Programación Declarativa

Sesión de laboratorio 3

Curso 2021/22

- Realiza los siguientes ejercicios en un mismo fichero .hs.
- Escribe tu nombre al comienzo del fichero como líneas comentadas. Incluye comentarios significativos y no olvides `declarar los tipos` de las expresiones que definas.
- Sube el fichero al Campus Virtual antes de que acabe la clase.

1. Define mediante `foldr` o `foldl`, en lugar de mediante recursión explícita, las siguientes funciones predefinidas en Prelude. Expresa mediante λ-expresiones el primer argumento de la la función `fold` que utilices.
   - `last`,
   - `reverse`,
   - `all`,
   - `minimum`,
   - `map`,
   - `filter`,
   - `takeWhile`.
2. Define una expresión Haskell para representar la lista $[1, -1, 2, -2, 3, -3, \ldots 100, -100]$ usando `foldl`.
3. Expresa la lista infinita $[(0,0),(0,1),(1,0),(0,2),(1,1),(2,0),(0,3),(1,2),\ldots]$, que representa una enumeración de todas los pares de números naturales, mediante una lista intensional.
4. Programa las siguientes funciones, usando orden superior o listas intensionales.
   (a) `sufijos xs` = lista de todos los sufijos de `xs`. Por ejemplo:
   `sufijos [1,2,3] = [[1,2,3],[2,3],[3],[]]`.
   (b) `sublistas xs` = lista de todas las sublistas de `xs`. Por ejemplo:
   `sublistas [1,2,3] = [[],[1],[2],[3],[1,2],[2,3],[1,2,3]]`.
   (c) `permuta xs` = lista de todas las permutaciones de `xs`. Por ejemplo:
   `permuta [1,2,3]=[[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]`.
   (d) `sumandos n` devuelve la lista de todas las descomposiciones en sumandos positivos de `n`. Por ejemplo:
   `sumandos 3 = [[1,1,1],[1,2],[3]]`.