Programando Funciones matemáticas

SECUENCIAS
(o listas)

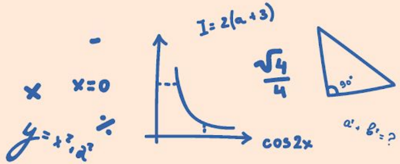
DEFINICIÓN INDUCTIVA DE LOS NATURALES

- $0 \in N$
- si $n \in N \Rightarrow succ(n) \in N$
- N sólo contiene elementos formados por las reglas anteriores donde succ es el la función sucesor.

$\Longrightarrow$

$1 = succ(0)$
$2 = succ(succ(0))$
$3 = succ(succ(succ(0)))$
$4 = succ(succ(succ(succ(0))))$

.
.
.



A partir de la definición inductiva de N podemos definir funciones recursivas:

- 0 ∈ N
- si n ∈ N ⇨ succ(n) ∈ N
- N sólo contiene elementos formados por las reglas anteriores donde succ es el la función sucesor

potencia: NxN -> N
potencia (m, n) = 1 si n == 0
o m * potencia (m, n - 1)

potencia(2, 3) = 2 * potencia(2, 2)
= 2 * 2 * potencia(2, 1)
= 2 * 2 * 2 * potencia(2, 0)
= 2 * 2 * 2 * 1

¿QUÉ ES UNA SECUENCIA?

Una secuencia de elementos de un conjunto A (Sec(A)) es una estructura inductiva que se define a partir de la secuencia vacía y una función cons (constructor)

Definición Inductiva de Sec(A):

- [] es una secuencia
- Si x es un elemento de A y s es una secuencia ⇨ cons (x,s) es una secuencia
- Los elementos de Sec(A) son solamente los construidos mediante las dos cláusulas anteriores

OBSERVACIONES

En los lenguajes de programación las secuencias se representan de la forma:

cons(0, []) como [0]
cons(1, cons(2, cons(3, []))) como [1,2,3]

Las secuencias están definidas por el orden, no se comportan como conjuntos:

[1,2,3] ≠ [3,2,1]
[1,1] ≠ [1]

cons(0, []) como [0]
cons(1, cons(2, cons(3,
[]))) como [1,2,3]

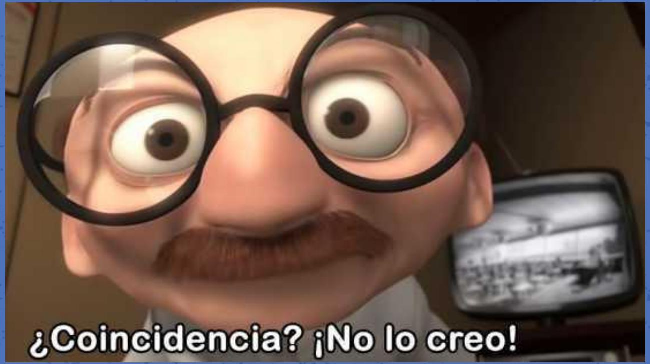
Las definiciones inductivas definen conjuntos partiendo de elementos base y mediante funciones que construyen un nuevo elemento a partir de elementos dados

0 y succ

- 0 ∈ a N
- si n ∈ a N ⇨ succ(n) ∈ a N
- N sólo contiene elementos formados por las reglas anteriores donde succ es el la función sucesor

[ ] y cons

- [] es una secuencia
- si x es un elemento de A y s es una secuencia ⇨ cons (x,s) es una secuencia
- los elementos de Sec(A) son solamente los construidos mediante las dos cláusulas anteriores



Secuencias en MateFun

En MateFun se utiliza el operador : y la secuencia vacía [] para formar secuencias.

<u>Otras funciones con secuencias:</u>

rango(inicial, final, paso) = Constructor de secuencias de 3 valores -> rango(2,10,2) = 2:4:6:8:10:[]
primero(Sec(N)) = Primer elemento de una secuencia -> primero(rango(2,10,2)) = 2
resto(Sec(N)) = Los elementos restantes de quitar el primero -> resto(rango(2,10,2)) = 4:6:8:10:[]

A partir de la definición inductiva de un conjunto podemos definir funciones recursivas para calcular sobre sus elementos. Ejemplo en N:

- 0 ∈ N
- si n ∈ N ⇨ succ(n) ∈ N
- N sólo contiene elementos formados por las reglas anteriores donde succ es el la función sucesor

factorial: N -> N
factorial (n) = 1 si n == 0
o n * factorial (n – 1)

factorial(5) = 5 * factorial(4)
= 5 * 4 * factorial(3)
= 5 * 4 * 3 * factorial(2)
= 5 * 4 * 3 * 2 * factorial(1)
= 5 * 4 * 3 * 2 * 1 * factorial(0)
= 5 * 4 * 3 * 2 * 1 * 1

A partir de la definición inductiva de un conjunto podemos definir funciones recursivas para calcular sobre sus elementos. Ejemplo en R*:

- [] ∈ R*
- si x ∈ R y xs ∈ R* ⇨ x:xs ∈ R*
- R* sólo contiene elementos formados por las reglas anteriores

donde : es la función cons

xs es primero(xs) : resto(xs)

```
sum :: R* -> R
sum (xs) = 0 si xs == []
           o primero(xs) + sum(resto(xs))

double :: R* -> R*
double (xs) = [] si xs == []
              o 2 * primero(xs) : double(resto(xs))
```

Principio de inducción estructural

Ejemplo en N

Sea P una propiedad (predicado)

Si se cumple

- $P(0)$ es verdadero
- Para todo $n \in N$ $(P(n) \Rightarrow P(n+1))$ es verdadero

entonces P es verdadero para todo $n \in N$

Principio de inducción estructural

Ejemplo en A*

Sea P una propiedad (predicado)

Si se cumple

- P([]) es verdadero
- Para todo $x \in A$ y $xs \in A^*$ $(P(xs) \Rightarrow P(x:xs))$ es verdadero

entonces P es verdadero para todo $xs \in A^*$

Principio de inducción estructural

Ejemplo en R*: Sea P *Para toda xs∈R*, sum(double(xs)) = 2 * sum(xs)*

Paso base (P([])

sum(double([])) == **def double**

sum([]) == **def sum**

0 == **aritmética**

2 * 0 == **def sum**

2 * sum([])

Principio de inducción estructural

Ejemplo en R*: Sea P *Para toda xs∈R*, sum(double(xs)) = 2 * sum(xs)*

Paso inductivo (P(xs) ⇨ P(x:xs))

sum(double(x:xs)) == **def double**

sum(2*primero(x:xs) : double(resto(x:xs))) == **def resto y primero**

sum(2*x : double(xs)) == **def sum**

2*x + sum(double(xs)) == **HI**

2*x + 2 * sum(xs) == **aritmética**

2 * (x + sum(xs)) == **def resto y primero**

2 * (primero(x:xs) + sum(resto(x:xs))) == **def sum**

2 * sum(x:xs)