Taller de Álgebra I

Clase 5 - Recursión con funciones auxiliares

Primer cuatrimestre 2020

Generalización de funciones

¿Una fácil?.. o no tanto

- `sumaDivisores :: Int -> Int` que calcule la suma de los divisores un entero positivo.

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

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

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

Ejercicios

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

La función `esPrimo` y sus primos

Ejercicios

Un entero $p > 1$ es **primo** sii no existe un natural k tal que $1 < k < p$ y k divide 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-esimo 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 Fobonacci.
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 $\texttt{sumaDivisores}(m) = \max\{\texttt{sumaDivisores}(i) \mid n_1 \leq i \leq n_2\}$
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 $\texttt{sumaDivisores}(m) = \min\{\texttt{sumaDivisores}(i) \mid n_1 \leq i \leq n_2\}$

Más ejercicios

Ejercicios

13. Implementar `esSumaDeDosPrimos :: Int -> Bool` que, dado un número natural n, determine si puede escribirse como suma de dos números primos.
14. **Conjetura de Christian Goldbach, 1742:** todo número par mayor que 2 puede escribirse como suma de dos números primos. Escribir una función que pruebe la conjetura hasta un cierto punto. `goldbach :: Int -> Bool` (hasta al menos 4.10^{18} debería ser cierto)
15. Los números naturales a y b forman un *par de primos gemelos* si $b = a + 2$ y tanto a como b son primos. Implementar `primosGem :: Int -> Int` que dado n, devuelve la cantidad de pares de primos gemelos (a, b) que verifican $b \leq n$. Por ejemplo: `primosGem` $5 = 1$ (porque 3 y 5 es un par de primos gemelos) `primosGem` $14 = 3$ (porque 3 y 5, 5 y 7, y 11 y 13 son tres pares de primos gemelos)
16. **Conjetura de los primos gemelos:** Existen infinitos pares de primos gemelos. Implementar la función `proxPrimosGem :: Int -> (Int,Int)` que dado n devuelve el primer par de gemelos (a, b) tal que $a > n$.

Y uno más

Ejercicios

17 **Conjetura de Lothar Collatz, 1937**: sea la siguiente definición:

$$a_{n+1} = \begin{cases} \frac{a_n}{2} & \text{si } a_n \text{ es par} \\ 3a_n + 1 & \text{si } a_n \text{ es impar} \end{cases}$$

empezando a_1 con cualquier entero positivo siempre se llega a 1. Por ejemplo, si $a_1 = 13$, obtenemos la siguiente secuencia: $13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$ (9 reducciones, o sea 9 flechas).

a. Implementar `largoSecuencia :: Int -> Int` que dado un $n > 0$ devuelve la cantidad de reducciones desde $a_1 = n$ hasta llegar a 1. Por ejemplo, `largoSecuencia 13` es 9.
b. Resolver usando Haskell: ¿qué número menor a 10.000 para a_1 produce la secuencia de números más larga hasta llegar a 1? Sugerencia: usar la idea de la función del ejercicio 11.