

Primer Parcial

Este examen debe ser resuelto en forma individual. No olvide poner su nombre y número de documento en el encabezado de la resolución.
La nota mínima de aprobación es de cinco puntos.

Ejercicio 1.[2.5pt] Dado un árbol binario de enteros (de la forma que fueron definidos en clases), definir una función en Haskell `f::BinTree Int -> Int` que retorne la suma de la información en los nodos.

Escriba un árbol que contenga los 4 primeros dígitos de su DNI.

Ejercicio 2.[2.5pt] Dado el siguiente tipo de datos definidos en Haskell.

```
data SIntExp = Cte Int | Sum SIntExp SIntExp | Mul SIntExp SIntExp
```

- Dar 4 expresiones que sean del tipo dado,
- Con el tipo dado dar una expresión correspondiente a $d_0 + (d_1 * d_2)$. En donde d_0, d_1, d_2 son los tres primeros dígitos de su DNI.

Ejercicio 3.[2.5pt] Defina la función `compact:: [Int] -> [Int]` que elimina repeticiones consecutivas de un elemento. Por ejemplo: `compact [2,2,3,1,3] = [2,3,1,3]`. Aplique la función a los dígitos de su DNI, y muestre las reducciones que haría Haskell.

Ejercicio 4.[2.5pt] Dadas las siguientes definiciones:

```
square :: Int -> Int  
square x = x*x
```

```
and :: Bool -> Bool -> Bool  
and true y = y  
and false x = false
```

```
inf :: Int -> Int  
inf = inf+1
```

Elija **solo dos** de las siguientes expresiones y aplique las evaluaciones normal y aplicativa. Cuáles de ellas tienen formal normal y cuáles no? Justificar su respuesta.

```
(square inf) + (square inf)
and (inf == inf) ((square 2) == 4)
and ((square 2) == 5) (inf == inf)
```