

# Programación Funcional

## Relación de ejercicios nº 1

1. Definir una función que calcule el número de negativos de una lista de números.
2. Definir una función que obtenga el mayor dígito de un número natural. Por ejemplo, para 17427 el resultado es 7 y para 3262 el resultado es 6. Indicación: Definir primero una función que obtenga la lista de dígitos de un número.
3. Definir una función que calcule las posiciones de un elemento dado en una lista dada.  
Ejemplo: posiciones 8 [8,9,5,8,2]  $\Rightarrow$  [0,3]
4. Definir una función que calcule la diferencia de dos listas (atención repeticiones).  
Ejemplo: dif [1,2,3,7,4,2,8,4] [3,5,2,4,2]  $\Rightarrow$  [1,7,8,4]
5. Usando la función dif anterior, definir una función perm que, dadas dos listas, decida si una lista es permutación de la otra.  
Ejemplos: perm "abc" "acb"  $\Rightarrow$  True  
perm [1,2,2,4] [2,4,1]  $\Rightarrow$  False
6. Usar elem y listas intensionales para definir una función  
intersec :: Eq a => [a] -> [a] -> [a]  
tal que intersec xs ys que obtenga la lista de todos los elementos de xs que también están en ys.
7. Generalizar la siguiente función conocida como *ordenación rápida*:  
quicksort [ ] = [ ]  
quicksort (x:xs)  
= quicksort [y | y<-xs, y<x] ++ [x] ++  
quicksort [y | y<-xs, y>=x]  
a una función genQsort que tenga como argumento (además de la lista) el predicado binario con respecto al que se ordenará la lista. De modo que  
quicksort = genQsort (<)  
Por ejemplo:  
genQsort (>) [3,2,3,4,5,2]  $\Rightarrow$  [5,4,3,3,2,2]  
genQsort (\(x,y) (u,v) -> x<u)  
[(3,'a'), (2,'b'), (3,'f'), (4,'a'), (5,'s'), (2,'h')]  
 $\Rightarrow$  [(2,'b'), (2,'h'), (3,'a'), (3,'f'), (4,'a'), (5,'s')]  
genQsort (==) [3,2,3,4,5,2]  $\Rightarrow$  [3,3,2,2,4,5]
8. Definir una función que decida si todos los elementos de una lista son iguales entre sí
9. Definir una función que decida si todos los elementos de una lista son distintos entre sí

10. Usar `map` para definir una función que dada una lista de pares obtenga la lista de los productos  $x*y$  de cada par  $(x,y)$ . ¿Qué tipo tiene? ¿Se ha usado también `uncurry`?
11. Definir una función que dadas dos listas decida si en ambas aparecen exactamente los mismos elementos (el número de apariciones es irrelevante).
12. Definir dos versiones (una usando `filter` y otra una lista ZF) de la función `posimpar` que, dada una lista, obtiene la lista de todos sus elementos cuya posición es impar (recuerda que la primera posición es 0).  
Ejemplo: `posimpar [0,1,2,3,4,5,6,7,8,9] ⇒ [1,3,5,7,9]`
13. Usando `length` y una lista ZF, define la función `numveces` que calcule el número de apariciones de un elemento dado en una lista dada.  
Ejemplo: `numveces 8 [8,9,5,8,2] ⇒ 2.`
14. Definir una función que dadas dos listas  $[x_1, x_2, \dots, x_n]$  e  $[y_1, y_2, \dots, y_m]$  obtenga la lista  
 a)  $[(x_1+x_2+ \dots +x_n)*y_1, \dots, (x_1+x_2+ \dots +x_n)*y_m]$   
Ejemplo: `f1 [1,2,3] [4,5] ⇒ [24,30]`  
 b)  $[x_1*(y_1+y_2+ \dots +y_m), \dots, x_n*(y_1+y_2+ \dots +y_m)]$   
Ejemplo: `f2 [1,2,3] [4,5] ⇒ [9,18,27]`
15. Definir una función que obtiene todos los triples de Pítagoras con valores entre 1 y  $n$ . Un triple  $(x, y, z)$  de enteros es de Pítagoras si  $x^2+y^2$  coincide con  $z^2$ .  
Ejemplo: `pitagoras 12 ⇒ [(3,4,5), (4,3,5), (6,8,10), (8,6,10)]`
16. Definir una función que dadas dos listas obtenga la lista que resulta de alternar los elementos de ambas. Por ejemplo, si los datos son  $[8,7,6]$  y  $[1,2,3,4]$ , el resultado debe ser  $[8,1,7,2,6,3,4]$ .
17. Dar el tipo más general de las siguientes funciones:  
`f1 x y = head y && fst x`  
`f2 x y = fst x == length y`  
`f3 x y = x (snd y)`  
`f4 = (+1) . (f0 5) where f0 x y = length (take x y)`  
`f5 = map f4`  
`f6 = map (f3 (5,3))`  
 sabiendo que el tipo de las predefinidas que usan es:  
`head :: [a] -> a`  
`fst :: (a,b) -> a`  
`snd :: (a,b) -> b`  
`length :: [a] -> Int`  
`map :: (a -> b) -> [a] -> [b]`

18. ¿Qué tipo tiene y que hace la siguiente función f?

```
f [] = [[]]
f (x:xs) = cs ++ (map (x:) cs) where cs = f xs
```

19. ¿Qué tipo tiene y que hace la siguiente función f?

```
f [] = ([], [])
f (x:xs) = (x:pile2, pile1)
           where (pile1, pile2) = f xs
```

¿Cuántas veces recorre la lista dato?

20. Definir una función que dada una lista y un natural n, obtiene la lista de sus n últimos elementos. Indicación: Usar drop .

21. Definir una función que dada una lista y un natural n, rota los n últimos elementos de la lista situándolos como los n primeros. Indicación: Usar take y drop .

¿Sin recorrer la lista dos veces? Indicación: Usar splitAt

22. Definir una función que dada una lista y dos naturales i, j obtiene la sublista de los elementos que hay entre ambas posiciones (ambas inclusive). Indicación: Usar take y drop .

23. ¿Qué tipo tienen y que calculan las dos siguientes funciones?

a) f1 = foldr (\a b -> (a:) . b) id

b) f2 = foldr ((.) . (:)) id

24. Definir una función que divida una lista, con respecto a un predicado p, en dos listas una formada por los elementos de la lista original que cumplan p y la otra por los que no lo cumplen. Definir dos versiones, una que recorra la lista dato dos veces (usando listans intensionales) y otra que la recorra una sola vez (gracias a una definición local).

Ejemplo: particion (<0) [1, -3, 5, 8, -7, 0] ⇒ ([-3, -7], [1, 5, 8, 0])

25. Define un predicado que decida si una lista está ordenada.

Ejemplos: ordenada ['a', 'y', 'u'] ⇒ False

ordenada [1, 3, 7, 12] ⇒ True

26. Define las funciones decimalAbinario y binarioAdecimal para transformar un número decimal a binario y viceversa (utiliza alguna función de “la familia fold”, cuando sea posible)

Ejemplos: decimalAbinario 13 ⇒ 1101

binarioAdecimal 1101 ⇒ 13

27. Define una función `permutar` que devuelva todas las permutaciones de una lista dada.  
Ejemplo: `permutar [2, 8, 3] ⇒`  
`[[2, 8, 3], [8, 2, 3], [8, 3, 2], [2, 3, 8], [3, 2, 8], [3, 8, 2]]`
28. Una lista es una sublista de otra si los elementos de la primera aparecen en la segunda en el mismo orden. Una lista es una subsecuencia de otra si aparece en ella como secuencia de elementos contiguos. Define las funciones `sublista` y `subsecuencia`.  
Ejemplos: `sublista "palma" "tapa la mesa" ⇒ True`  
`subsecuencia "palma" "tapa la mesa" ⇒ False`
29. Usando `zip` y `foldr1`, definir una función que, dada una lista de números, calcule el par `(dif, pos)` tal que `dif` es la mayor diferencia en valor absoluto entre dos elementos consecutivos, y `pos` es la posición (en la lista `dato`) del primero de dichos elementos. Ejemplo: `mayordif [1, 17, 43, 20, -23, 14] ⇒ (43, 3)`
30. Usando `foldr1`, definir una función que calcule la palabra más larga de una lista de palabras.