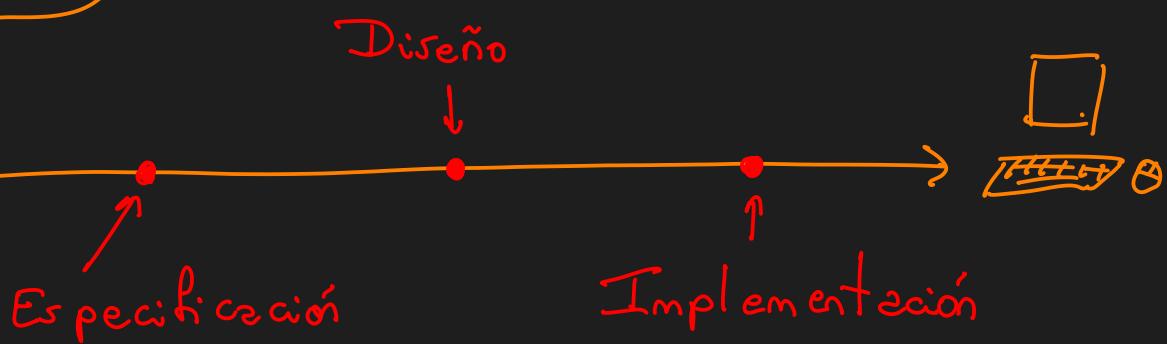


## Algo 2. Teórica

Dr. Esteban Feuerstein



Intro



• Problemas Fundamentales

↳ Organización y Acceso a la Información.

Especificación ↓

Algoritmo

“Secuencia ordenada y finita de pasos bien determinados que nos llevan de un estado inicial a uno final”

Resolver un problema genérico

Dada una descripción → Proponer un algoritmo que resuelve cualquier instancia del problema

# Ejemplos.

- Descripción del problema: "Dados cuatro reales encontrar al entero más chico que esté por encima del mínimo cuadrado perfecto mayor que el mínimo de los dos primeros pero menor que la suma de los otros dos".

Problema ideaTresnochada ( $a, b, c, d$ ) = resultado : int {  
    requiere  $a > 0 \wedge b > 0$ ;      $\swarrow x$  cuadrado perfecto.  
    asegura  $z = \min \{x / \frac{\sqrt{x}}{\text{int}(\sqrt{x})} = 1 \wedge$   
                 $\min(a, b) < x < c + d \wedge$   
                resultado =  $\min \{w / w \geq z\}$   
    }

Sigue

Dr. Emmanuel Tarusi



- Descripción: "El dueño de un restaurante quiere asegurarse de que los pedidos de sus clientes sean atendidos con prontitud. Los mozos llevan los pedidos hasta la cocina donde los colocan en un rotador de tarjetas (tipo rolodex). Cuando el cocinero se libera, saca la primera y prepara el plato allí indicado. El dueño quiere saber cuál es el próximo plato a preparar y cuántos pedidos atiende el cocinero cada día, y cuál fue el día con menos pedidos."



Algoritmo : ?

Especificación : ?



Del ej:

Hay plato, dia y (el menor 1) restaurant.

Renombrar:

- TAD DÍA ES NAT
- TAD PLATO ES STRING
- TAD RESTAURANT *"forma de referirnos al tipo"*  
géneros restaurant

operación es *no recibe paráms* *ddevuelve*  
inaugurar:  $\rightarrow$  restaurant

cent-plato-pendiente: restaurant  $\rightarrow$  nat

próx - pedido: restaurant  $r$   $\rightarrow$  plato  
 $\{ \text{cent-plato-pend.}(r) > 0 \}$  *restricciones*

preparar - plato: restaurant  $r$   $\rightarrow$  restaurant  
 $\{ \text{cent-plato-pend.}(r) > 0 \}$

tomar - pedido: restaurant  $\times$  plato  $\rightarrow$  restaurant  
*dos parámetros de entrada*

⋮

FIN TAD

(Esteban)

Obs :

Las operaciones de los TADs son funciones totales

↳ definir para cada valor del dominio

(por eso debemos restringir el dominio  $\{ \}$ )

Necesitamos darle semántica, comportamiento a las operaciones

↳ en TADs, usaremos axiomas

ej:  $\text{día\_actual}(\text{nuevo\_día}(r)) \equiv \text{día\_actual}(r) + 1$

• Expresan reescrituras de "términos" (def. más adelante)

## Términos

"Lo que resulta de aplicar operaciones sin parámetros, o bien

Lo que resulta de aplicar operaciones con parámetros a parámetros que son términos correctos."

ej:  $\begin{array}{c} \text{tip} \\ \downarrow \\ a : x \end{array}$

b : y

$f : x \times y \rightarrow z$

$f(a, b) : z$

↑ término correcto si a y b son términos correctos.

Axiomas

Escribir las axiomas para las operaciones de enter

$$\text{días-actual}(\text{iniciar}()) \equiv 0$$

$$(\forall r : \text{restaurant}) \quad \text{días-actual}(\text{nuevo-día}(r)) \equiv \text{días-actual}(r) + 1$$

⋮

Nota: abusaremos de la notación y no escribiremos las quantificadas de las variables libres

$$\text{días-actual}(\text{nuevo-día}(r)) \equiv \text{días-actual}(r) + 1$$

↑ variable libre; resume cantidad fijada  $\forall$

(ver más por para especificación completa.)

$$\text{platos-por-día}(d, \text{iniciar}()) \equiv 0$$

$$\text{platos-por-día}(d, \text{tomar-pedidos}(r, p)) \equiv \text{platos-por-día}(d, r)$$

$$\text{platos-por-día}(d, \text{preparar-plato}(r)) \equiv$$

if  $\text{días-actual}(r) \equiv d$  then

$$\text{platos-por-día}(d, r) + 1$$

else

$$\text{platos-por-día}(d, r)$$

fi

## Otras formas de axiomatizar

$$(\forall d : \text{dia}) \quad 0 \leq d \leq \text{dia-actual}(r) \Rightarrow_L$$

$$\text{platos-por-dia}(r, \text{dia-menor-pedidos}(r)) \leq \text{platos-por-dia}(r, d')$$

(Emma)

TADs permite inducción estructural.

↳ "cada TAD constituye una teoría de primer orden con igualdad"



↳ dominar los tiene gran transcripción a POO

"Paréntesis" lógicos

Variables libres  $\xrightarrow{\text{rango } R(x)}$   $\xrightarrow{\text{predicado } P(x)}$

$$(\forall x : \text{nat}) \quad (1 \leq x) \Rightarrow_L \frac{x}{x} = 1$$

$$(\forall x : \text{género}) \quad (R(x) \Rightarrow_L P(x))$$

Con existencial  $\exists$   $\nwarrow$  notar: si  $R(x) = \emptyset \Rightarrow$  es verdadero!

$$(\exists x : \text{género}) \quad (R(x) \wedge_L P(x))$$

Macros

$$(\forall x : \text{género}, R(x)) \quad P(x) \equiv$$

$$(\forall x : \text{género} \mid R(x)) \quad P(x) \equiv$$

$$(\forall x : \text{género}) \quad (R(x) \Rightarrow_L P(x))$$

# Tareas: específicas

- Existen naturales pares menores que 33.

$$\left( \exists x : \text{nat}, \quad x < 33 \wedge x \bmod 2 = 0 \right)$$

- Todos las secuencias de longitud par se pueden escribir como la concatenación de otras dos secuencias.
- Todos los números primos tienen un elemento mayor y otro menor.
- Cada secuencia de naturales puede ser extendida en un elemento en tantas secuencias como naturales.
- Todo  $Z_n$  tiene su neutro aditivo.
- Hay un neutro aditivo e para todo  $Z_n$ .

(Esteban)

TAD Bool

géneros bool

tendrán significado semántico  
✓ al definir los axiomas.

exporta bool, generadores,  $\neg$ ,  $\vee$ ,  $\wedge$ ,  $\Rightarrow$

generadores

true :  $\rightarrow \text{bool}$

false :  $\rightarrow \text{bool}$

otras operaciones

$\neg \bullet : \text{bool} \rightarrow \text{bool}$

$\bullet \vee \bullet : \text{bool} \times \text{bool} \rightarrow \text{bool}$

$\bullet \wedge \bullet : \text{bool} \times \text{bool} \rightarrow \text{bool}$

$\bullet \Rightarrow \bullet : \text{bool} \times \text{bool} \rightarrow \text{bool}$

↑  
por sintaxis, los axiomas correspondientes:

axiomas

$\neg \text{true} \equiv \text{false}$

$\neg \text{false} \equiv \text{true}$

$(\forall x : \text{bool}) \text{ true } \vee x \equiv \text{true}$

$(\forall x : \text{bool}) \text{ false } \vee x \equiv x$

$(\forall x : \text{bool}) \text{ true } \wedge x \equiv x$

$(\forall x : \text{bool}) \text{ false } \wedge x \equiv \text{false}$

$(\forall x, y : \text{bool}) x \Rightarrow y \equiv \neg x \vee y$

- Géneros (en general hay solo 1)

Nombre que recibe el conjunto de valores del tipo o conjunto de instancias del tipo

Distinto al "nombre del TAD"

ej:  $(\mathbb{N}, +)$  vs  $\mathbb{N}$  ↘

↑ género

- Usz

Inducción de géneros y operaciones **exportadas** en los tipos mencionados

- Exporte

Operaciones y géneros a disposición del usuario del tipo

- Generadores

Operaciones que permiten construir instancias del tipo.

Un conjunto de generadores está bien armado / definido si alguna combinación de ellos permite construir cualquier instancia posible del tipo.

- Axiomas

Reglas que nos explican el comportamiento de las funciones.

( $\Sigma_{\text{NAT}}$ )

TAD NAT

$(\forall n, m : \text{nat})$

$0 = 0 ? \equiv \text{true}$

$\text{suc}(n) = 0 ? \equiv \text{false}$

$\text{pred}(\text{suc}(n)) \equiv n$

$n + m \equiv \begin{cases} \text{if } m = 0 ? \text{ then } n \\ \text{else } \text{suc}(n + \text{pred}(m)) \end{cases}$

fi

.

:

$n = m \equiv (n = 0? \equiv m = 0?) \wedge_L (\neg(m = 0?) \Rightarrow_L (\text{pred}(n) = \text{pred}(m)))$

$n < m \equiv \text{if } m = 0? \text{ then }$

*false*

**else**

**if**  $n = 0?$  **then**

*true*

**else**

$\text{pred}(n) < \text{pred}(m)$

**fi**

**fi**

T<sup>AD</sup> Bool Extendido ( $\alpha$ )

extiende Bool

l parámetro

otras operaciones

if • then • else • fi : bool  $\times \alpha \times \alpha \rightarrow \alpha$

:

ejemplos

( $\forall x, y : \text{bool}$ ), ( $\forall a, b : \alpha$ )

↓

if true then a else b fi  $\equiv$  a

if false then a else b fi  $\equiv$  b

:











