

# Taller de Álgebra I

Clase 5 - Recursión con funciones auxiliares

Segundo cuatrimestre 2022

## ¿Una fácil?... o no tanto

- ▶ `sumaDivisores :: Int -> Int` que calcule la suma de los divisores de un número natural.

¿Qué sucede si construimos una función **más general** que nos facilita el trabajo?

```
sumaDivisoresHasta :: Int -> Int -> Int
```

que devuelve la suma de los divisores de un natural hasta cierto punto.

## Ejercicios

- 1 Implementar una función `sumaDivisoresHasta :: Int -> Int -> Int`.
- 2 Implementar la función `sumaDivisores` utilizando la función anterior.

## Ejercicios

Un entero  $p > 1$  es **primo** sii no existe un natural  $k$  tal que  $1 < k < p$  y  $k$  divida a  $p$ .

- 3 Implementar `menorDivisor :: Int -> Int` que calcule el menor divisor (mayor que 1) de un natural  $n$ .
- 4 Implementar la función `esPrimo :: Int -> Bool`.
- 5 Implementar la función `nEsimoPrimo :: Int -> Int` que devuelve el  $n$ -ésimo primo ( $n \geq 1$ , el primer primo es el 2, el segundo es el 3, el tercero es el 5, etc.)

## Ejercicios

- 6 Implementar `menorFactDesde :: Int -> Int` que dado  $m \geq 1$  encuentra el mínimo  $n \geq m$  tal que  $n = k!$  para algún  $k$ .
- 7 Implementar `mayorFactHasta :: Int -> Int` que dado  $m \geq 1$  encuentra el máximo  $n \leq m$  tal que  $n = k!$  para algún  $k$ .
- 8 Implementar `esFact :: Int -> Bool` que dado  $n \geq 0$  decide si existe un número entero  $k \geq 0$  tal que  $n = k!$

## Ejercicios

- 9 Implementar `esFibonacci :: Int -> Bool` que dado un número entero  $n \geq 0$  decide si  $n$  es un número de Fibonacci.
- 10 Implementar `esSumaInicialDePrimos :: Int -> Bool` que dado un número entero  $n \geq 0$  decide si  $n$  es igual a la suma de los  $m$  primeros números primos, para algún  $m$ .

## Ejercicios

- 11 Implementar `tomaValorMax :: Int -> Int -> Int` que dado un número entero  $n_1 \geq 1$  y un  $n_2 \geq n_1$  devuelve algún  $m$  entre  $n_1$  y  $n_2$  tal que  $\text{sumaDivisores}(m) = \max\{\text{sumaDivisores}(i) \mid n_1 \leq i \leq n_2\}$
- 12 Implementar `tomaValorMin :: Int -> Int -> Int` que dado un número entero  $n_1 \geq 1$  y un  $n_2 \geq n_1$  devuelve algún  $m$  entre  $n_1$  y  $n_2$  tal que  $\text{sumaDivisores}(m) = \min\{\text{sumaDivisores}(i) \mid n_1 \leq i \leq n_2\}$