

Ejercicios de Examen de Matemática I

Ejercicio 1

Define en Haskell las siguientes funciones:

- a) Una función polimórfica **eliminarDuplicados** que devuelva la lista que se obtiene al eliminar los elementos duplicados de una lista, por ejemplo:

`eliminarDuplicados [1,2,1,2,3,2,4] = [1,2,3,4]`

- b) Una función **paresImpares** que tome como argumento una lista de enteros y devuelva una tupla con dos listas. En la primera aparecerán los elementos pares del argumento y en la segunda los impares, por ejemplo:

`paresImpares [10,12,13,14,4,7] = ([10,12,14,4], [13,7])`

Ejercicio 2

La conjetura de Goldbach es la siguiente: Todo número par mayor que 2 puede escribirse como suma de dos números primos. (Se puede emplear dos veces el mismo número primo).

- a) Para comprobar la conjetura, escribe una función **goldbach** que tome un número par mayor que 2 y devuelva una lista con los pares de números primos tales que su suma igualan al número leído.

Por ejemplo: `goldbach 58 = [(5,53), (11,47), (17,41), (29,29)]`

- b) Escribe una función **cumplenGoldbach** que tome una lista de números pares mayor que 2 y devuelva True si todos cumplen la conjetura, y en otro caso devuelve False.

Ejercicio 3

- a) Define una función que devuelva las partes (todas las sublistas) de una lista

`partes [1,2,3] = [[], [1], [2], [3], [1,2], [2,3], [1,3], [1,2,3]]`

Nota: no es necesario que los resultados aparezcan en el mismo orden que en el ejemplo.

- b) Define usando listas por comprensión la función concatenar.

Ejercicio 4

Ejercicios:

- a) Se pide una función que dado un elemento y una lista añada dicho elemento al final de la misma. `añadir ([1,3,7,9],4) = [1,3,7,9,4]`

- b) Diseñar en Haskell una función que determine si un carácter se encuentra dentro de una palabra. `pertenece ("programación", 'g') = True`

- c) Diseñar una función en Haskell que reciba una palabra y sustituya un cierto carácter por otro, devolviendo la palabra modificada. `palabra ("certeza", 't', 'v') = "cerveza"`

- d) Diseña una función en Haskell que muestre los números primos existentes en un intervalo. `listaPrimos (19,93) = [19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89]`

- e) Diseña una función en Haskell que recibe 2 listas y va cogiendo un elemento de la primera y dos de la segunda, creando una lista final de ternas. En caso de que una de las dos listas se acaben, mostrará la lista de ternas construidas hasta entonces.

`lista ([4,5,8,90],[0,5,6,-9,8,-1,9,52,22]) = [(4,0,5),(5,6,-9),(8,8,-1),(90,9,52)]`

- f) Implementa una función en Haskell que elimine de una lista aquellos números que se encuentren en una posición múltipla de x.

`cribar([0,5,8,9,-9,6,0,85,-12,15],2) = [0,8,-9,0,-12]`