

Practico N° 9 – Principio de Inducción estructural

Notas:

- Previo a la realización de este práctico es necesario leer el siguiente material sobre:
- [Principio de Inducción Estructural](#)

Inducción sobre N y NxN:

1. Dar una definición recursiva para la función factorial $f(n)=n!$. Luego implementar la función factorial en Haskell, utilizar el conjunto N, proporcionado en el módulo Naturales.
2. Dar una definición recursiva para a^n , donde a es un número real no nulo, y n es un entero no negativo.
3. Considere la siguiente función definida por recursión en N: $f:N \rightarrow N$
 $f(0)=0$
 $f(S(n))=f(n) + 2.S(n)$
Pruebe que $\forall n \in N: f(n) = n.S(n)$
4. Identifique las siguientes funciones, o sea, señale qué función conocida es computada por las siguientes ecuaciones:

$g:N \times N \rightarrow N$	$f:N \times N \rightarrow N$
$g(n,0) = n$	$f(0, n)=0$
$S(g(n,S(m))) = g(n,m)$	$f(S(n),0)=0$
	$f(S(n),S(m))=S(f(n,m))$
5. Teniendo en cuenta las funciones del ejercicio anterior, pruebe mediante inducción que
a) $(\forall n \in N: f(n,0)=0)$
b) $(\forall (n, m) \in N \times N : g(n, m) = g(S(n),S(m)))$
6. Probar las siguientes propiedades:
a. $\text{suma_S}:(\forall n,m \in N)((\text{suma } n (S m)) = (\text{suma } (S n) m))$
b. $\text{suma_conm}:(\forall n,m \in N)((\text{suma } n m) = (\text{suma } m n))$

Inducción sobre Listas:

7. Probar las siguientes propiedades:
 - a. $P1:(\forall l \in \text{list } A)((\text{concat } l \text{ nil}) = l)$
 - b. $P2:(\forall l \in \text{list } A)(\forall a \in A)(\text{not}((\text{cons } a l) = \text{nil}))$
 - c. $P3:(\forall l, q \in \text{list } A)(\forall n \in A)((\text{cons } n (\text{concat } l q))=(\text{concat } (\text{cons } n l) q))$
 - d. $P4:(\forall l, q \in \text{list } A)((\text{largo } (\text{concat } l q)) = (\text{largo } l)+(\text{largo } q))$
 - e. $P5:(\forall l \in \text{list } A)((\text{largo } (\text{invertir } l))=(\text{largo } l))$
 - f. $P6:(\forall l, q \in \text{list } A)((\text{concat } l (\text{concat } q p))=(\text{concat } (\text{concat } l q) p))$
 - g. $P7:(\forall l, q \in \text{list } A)((\text{invertir } (\text{concat } l q)) = (\text{concat } (\text{invertir } q) (\text{invertir } l)))$
 - h. $P8:(\forall l, q \in \text{list } A)(\forall f:N \rightarrow N)((\text{mapear } f (\text{concat } l q))=(\text{concat } (\text{mapear } f l) (\text{mapear } f q)))$
 - i. $P9:(\forall l, q \in \text{list } A)(\forall P:N \rightarrow \text{bool})((\text{filtrar } P (\text{concat } l q))=(\text{concat } (\text{filtrar } P l) (\text{filtrar } P q)))$
 - j. $P10:(\forall l \in \text{list } A)((\text{invertir } (\text{invertir } l)) = l)$

8. Definir una función **esPerm** que dadas dos listas indique si la primera es una permutación de la segunda. Ejemplo:
 $\text{esPerm } [3, 4, 6, 3] [4, 3, 6, 3] = \text{True}$
Probar que para todo par de listas xs, ys , si $(\text{esPerm } xs \ ys)$, entonces $(\text{largo } xs) = (\text{largo } ys)$.
Nota: Se considera verdadera la siguiente proposición
 $(\forall x \in A)(\forall y \in \text{List } A)((\text{miembro } x \ ys) \rightarrow ((\text{largo } (\text{borrar } x \ ys)) + 1) = (\text{largo } ys))$
9. Definir en Haskell una función **intercalar** que dadas dos listas de números enteros, devuelva una nueva lista de enteros ordenada en forma creciente, formada por los elementos de las dos listas que toma como argumentos.
Nota: suponer las listas ordenadas en forma creciente.
Ejemplo:
 $\text{intercalar } [-7, -3, 0, 2, 4] [-10, -4, 1, 2, 8] = [-10, -7, -4, -3, 0, 1, 2, 2, 4, 8]$
10. Probar por inducción que: $\forall L_1, L_2 \in \text{List } A$, se cumple:
 $\text{largo}(\text{intercalar } L_1 \ L_2) = \text{largo } L_1 + \text{largo } L_2$

Ejercicios complementarios de Inducción:

11. Probar las siguientes propiedades:

- a) $\text{suma_Suc} : \forall n, m \in \mathbb{N}, \text{suma } n \ (\text{S } m) = \text{suma } (\text{S } n) \ m.$
- b) $\text{suma_conm} : \forall n, m \in \mathbb{N}, \text{suma } n \ m = \text{suma } m \ n.$
- c) $\text{suma_n_Suc} : \forall n, m \in \mathbb{N}, (\text{S } (\text{suma } n \ m)) = \text{suma } n \ (\text{S } m).$
- d) $n\text{Sucn} : \forall n \in \mathbb{N}, \text{distintos } n \ (\text{S } n).$
- e) $\text{prod_n_O} : \forall n \in \mathbb{N}, 0 = \text{producto } n \ 0.$
- f) $\text{mult_n_Sm} : \forall n, m \in \mathbb{N}, \text{producto } n \ (\text{suma } m \ n) = \text{producto } n \ (\text{S } m).$

12. Demostrar los siguientes propiedades, utilizando si es necesario inducción estructural sobre listas o naturales:

- a) P1: $\forall l \in \text{list } \mathbb{N}, (\text{concat } l \ \text{nilN}) = l$
- b) P2: $\forall (l \in \text{list } \mathbb{N}) (a \in \mathbb{N}), \text{not } (\text{consN } a \ l) = \text{nilN}.$
- c) P3: $\forall (l \ q \in \text{list } \mathbb{N}) (n \in \mathbb{N}), (\text{consN } n \ (\text{concat } l \ q)) = (\text{concat } (\text{consN } n \ l) \ q).$
- d) P4: $\forall (l \ q \in \text{list } \mathbb{N}), (\text{largo } (\text{concat } l \ q)) = (\text{largo } l) + (\text{largo } q).$
- e) P5: $\forall (l \in \text{list } \mathbb{N}), (\text{largo } (\text{invertir } l)) = (\text{largo } l).$
- f) P6: $\forall (l \ q \ p \in \text{list } \mathbb{N}), (\text{concat } l \ (\text{concat } q \ p)) = (\text{concat } (\text{concat } l \ q) \ p).$
- g) P7: $\forall l \ q \in \text{list } \mathbb{N}, (\text{invertir } (\text{concat } l \ q)) = (\text{concat } (\text{invertir } q) \ (\text{invertir } l)).$
- h) P8: $\forall (l \ q \in \text{list } \mathbb{N}) (f: \mathbb{N} \rightarrow \mathbb{N}), (\text{mapear } f \ (\text{concat } l \ q)) = (\text{concat } (\text{mapear } f \ l) \ (\text{mapear } f \ q)).$
- i) P9: $\forall (l \ q \in \text{list } \mathbb{N}) (P: \mathbb{N} \rightarrow \text{Bool}), (\text{filtrar } P \ (\text{concat } l \ q)) = (\text{concat } (\text{filtrar } P \ l) \ (\text{filtrar } P \ q)).$
- j) P10: $\forall l \in \text{list } \mathbb{N}, (\text{invertir } (\text{invertir } l)) = l.$