Taller de Álgebra 1
Ejercicios

April 19, 2021

1. La sucesión de los números de Fibonacci es la sucesión de enteros $(f_n)_{n\geq 0}$ tal que

$$f_0 = 0, \quad f_1 = 1, \quad f_{n+2} = f_{n+1} + f_n, \quad \forall n \geq 0.$$

Escriba una función `fibonacci :: Integer -> Integer` tal que `fibonacci n` sea el n-ñesimo número de Fibonacci.

```
*Main> fibonacci 30
832040
*Main> fibonacci 2
1
*Main> fi
fibonacci  filter
*Main> fibonacci 0
0
```

2. La sucesión de los números de Lucas es la sucesión de enteros $(l_n)_{n\geq 0}$ tal que

$$l_0 = 2, \quad l_1 = 1, \quad l_{n+2} = l_{n+1} + l_n, \quad \forall n \geq 0.$$

Escriba una función `lucas :: Integer -> Integer` tal que `lucas n` sea el n-ñesimo número de Lucas.

```
*Main> lucas 25
167761
*Main> lucas 10
123
*Main> lucas 2
3
```

3. Si a, b, c y d son números enteros, definimos una sucesión de enteros $(x_n)_{n\geq 0}$ de manera que

$$l_0 = a, \quad l_1 = b, \quad l_{n+2} = cl_{n+1} + dl_n, \quad \forall n \geq 0.$$

Defina una función

```
sucesión :: Integer -> Integer -> Integer -> Integer -> Integer -> Integer
```

tal que `sucesión a b c d e n` sea el n-ésimo elemento de esa sucesión.

Use esta función `sucesión` para dar nuevas deficiones de las sucesiones `fibonacci` y `lucas` de los ejercicios anteriores

```
*Main> sucesión 3 4 5 6 7
76
*Main> sucesión 7 6 5 4 3
19
*Main> sucesión 2 2 2 2 2
4
```

4. Usando la función `fibonacci` que escribió en el ejercicio ?? ¿puede calcular el 100-ésimo o 1000-ésimo número de Fibonacci? Estos dos números son

$$f_{100} = 354224848179261915075$$

y

$$\begin{aligned} f_{1000} = {} & 43466557686937456435688527675040625802564660517371780402481729089536555417949051890403879840079255169295922593080322634775209689623239873322471161642996440906533187938298969649928516003704476137795166849228875, \end{aligned}$$

respectivamente — el segundo tiene 209 dígitos.

¿Puede determinar cuántas veces se hace la suma de dos números para calcular `fibonacci 10` usando la definición que dio? ¿Cómo cree que yo pude calcular f_{1000}?

5. Escriba un programa que le permita encontrar el primer número de Fibonacci cuyas últimas tres cifras son 946.

6. El triangulo de Pascal es la siguiente tabla triangular cuyas primeras filas son las siguientes:

		k										
		0	1	2	3	4	5	6	7	8	9	10
n	0	1										
	1	1	1									
	2	1	2	1								
	3	1	3	3	1							
	4	1	4	6	4	1						
	5	1	5	10	10	5	1					
	6	1	6	15	20	15	6	1				
	7	1	7	21	35	35	21	7	1			
	8	1	8	28	56	70	56	28	8	1		
	9	1	9	36	84	126	126	84	36	9	1	
	10	1	10	45	120	210	252	210	120	45	10	1

que se arma de la siguiente manera:

- la columna con $n = 0$ está completa con 1,
- la diagonal en la que $n = k$ está completa con 1,
- todas todas las otras entradas son la suma de la que está arriba y la que está justo arriba a la izquierda: por ejemplo el 126 que está en la fila 9 y la columna 4 es la suma del 70 que está inmediatamente arriba y el 56 que está a la izquierda de este.

Escriba una función `pascal :: Integer -> Integer -> Integer` tal que la expresión |pascal n k| sea el entero que está en la tabla en la fila n-ésima y la columa k-ésima.

```
*Main> pascal 20 10
184756
*Main> pascal 6 3
20
*Main> pascal 8 5
56
```

7. Escriba una función `dígitoMásFrecuente :: Integer -> Integer` tal que cada vez que `n` es un entero positivo `dígitoMásFrecuente n` sea el número de $\{0, \ldots, n\}$ que más veces aparece como dígito de n. Si hay «empates», elija el más chico.

```
*Main> dígitoMásfrecuente 1223334444
4
*Main> dígitoMásfrecuente 111222333
1
*Main> dígitoMásfrecuente 333222111
1
*Main> dígitoMásfrecuente 1234567890
0
*Main> dígitoMásfrecuente 1
1
*Main> dígitoMásfrecuente 7
7
```

8. Escriba una función `dígitosDecrecientes :: Integer -> Bool` tal que para cada entero positivo `n` la expresión `dígitosDecrecientes n` sea `True` o `False` dependiendo de si los dígitos de `n` decrecen de derecha a izquierda.

```
*Main> dígitoDecrecientes 12345
True
*Main> dígitoDecrecientes 111222233334444455555
True
*Main> dígitoDecrecientes 111222233334344455555
False
*Main> dígitoDecrecientes 9876542321
False
```

9. Escriba una función `dígitosOrdenados :: Integer -> Bool` tal que para cada entero positivo `n` la expresión `dígitosDecrecientes n` sea `True` o `False` dependiendo de si los dígitos de `n` o bien crecen o bien decrecen de derecha a izquierda.

```
*Main> dígitosOrdenados 12345
True
*Main> dígitosOrdenados 54321
True
*Main> dígitosOrdenados 55444333222111
True
*Main> dígitosOrdenados 122334444555777999
True
*Main> dígitosOrdenados 123454321
False
*Main> dígitosOrdenados 1221212
False
```