UDE
UNIVERSIDAD DE LA EMPRESA

Matemática Discreta

- Licenciatura en Informática – Ingeniería en Informática
- 1º año

Conjuntos inductivos

Conjuntos definidos por inducción: Ya hemos visto cómo definir conjuntos por extensión y por comprensión. Ahora veremos un tercer mecanismo para definir conjuntos, que es la **inducción**.

Al definir un conjunto por inducción…

- **No** se usan llaves { } (como en extensión y comprensión)
- **No** se enumeran los elementos (como en extensión)
- **No** se especifica condición de pertenencia (como en comprensión)

Lo que se hace es definir **reglas** o **cláusulas** (se les llama de ambas maneras) que dicen cómo construir los elementos del conjunto.

Observación: Podemos pensar que una definición por inducción es como una “fábrica de elementos” que nos dice cómo hacer para ir creando los elementos del conjunto.

En una definición inductiva se utilizan 3 tipos de cláusulas:

- **Cláusulas BASE:** Dicen “dónde empezar”. Definen en forma explícita los primeros elementos que pertenecen al conjunto que se va a definir.

- **Cláusulas INDUCTIVAS:** Dicen “cómo seguir”. Definen nuevos elementos pertenecientes al conjunto a partir de otros elementos que ya formaban parte del conjunto.

- **Cláusula EXTREMA o de CLAUSURA:** Dice que “éstos son todos los elementos que pertenecen al conjunto”. Esta cláusula establece que no pueden existir en el conjunto elementos que no hayan sido construidos con las cláusulas anteriores.

UNIVERSIDAD DE LA EMPRESA

Ejemplos de Conjuntos definidos por inducción:

1) El conjunto **N** de los números naturales se define inductivamente mediante las siguientes cláusulas:

i.	**$0 \in N$**	(cláusula **base**)
ii.	Si **$n \in N$** entonces **$s(n) \in N$**	(cláusula **inductiva**)
iii.	Extrema	(cláusula **extrema**)

Observación: s(n) significa el sucesor de n. Es decir, el siguiente natural que viene luego de n (o sea que s(n) = n + 1). Por ejemplo, el 3 se puede representar como s(2) o como s(s(s(0))).

Definir por inducción el conjunto de los naturales impares (I).

Definir por inducción la relación de identidad entre naturales (Id_N).

UNIVERSIDAD DE LA EMPRESA

Ejemplos de Conjuntos definidos por inducción:

Definir por inducción el conjunto de los naturales impares (I).

i) $1 \in I$ (base)

ii) Si $n \in I$ entonces $(n + 2) \in I$ (inductiva)

iii) Extrema

Definir por inducción la relación de identidad entre naturales (Id_N).

i) $(0,0) \in Id_N$ (base)

ii) Si $(n,n) \in Id_N$ entonces $(n+1, n+1) \in Id_N$ (inductiva)

iii) Extrema

UNIVERSIDAD DE LA EMPRESA

Ejemplos de Conjuntos definidos por inducción:

2) El conjunto **PROP** de todas las proposiciones se define inductivamente mediante las siguientes cláusulas:

i.	$\bot \in$ **PROP**	(base)
ii.	$\mathbf{p_k} \in$ **PROP**	(base)
iii.	Si $\alpha \in$ **PROP** y $\beta \in$ **PROP** $\Rightarrow$ **(**$\alpha \wedge \beta$**)** $\in$ **PROP**	(inductiva)
iv.	Si $\alpha \in$ **PROP** y $\beta \in$ **PROP** $\Rightarrow$ **(**$\alpha \vee \beta$**)** $\in$ **PROP**	(inductiva)
v.	Si $\alpha \in$ **PROP** y $\beta \in$ **PROP** $\Rightarrow$ **(**$\alpha \rightarrow \beta$**)** $\in$ **PROP**	(inductiva)
vi.	Si $\alpha \in$ **PROP** y $\beta \in$ **PROP** $\Rightarrow$ **(**$\alpha \leftrightarrow \beta$**)** $\in$ **PROP**	(inductiva)
vii.	Si $\alpha \in$ **PROP** $\Rightarrow$ **(**$\neg\alpha$**)** $\in$ **PROP**	(inductiva)
viii.	Extrema	(extrema)

Observación: La cantidad de cláusulas base e inductivas que se necesitan depende del conjunto que se esté definiendo. En cambio, la cláusula extrema siempre es única.

UNIVERSIDAD DE LA EMPRESA

Observaciones:

- Si en una definición inductiva usamos solamente **cláusulas base**, el conjunto puede resultar **finito** o **infinito**.
- Si en una definición inductiva usamos solamente **cláusulas inductivas**, el conjunto siempre resulta **vacío**.
- Si en una definición inductiva usamos ambos tipos de cláusulas (**base** e **inductivas**), el conjunto siempre resulta **infinito**.
- En toda cláusula inductiva, el nombre del conjunto que se está definiendo siempre debe aparecer a **ambos** lados del entonces ($\Rightarrow$).

Secuencias de elementos: Las secuencias son estructuras matemáticas de amplio uso en el terreno de la programación. Una secuencia es una lista de elementos, donde cada elemento tiene una posición y donde puede haber elementos repetidos.

Las secuencias se suelen denotar informalmente entre corchetes, veamos algunos ejemplos:

- [3, 10, 2, 3, 25] es una secuencia de naturales – **Sec (N)**
- [a, m, c, h] es una secuencia de caracteres – **Sec (Char)**
- [(5,b), (1,c), (2,b)] es una secuencia de pares ordenados de naturales con caracteres – **Sec (N x Char)**
- [ ] es una secuencia **vacía**

Observación: No confundir una **secuencia** con un **conjunto**. Son conceptos distintos.

Secuencias de elementos (continuación): La notación de corchetes es una notación informal, pero para definir una secuencia de manera formal se hace uso de dos constructores denominados **nil** y **cons**.

- **nil** representa la secuencia vacía
- **cons** permite agregar un nuevo elemento al principio de una secuencia. Así, dados una secuencia **s**, y un nuevo elemento **x**, la notación **cons (x, s)** denota la nueva secuencia que resulta de insertar el nuevo elemento **x** delante de la secuencia **s**.

Ejemplos de secuencias:

1) **Escribir la secuencia [8] de manera formal usando los operadores nil y cons.**

2) **Escribir la secuencia [3, 5, 2] de manera formal usando los operadores nil y cons.**

UNIVERSIDAD DE LA EMPRESA

Ejemplos de secuencias:

1) **Escribir la secuencia [8] de manera formal usando los operadores nil y cons.**

   cons (8, nil)

2) **Escribir la secuencia [3, 5, 2] de manera formal usando los operadores nil y cons.**

   cons (3, cons (5, cons (2, nil)))

Secuencias de elementos (continuación): Utilizando los operadores **nil** y **cons** es posible definir inductivamente el conjunto de todas las secuencias posibles de la siguiente manera:

Dado un conjunto **A**, el conjunto **Sec(A)** de las secuencias de elementos de tipo **A** se define inductivamente mediante las siguientes cláusulas:

i. **nil** $\in$ **Sec(A)** (base)
ii. Si **x** $\in$ **A** y **s** $\in$ **Sec(A)** $\Rightarrow$ **cons (x,s)** $\in$ **Sec(A)** (inductiva)
iii. Extrema (extrema)

Ejemplos de aplicación de la definición de Sec(A):

1) **Demostrar que la secuencia [3, 5, 2] $\in$ Sec(N) utilizando la definición inductiva anterior.**

2) **Demostrar que la secuencia [a, 3, 6] $\notin$ Sec(N) utilizando la definición inductiva anterior.**

UNIVERSIDAD DE LA EMPRESA

Ejemplos de aplicación de la definición de Sec(A):

1) **Demostrar que la secuencia [3, 5, 2] $\in$ Sec(N) utilizando la definición inductiva anterior.**

   Por cláusula i), nil $\in$ Sec(N)

   Luego, por cláusula ii), cons (2, nil) $\in$ Sec(N)

   Luego, por cláusula ii), cons (5, cons (2, nil)) $\in$ Sec(N)

   Luego, por cláusula ii), cons (3, cons (5, cons (2, nil))) $\in$ Sec(N)

2) **Demostrar que la secuencia [a, 3, 6] $\notin$ Sec(N) utilizando la definición inductiva anterior.**

   cons (a, cons (3, cons (6, nil))) $\notin$ Sec(N) porque a $\notin$ N, por lo tanto no cumple con la cláusula ii)

UDE
UNIVERSIDAD DE LA EMPRESA

Recursividad

Recursividad: Es un mecanismo matemático que permite definir funciones. Consiste en que la función se utilice a sí misma con valores más pequeños a efectos de calcular el resultado. Veremos, mediante un ejemplo introductorio, cómo funciona este mecanismo.

Ejemplo introductorio: Se desea definir la función **Fact : N → N** que, dado un natural cualquiera, calcula el valor de su factorial.

Hagamos el cálculo para algunos casos concretos:

Fact (0) = 0! = 1
Fact (1) = 1! = 1
Fact (2) = 2! = 2 * 1
Fact (3) = 3! = 3 * 2 * 1
Fact (4) = 4! = 4 * 3 * 2 * 1
Fact (5) = 5! = 5 * 4 * 3 * 2 * 1

UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplo introductorio (continuación):

Obsérvese que, para calcular el factorial de un número, se hace uso del valor del factorial del número inmediatamente anterior:

Fact (0) = 0! = 1	
Fact (1) = 1! = 1	= 1 * Fact (0)
Fact (2) = 2! = 2 * 1	= 2 * Fact (1)
Fact (3) = 3! = 3 * 2 * 1	= 3 * Fact (2)
Fact (4) = 4! = 4 * 3 * 2 * 1	= 4 * Fact (3)
Fact (5) = 5! = 5 * 4 * 3 * 2 * 1	= 5 * Fact (4)

Generalizando este cálculo a cualquier natural, se obtiene el siguiente caso general para la función factorial:

Fact: N $\rightarrow$ N	(dominio y codominio de la función)
Fact (0) = 1	(paso **base** de la función)
Fact (s(n)) = s(n) * Fact (n)	(paso **recursivo** de la función)

UNIVERSIDAD DE LA EMPRESA

Pasos base y pasos recursivos:

Una función definida en forma **recursiva** debe poseer **pasos base** y **pasos recursivos**.

Un **paso base** es una igualdad que, dado un valor, devuelve el resultado para dicho valor en forma directa, sin necesidad de realizar ningún cálculo.

Fact (0) = 1 (paso **base** de la función)

Un **paso recursivo** es una igualdad que, dado un valor, calcula el resultado para dicho valor utilizando el resultado devuelto por la ***misma función que se está definiendo***, pero aplicada a valores más pequeños que el valor actual.

Fact (s(n)) = s(n) * Fact (n) (paso **recursivo** de la función)

UDE
UNIVERSIDAD DE LA EMPRESA

Criterios para definir correctamente funciones recursivas:

- Toda función definida en forma recursiva debe poseer **por lo menos un paso base (*criterio de terminación*)**
- Cada paso recursivo debe ser tal que utilice recursivamente a la misma función con valores **más pequeños *(criterio de acercamiento a algún paso base)***
- La cantidad de pasos base y de pasos recursivos depende de la función que se esté definiendo, pero siempre debe haber al menos uno de cada uno (para que sea efectivamente recursiva).

Ejemplos de funciones recursivas:

1) **Definir de forma recursiva la función sum: N → N que, dado un natural cualquiera, calcula la suma de todos los naturales menores o iguales que él.**
2) **Calcular el resultado de sum (4) aplicando paso a paso la definición de la función.**

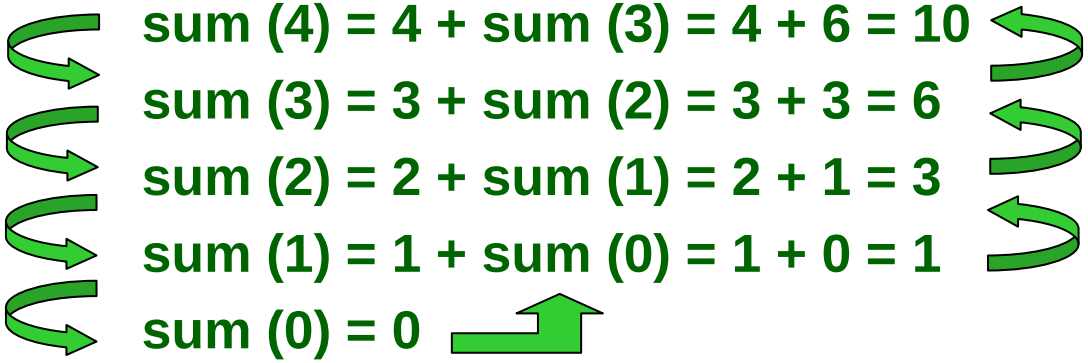
UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplos de funciones recursivas:

1) **Definir de forma recursiva la función sum: N → N que, dado un natural cualquiera, calcula la suma de todos los naturales menores o iguales que él.**

   sum: N → N
   sum (0) = 0 (paso **base**)
   sum (s(n)) = s(n) + sum (n) (paso **recursivo**)

2) **Calcular el resultado de sum (4) aplicando paso a paso la definición de la función.**



UDE
UNIVERSIDAD DE LA EMPRESA

Relación entre funciones recursivas y conjuntos inductivos:

Se cumple **siempre** que el dominio de toda ***función recursiva*** es necesariamente un ***conjunto inductivo***. Es decir, las funciones recursivas se definen exclusivamente sobre elementos de algún conjunto definido por inducción.

La definición inductiva del conjunto puede ser de utilidad a la hora de pensar los **pasos base** y los **pasos recursivos** de la función:

- Los **pasos base** de la función se pueden definir a partir de las **cláusulas base** del conjunto.
- Los **pasos recursivos** de la función se pueden definir a partir de las **cláusulas inductivas** del conjunto.

Observación: La cantidad de cláusulas base (inductivas) establece la cantidad **mínima** de pasos base (recursivos) de la función.

UDE
UNIVERSIDAD DE LA EMPRESA

Relación entre funciones recursivas y conjuntos inductivos:

Por ejemplo:

Definición inductiva	Función recursiva
Conjunto de naturales **N**	Función **Fact: N → N**
i. **0 ∈ N**	**Fact (0) = 1**
ii. Si **n ∈ N ⇒ s(n) ∈ N**	**Fact (s(n)) = s(n) * Fact (n)**

El valor **0** definido en la ***cláusula base*** del conjunto es usado en el ***paso base*** de la función.

Relación entre funciones recursivas y conjuntos inductivos:

Por ejemplo:

Definición inductiva	Función recursiva
Conjunto de naturales **N**	Función **Fact: N → N**
i. **0 ∈ N**	**Fact (0) = 1**
ii. Si **n ∈ N ⇒ s(n) ∈ N**	**Fact (s(n)) = s(n) * Fact (n)**

El valor **n** definido a la ***izquierda del*** ⇒ en la ***cláusula inductiva*** del conjunto es usado a la ***derecha del =*** en el ***paso recursivo*** de la función.

Relación entre funciones recursivas y conjuntos inductivos:

Por ejemplo:

Definición inductiva	Función recursiva
Conjunto de naturales **N**	Función **Fact: N → N**
i. **0 ∈ N**	**Fact (0) = 1**
ii. Si **n ∈ N ⇒ s(n) ∈ N**	**Fact (s(n)) = s(n) * Fact (n)**

El valor **s(n)** definido a la ***derecha del*** ⇒ en la ***cláusula inductiva*** del conjunto es usado a la ***izquierda del =*** en el ***paso recursivo*** de la función.

Concordancia de Patrones

Concordancia de Patrones: Es un método para definir funciones de manera formal. Consiste en plantear igualdades que indican cómo calcular el resultado para patrones de valores determinados.

Para una función **func** de dominio **A** y codominio **B**, la estructura de su definición por concordancia de patrones tiene la siguiente forma:

func: A → B
func (patrón1) = expresión1
func (patrón2) = expresión2
…
func (patrónN) = expresiónN

Cada **patrón** representa un valor perteneciente al dominio. El patrón corresponde a la **forma** que posee dicho valor.

Cada **expresión** es un valor o bien una cadena de operadores y operandos que permite calcular el resultado de la función para el valor de entrada dado por el patrón.

UNIVERSIDAD DE LA EMPRESA

Ejemplos de funciones por Concordancia de Patrones:

1) Definimos la función **esVacía** que, dada una secuencia de valores cualquiera, indica si la misma está vacía o no.

esVacía: Sec(A) → boolean
esVacía (nil) = true
esVacía (cons(x,s)) = false

El primer patrón es **nil** y corresponde a la secuencia vacía. La imagen de la secuencia vacía es el valor **true**. El segundo patrón es **cons(x,s)** y corresponde a cualquier secuencia que no es vacía. La imagen correspondiente es **false**.

- **Calcular esVacia ([3]) y esVacia ([1, 2]) usando la definición.**
- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**

UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplos de funciones por Concordancia de Patrones:

- **Calcular esVacia ([3]) y esVacia ([1, 2]) usando la definición.**

  esVacia ([3]) = esVacia (cons (3, nil)) = false
  esVacia ([1, 2]) = esVacia (cons (1, cons (2, nil)) = false

- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**

  Sí es total, ya que para cualquier secuencia la función produce un resultado. Para la secuencia vacía, el 1º patrón lo produce. Para cualquier otra, el 2º patrón lo produce.

  No es inyectiva, ya que hay dos secuencias distintas con el mismo resultado (por ejemplo, [3] y [1,2])

  Sí es sobreyectiva, ya que ambos elementos del codominio (false y true) tienen al menos una preimagen en la función

UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplos de funciones por Concordancia de Patrones:

2) Definimos la función **fact** que, dado un natural cualquiera, calcula el valor de su factorial.

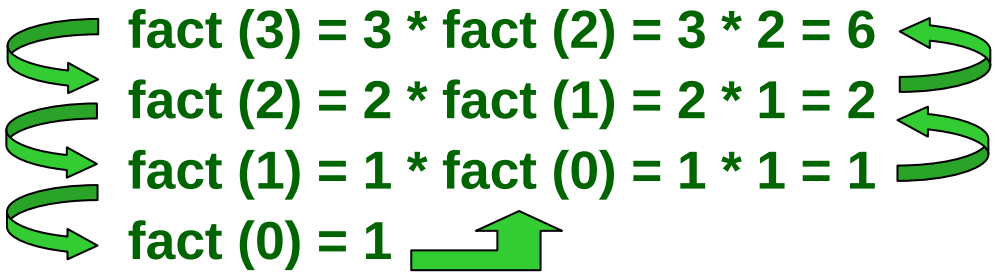
fact: N → N
fact (0) = 1
fact (s(n)) = s(n) * fact (n)

El primer patrón es **0** y corresponde al valor cero. La imagen del cero es el valor **1**. El segundo patrón es **s(n)** y corresponde a cualquier natural que no sea el cero. La imagen correspondiente es el resultado de la expresión **s(n) * fact (n)**.

- **Calcular fact (3) usando la definición.**
- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**

Ejemplos de funciones por Concordancia de Patrones:

- **Calcular fact (3) usando la definición.**



- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**

  Sí es total, ya que para cualquier natural la función produce un resultado. Para el 0, el 1º patrón lo produce. Para cualquier otro, el 2º patrón lo produce.

  No es inyectiva, ya que hay dos naturales distintos con el mismo resultado (0 y 1).

  No es sobreyectiva, ya que naturales en el codominio sin preimagen en el dominio (ejemplo: 3).

UNIVERSIDAD DE LA EMPRESA

Observaciones:

- El método de concordancia de patrones es una extensión a la ***notación prefija*** para definir funciones, vista antes en el curso.
- Cada **patrón** debe aparecer en ***una sola igualdad***. Ningún patrón debe aparecer en más de una igualdad.
  - **¿Qué riesgo se corre si esto no se cumple?**
- Las funciones pueden ser ***recursivas*** (por ejemplo: **fact**) o no serlo (por ejemplo: **esVacía**). En caso de serlo, deben ***respetar los criterios para funciones recursivas*** vistos anteriormente.
- En cada **expresión** (lado derecho de la igualdad) se permite…
  - Llamar a otras funciones definidas anteriormente.
  - Utilizar selección simple (**Si** / **Sino**) cuando se quiere hallar el resultado dependiendo de una condición booleana.
  - **NO** se permite usar iteraciones (**Mientras**, **Para cada**, etc.)

UNIVERSIDAD DE LA EMPRESA

Más ejemplos de funciones por Concordancia de Patrones:

3) Definimos la función **largo** que, dada una secuencia de valores cualquiera, cuenta cuántos elementos posee.

largo: Sec(A) → N
largo (nil) = 0
largo (cons (x,s)) = 1 + largo(s)

- **Calcular largo ([a, b, c]) usando la definición.**
- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**
- **La función ¿es recursiva?**

Más ejemplos de funciones por Concordancia de Patrones:

- **Calcular largo ([a, b, c]) usando la definición.**

largo (cons (a, cons (b, cons (c, nil)))) =
1 + largo (cons (b, cons (c, nil))) = 1 + 2 = 3

largo (cons (b, cons (c, nil))) =
1 + largo (cons (c, nil)) = 1 + 1 = 2

largo (cons (c, nil)) =
1 + largo (nil) = 1 + 0 = 1

largo (nil) = 0

UDE
UNIVERSIDAD DE LA EMPRESA

Más ejemplos de funciones por Concordancia de Patrones:

- La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?
- La función ¿es recursiva?

Sí es total, ya que para cualquier secuencia la función produce un resultado. Para la secuencia vacía, el 1º patrón lo produce. Para cualquier otra, el 2º patrón lo produce.

No es inyectiva, ya que hay dos secuencias distintas con el mismo resultado (por ejemplo, [4,7,3] y [1,5,2]).

Sí es sobreyectiva, ya que cualquier natural del codominio es el largo de alguna secuencia en el dominio (0 es el largo de la secuencia vacía, 1 es el largo de cualquier secuencia con un elemento, 2 es el largo de cualquier secuencia con dos elementos, etc.)

Sí es recursiva, ya que se utiliza a sí misma.

UNIVERSIDAD DE LA EMPRESA

Más ejemplos de funciones por Concordancia de Patrones:

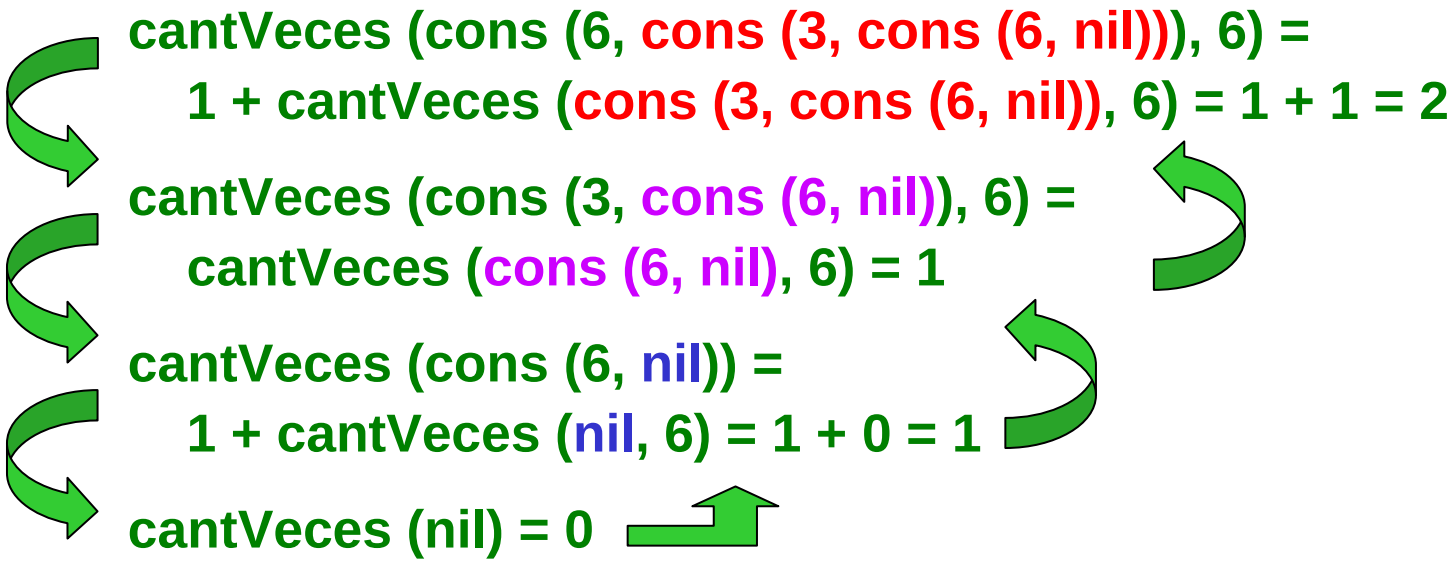
4) Definimos la función **cantVeces** que, dados una secuencia de valores cualquiera y un valor, cuenta cuántas veces aparece dicho valor en la secuencia.

cantVeces: Sec(A) x A → N
cantVeces (nil, y) = 0
cantVeces (cons (x,s), y) = Si (x == y)
1 + cantVeces (s, y)
Sino
cantVeces (s, y)

- **Calcular cantVeces ([6, 3, 6], 6) usando la definición.**
- **La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?**
- **La función ¿es recursiva?**

Más ejemplos de funciones por Concordancia de Patrones:

- **Calcular cantVeces ([6, 3, 6], 6) usando la definición.**



UDE
UNIVERSIDAD DE LA EMPRESA

Más ejemplos de funciones por Concordancia de Patrones:

- La función ¿es total? ¿es inyectiva? ¿es sobreyectiva?
- La función ¿es recursiva?

Sí es total, ya que para cualquier secuencia la función produce un resultado. Para la secuencia vacía, el 1º patrón lo produce. Para cualquier otra, el 2º patrón lo produce.

No es inyectiva, ya que hay dos secuencias distintas con el mismo resultado (por ejemplo, [4,3,3] y [5,2,5]).

Sí es sobreyectiva, ya que cualquier natural del codominio es la cantidad de veces que algún elemento aparece en una secuencia (hay elementos que aparecen 0 veces, 1 vez, 2 veces, 3 veces, etc.)

Sí es recursiva, ya que se utiliza a sí misma.

UNIVERSIDAD DE LA EMPRESA

Definir las siguientes funciones Concordancia de Patrones:

a) **suma: N x N → N la cual, dados dos naturales cualesquiera, retorna el valor de su suma.**

b) **primero: Sec(A) → A la cual, dada una secuencia cualquiera, retorna el primer valor almacenado en ella.**

c) **sumPares: Sec(N) → N la cual, dada una secuencia de naturales cualquiera, suma todos los valores pares que hay en ella.**

- **Las funciones definidas…**
  - ❖ **¿Son totales? ¿Son inyectivas? ¿Son sobreyectivas?**
  - ❖ **¿Son recursivas?**

UDE
UNIVERSIDAD DE LA EMPRESA

Definir las siguientes funciones Concordancia de Patrones:

a) **suma: N x N → N la cual, dados dos naturales cualesquiera, retorna el valor de su suma.**

suma: N x N → N
suma (m, n) = m + n

Sí es total, ya que para cualquier pareja de naturales, la función produce un resultado. Sin importar cuánto valgan m y n, siempre es posible sumarlos.

No es inyectiva, ya que hay parejas distintas de naturales con el mismo resultado (por ejemplo, (2,3) y (3,2)).

Sí es sobreyectiva, ya que cualquier natural se puede obtener como resultado de sumar otros dos naturales (por ejemplo, sumando 0 con el propio natural).

No es recursiva, pues no se utiliza a sí misma.

Definir las siguientes funciones Concordancia de Patrones:

b) primero: Sec(A) → A la cual, dada una secuencia cualquiera, retorna el primer valor almacenado en ella.

primero: Sec(A) → A
primero (cons (x,s)) = x

No es total, para la secuencia vacía no produce resultado.

No es inyectiva, ya que hay secuencias distintas con el mismo resultado (por ejemplo, [4,2,7] y [4,8,5]).

Sí es sobreyectiva, ya que cualquier valor del conjunto A puede ocupar la 1° posición de alguna secuencia.

No es recursiva, ya que no se utiliza a sí misma.

UNIVERSIDAD DE LA EMPRESA

Definir las siguientes funciones Concordancia de Patrones:

c) sumPares: Sec(N) → N la cual, dada una secuencia de naturales cualquiera, suma todos los valores pares que hay en ella.

```
sumPares: Sec(N) → N
sumPares (nil) = 0
sumPares (cons (x,s)) = SI (x % 2 == 0) entonces
                            x + sumPares (s)
                        SINO
                            sumPares (s)
```

UNIVERSIDAD DE LA EMPRESA

Definir las siguientes funciones Concordancia de Patrones:

c) sumPares: Sec(N) → N la cual, dada una secuencia de naturales cualquiera, suma todos los valores pares que hay en ella.

Sí es total, ya que para cualquier secuencia la función produce un resultado. Para la secuencia vacía, el 1º patrón lo produce. Para cualquier otra, el 2º patrón lo produce.

No es inyectiva, ya que hay dos secuencias distintas con el mismo resultado (por ejemplo, [6,2,5] y [4,1,4]).

No es sobreyectiva, cualquier natural *impar* del codominio nunca puede ser el resultado de sumar los valores *pares* de una secuencia (por ejemplo, ninguna suma de pares da 3).

Sí es recursiva, ya que se utiliza a sí misma.

Demostraciones por inducción estructural

UDE
UNIVERSIDAD DE LA EMPRESA

Antes en el curso hemos visto los siguientes métodos de prueba:

- **Prueba directa, Por absurdo, Por contrarrecíproco**

Veremos ahora un 4º método de prueba que sirve para demostrar propiedades sobre elementos de conjuntos ***inductivos***.

Método de inducción estructural: Sea $\mathbf{I}$ un conjunto definido inductivamente y sea **P** una propiedad. Para probar que todos los elementos de $\mathbf{I}$ cumplen la propiedad **P** hay que demostrar que:

Paso base: Los elementos definidos en las cláusulas ***base*** de $\mathbf{I}$ cumplen la propiedad **P**.

Paso inductivo: Si los elementos definidos ***antes*** del $\Rightarrow$ en las cláusulas inductivas cumplen la propiedad **P**, (***hipótesis inductiva***) entonces los elementos definidos ***después*** del $\Rightarrow$ en las cláusulas inductivas cumplen la propiedad **P**. (***tesis inductiva***)

UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplo: Aplicación del método cuando I = N (naturales):

Recordemos la definición inductiva del conjunto…

i. **0** ∈ **N** (base)

ii. Si **n** ∈ **N** ⇒ **s(n)** ∈ **N** (inductiva)

Sea **P** la propiedad que se desea demostrar. De acuerdo con el método, el formato de la prueba será el siguiente:

Paso base: Probar que el **0** cumple la propiedad **P**.

Dem: … *aquí se demuestra que el **0** cumple **P** …*

Paso inductivo:

Hipótesis inductiva: Asumimos que **n** cumple **P**.

Tesis inductiva: Debemos probar que **s(n)** cumple **P**.

Dem: … *aquí se demuestra que **s(n)** cumple **P**, partiendo de la hipótesis que dice que **n** cumple **P** …*

Observaciones:

- La cantidad de **pasos base** a demostrar coincide con la cantidad de **cláusulas base** del conjunto
- La cantidad de **pasos inductivos** a demostrar coincide con la cantidad de **cláusulas inductivas**.
- Para el caso de los naturales hay un solo paso base porque hay una sola cláusula base, y hay un solo paso inductivo porque hay una sola cláusula inductiva.

Ejemplo: **Se desea demostrar que $\forall n \in N$ se cumple que $n \leq 2^n$.**

- **¿Cuál es la propiedad P en este caso?**
- **Plantear el paso base y demostrarlo.**
- **Plantear el paso inductivo (hipótesis inductiva y tesis inductiva) y demostrarlo.**

Ejemplo: Se desea demostrar que $\forall n \in N$ se cumple que $n \leq 2^n$.

- ¿Cuál es la propiedad P en este caso? $P = n \leq 2^n$.
- Plantear el paso base y demostrarlo.

  Paso base: $0 \leq 2^0$.

  Dem:

  Sabemos que $2^0 = 1 \Rightarrow$ Se cumple que $0 \leq 2^0$

- Plantear el paso inductivo (hipótesis inductiva y tesis inductiva) y demostrarlo.

  Hipótesis inductiva: $n \leq 2^n$.

  Tesis inductiva: $s(n) \leq 2^{s(n)}$.

  Dem:

$$s(n) = n + 1 \leq 2^n + 1 \leq 2^n + 2^n = 2.2^n = 2^{n+1} = 2^{s(n)}$$

(def. sucesor) (H.I.) ($1 \leq 2^n$) (prop. potencia) (def.sucesor)

UDE
UNIVERSIDAD DE LA EMPRESA

Ejemplo: Aplicación del método cuando I = Sec(A):

Recordemos la definición inductiva del conjunto…

i. **nil** ∈ **Sec(A)** (base)
ii. Si x∈ **A y s** ∈ **Sec(A)** ⇒ **cons (x,s)** ∈ **Sec(A)** (inductiva)

Sea **P** la propiedad que se desea demostrar. De acuerdo con el método, el formato de la prueba será el siguiente:

Paso base: Probar que **nil** cumple la propiedad **P**.

Dem: … *aquí se demuestra que **nil** cumple **P*** …

Paso inductivo:

Hipótesis inductiva: Asumimos que **s** cumple **P**.

Tesis inductiva: Debemos probar que **cons (x,s)** cumple **P**.

Dem: … *aquí se demuestra que **cons (x,s)** cumple **P**, partiendo de la hipótesis que dice que **s** cumple **P*** …

Ejemplo: Dadas las siguientes funciones…

- **doble: Sec(N) → Sec(N)**
  doble (nil) = nil
  doble (cons (x,s)) = cons (2x, doble (s))
- **sumar: Sec(N) → N**
  sumar (nil) = 0
  sumar (cons (x,s)) = x + sumar (s)

Se desea demostrar que:

$\forall s \in$ Sec(N) se cumple que sumar (doble (s)) = 2 * sumar (s).

- **¿Cuál es la propiedad P en este caso?**
- **Plantear el paso base y demostrarlo.**
- **Plantear el paso inductivo (hipótesis inductiva y tesis inductiva) y demostrarlo.**

$\forall s \in$ Sec(N) se cumple que sumar (doble (s)) = 2 * sumar (s).

- ¿Cuál es la propiedad P en este caso?

  P = sumar (doble (s)) = 2 * (sumar (s))

- Plantear el paso base y demostrarlo.

  Paso base: sumar (doble (nil)) = 2 * sumar (nil).

  Dem:

  sumar (doble (nil)) = sumar (nil) = 0 (*)

  (p.base doble) (p.base sumar)

  2 * sumar (nil) = 2 * 0 = 0 (**)

  (p.base sumar)

  Ambas expresiones (*) y (**) son iguales $\Rightarrow$ se cumple

$\forall s \in$ Sec(N) se cumple que sumar (doble (s)) = 2 * sumar (s).

- Plantear el paso inductivo (H.I. y T.I.) y demostrarlo.

Hipótesis inductiva: sumar (doble (s)) = 2 * sumar (s).

Tesis inductiva: sumar (doble (cons (x,s))) = 2 * sumar (cons (x,s)).

Dem:

sumar (doble (cons (x,s))) = (p.recu doble) sumar (cons (2x, doble (s))) = (p.recu sumar)

2x + sumar (doble (s)) = 2x + 2 * sumar (s) (*)

(H.I.)

2 * sumar (cons (x,s)) = (p.recu sumar) 2 * (x + sumar (s)) = (prop. distributiva) 2x + 2 * sumar (s) (**)

Ambas expresiones (*) y (**) son iguales $\Rightarrow$ se cumple