Departamento de Computación – FCEyN – UBA

# Taller de Álgebra I - Simulacro de parcial

### Aclaraciones

- El parcial se aprueba con tres ejercicios bien resueltos.
- Programe todas las funciones en lenguaje Haskell. El código debe ser autocontenido. Si utiliza funciones que no existen en Haskell, debe programarlas. Incluya la signatura de todas las funciones que escriba.
- No está permitido alterar los tipos de datos presentados en el enunciado, ni utilizar técnicas no vistas en clase para resolver los ejercicios.

### Ejercicio 1

Implementar la función `menorLex :: (Float, Float, Float) -> (Float, Float, Float) -> Bool` que dados dos vectores $x, y \in \mathbb{R}^3$ decida si $x$ es menor a $y$ en el sentido lexicográfico, es decir, $x < y$ si la primera coordenada de $x$ es menor que la primera coordenada de $y$ o son iguales y se satisface esto mismo para el resto del vector.

*Por ejemplo:*

`menorLex (3,-1,2) (5,10,0)` ⇝ `True` (pues $3 < 5$)
`menorLex (4,-1,7) (4,21,5)` ⇝ `True` (pues coinciden en la primera coordenada, y $-1 < 21$)
`menorLex (2,1,31) (2,1,-5)` ⇝ `False` (pues coinciden en las dos primeras coordenadas, pero $31 \not< -5$)

### Ejercicio 2

Implementar una función `sumaFibonacci :: Integer -> Integer` que para cada $n \geq 1$ calcule $\sum_{j=0}^{n} f_n$, donde $f_n$ es $n$–ésimo término de la sucesión de Fibonacci.

*Por ejemplo:*

`sumaFibonacci 2` ⇝ `1+1+2` ⇝ `4`
`sumaFibonacci 4` ⇝ `1+1+2+3+5` ⇝ `12`

### Ejercicio 3

Implementar la funcion `esDefectivo :: Integer -> Bool` que dado un $n \in \mathbb{N}_{>0}$ determine si es *defectivo*, lo cual vale si y solo si la suma de los divisores propios positivos[1] de $n$ es menor que $n$.

*Por ejemplo:*

`esDefectivo 16` ⇝ `True`
`esDefectivo 12` ⇝ `False`

ya que la suma de los divisores propios de 16 es $1 + 2 + 4 + 8 = 15$, que es menor que 16; y la suma de los divisores propios de 12 es $1 + 2 + 3 + 4 + 6 = 16$, que no es menor que 12.

### Ejercicio 4

Programe la función `maximaDistancia :: [Integer] -> Integer`, que determina cuál es la máxima distancia entre dos elementos consecutivos en una lista de números enteros.

*Por ejemplo:*

`maximaDistancia [1,6,2,7,8]` ⇝ `5` (la máxima distancia es entre 1 y 6)
`maximaDistancia [1,6,2,7,1]` ⇝ `6` (la máxima distancia es entre 7 y 1)
`maximaDistancia [1,5,-10,3]` ⇝ `15` (la máxima distancia es entre 5 y $-10$)

**Aclaración:** Puede asumir que la lista tiene al menos dos elementos.

### Ejercicio 5

Programe la función `comprimir :: [Integer] -> [(Integer, Integer)]` que dada una lista de números enteros devuelva una lista que contenga una tupla (número, cantidad de apariciones) por cada ráfaga de números iguales adyacentes.

*Por ejemplo:*

`comprimir [1,1,7,7,4,4,1,4,4,4,3,3,3]` ⇝ `[(1,2),(7,2),(4,2),(1,1),(4,3),(3,3)]`

Sugerencia: Empiece reemplazando cada número n por una tupla (n, 1).

[1] Recordar: decimos que $d$ es un *divisor propio* de $n$ si $d \mid n$ y $d \neq n$.