1	2	3	4	5	6	total
0.5+1.0+0.5+1.0	*1.0*	*1.5*	*1.0*	*1.5*	*1.5 + 0.5*	*10.0*

- 1º y 3º: $\boxed{1,4,5,6}$ (puntos 7.5)
- 2º y 3º: $\boxed{1D,3,5,6}$ (puntos 6)
- solo 3º : $\boxed{4,5,6}$ (puntos 4.5)
- Todo : $\boxed{1,2,3,5,6A}$ (puntos 8.5)

---

1 Consideremos las siguientes declaraciones para representar y manipular los números enteros:

data $E \;=\; O \mid S\,E \mid P\,E$; $uno \;=\; S\,O$; $menos\_dos \;=\; P\,(P\,O)$; ...

$$\begin{array}{lll} doble & = & itera\ (S.S)\ (P.P)\ O \\ rap & = & itera\ not\ not\ True \\ nag & = & itera\ P\ S\ O \end{array} \qquad \begin{array}{lll} itera\ s\ p\ z\ O & = & z \\ itera\ f\ g\ z\ (S\ n) & = & f\ (itera\ f\ g\ z\ n) \\ itera\ f\ g\ z\ (P\ n) & = & g\ (itera\ f\ g\ z\ n) \end{array}$$

A Calcula la FN de $doble(P\ uno)$ ¿Es igual a O?

B Demuestra que el tipo de *itera* es $itera ::\ \ldots.$

En efecto:

C ¿Qué computan las funciones *rap* y *nag*?

D Demuestra las siguientes propiedades universales: ♡

$$\begin{cases} doble\ O = O & rap\ O = True \\ doble(S\ x) = S(S(doble\ x)) & rap(S\ x) = not(rap\ x) \\ doble(P\ x) = P(P(doble\ x)) & rap(P\ x) = not(rap\ x) \end{cases}$$

y utilízalas para demostrar por inducción sobre enteros $\forall\, n\ .\ n :: E\ .\ rap\ (doble\ x)\ ==\ True$

---

2 Define la función $valor : Integer \to Integer$ que toma un natural y calcula la suma de sus cifras (en base 10) de forma reiterada hasta conseguir un entero menor que 10. Por ejemplo $2741 \xrightarrow[2+7+4+1]{} 14 \xrightarrow[1+4]{} 5$; así $valor\ 2741 = 5$.

---

3 Las cartas de una baraja contienen dos números naturales (uno en cada cara) cuya suma es 9. Una $Mano$ es una colección (lista) de 4 cartas: p.e. [2,3,1,5], representa la misma mano que $[4, 6, 1, 2]$. Una jugada es una lista que representa los valores visibles de las cartas de una mano puestas sobre la mesa; los puntos de una $Jugada$ se calculan con en el Ejercicio 2: $valorJugada[2, 3, 1, 5] = valor\ 2315 = 2$.

type $Mano \;=\; [Int]$; **type** $Jugada \;=\; [Int]$; $valorJugada\ :\ Jugada \to Int$

Escribe las siguientes funciones

$posibles :: Mano \rightarrow [Jugada]$
— calcula las lista de jugadas para una $Mano$
$posibles\ [a, b, c, d] \ = \ [\,[x, y, z, t]\,|$
$x <\!\!-[a, \ldots$
$\ldots$

$valorJugada : Jugada \rightarrow Int$
$valorJugada \ = \ \ldots$
$mejorJugada :: Mano \rightarrow Jugada$
— calcula la jugada con puntuación máxima
$mejorJugada \ = \ \ldots$

$\boxed{\mathbf{4}}$ Generaliza la función *posibles* si una mano puede tener un número arbitrario de cartas:
$Main >\ posibles\ [1, 2, 5]$
$[[1, 2, 5], [1, 2, 4], [1, 7, 5], [1, 7, 4], [8, 2, 5], [8, 2, 4], [8, 7, 5], [8, 7, 4]] \ :: \ [Jugada]$

$\boxed{\mathbf{5}}$ Escribe una función para comprobar si con una mano se pueden obtener todos los puntos del 1 al 8:
$Main >\ map\ valorJugada\ (posibles\ [1, 2, 5])$
$[8, 7, 4, 3, 6, 5, 2, 1] \ :: \ [Int]$
$Main >\ todos\ [1, 2, 5]$
$True \ :: \ Bool$

$\boxed{\mathbf{6}}$ Se considera la sucesión $\{f_n\}_{n\geq 0}$ definida en forma inductiva en la forma siguiente:

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

$\boxed{\mathbf{A}}$ 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

$\boxed{\mathbf{B}}$ Use la red anterior para comprobar que la salida es la lista cíclica $[1, -1, 1, -1, 1, -1, ...]$: