

Práctico 3: Programación Funcional

El objetivo de estos ejercicios es ayudar a afianzar los conocimientos sobre programación funcional. En cada ejercicio se debe dar el perfil de la función.

NOTA Los ejercicios con * son para resolver en tu casa.

1. Define una función que, dadas dos listas **ys** y **xs** de naturales ordenadas, retorne el *merge* de estas listas, es decir, la lista ordenada compuesta por los elementos de **ys** y **xs**.
2. Define una función que, dada una lista de naturales, la ordene.
3. Define una función que, recursivamente y sólo utilizando adición y multiplicación, calcule, dado un natural n , el número 2^n .
4. Define una función que, dado un número natural n , retorne su representación binaria como secuencia de bits.
- 5 *. Define una función que, dado un número natural n en su representación binaria, decida si n es par o no.
6. Define la función que retorne la distancia de Hamming: dadas dos listas es el número de posiciones en que los correspondientes elementos son distintos. Por ejemplo: $\text{distanciaH } \text{"roma"} \text{"camino"} = 3$
 $\text{distanciaH } \text{"romano"} \text{"rama"} = 1$
7. Define la función que, dado un número natural, decida si el mismo es un cuadrado perfecto o no.
8. Define la función *repetidos* de forma tal que dado un elemento **z** y un entero **n**; **z** aparece **n** veces.
9. Define la función *nelem* tal que *nelem xs n* es elemento en-ésimo de **xs**, empezando a numerar desde el 0. Por ejemplo:
 $\text{nelem } [1, 3, 2, 4, 9, 7] 3 = 4$
- 10 *. Define la función *posicionesC* tal que *posicionesC xs c* es la lista de la posiciones del carácter **c** en la cadena **xs**. Por ejemplo:
 $\text{posicionesC } \text{"Catamarca"} \text{'a'} = [1, 3, 5, 8]$
11. Define la función *compact*, dada una lista retorna la lista sin los elementos repetidos consecutivos. Por ejemplo: $\text{compact } [1, 3, 3, 5, 8, 3] = [1, 3, 5, 8, 3]$.