Tipos abstractos de datos básicos

Algoritmos y Estructuras de Datos II, DC, UBA.

1. TAD Tablero

TAD Tablero

géneros tablero

exporta bool, generadores, $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\vee_L$, $\wedge_L$, $\Rightarrow_L$

igualdad observacional

$$(\forall t, t' : \text{tablero}) \left(t =_{\text{obs}} t' \iff \begin{pmatrix} (\#casilleros(t) =_{\text{obs}} \#casilleros(t')) \wedge_L \\ (\forall (c, c' : nat)) c, c' \leq \#casilleros(t) \Rightarrow_L \\ (cont(c,t) =_{\text{obs}} cont(c,t') \wedge \\ movsDeA(c,c',t) =_{\text{obs}} movsDeA(c,c',t'))) \end{pmatrix} \right)$$

generadores

crearTablero : continente $\times$ continente $\times$ mov $\times$ mov $\longrightarrow$ tablero

agregarCasillero : casillero c $\times$ continente k $\times$ mov m $\times$ mov m' $\times$ tablero t $\longrightarrow$ tablero

$$\left\{ \begin{array}{l} c \leq \#\text{casilleros}(t) \wedge_L (k =_{\text{obs}} cont(c,t) \vee ((\forall c' : nat) c' \leq \#\text{casilleros}(t) \Rightarrow_L cont(c',t) \neq k)) \wedge_L \\ m' \notin \text{todosLosMovs}(c,t) \end{array} \right\}$$

conectar : casillero c $\times$ casillero c' $\times$ mov m $\times$ mov m' $\times$ tablero t $\longrightarrow$ tablero

$$\{c, c' \leq \#\text{casilleros}(t) \wedge c \neq c' \wedge_L m \notin \text{todosLosMovs}(c,t) \wedge_L m' \notin \text{todosLosMovs}(c',t)\}$$

agregarFlecha : casillero c $\times$ casillero c' $\times$ mov m $\times$ tablero t $\longrightarrow$ tablero

$$\{c, c' \leq \#\text{casilleros}(t) \wedge_L \text{conectados?}(c,c',t) \wedge m \notin \text{todosLosMovs}(c,t)\}$$

observadores básicos

#casilleros : tablero $\longrightarrow$ nat

cont : casillero c $\times$ tab t $\longrightarrow$ continente $\quad \{c \leq \#casilleros(t)\}$

movsDeA : casillero c $\times$ casillero c' $\times$ tab t $\longrightarrow$ conj(mov) $\quad \{c, c' \leq \#casilleros(t)\}$

otras operaciones

todosLosMovs : casillero c $\times$ tab t $\longrightarrow$ conj(mov) $\quad \{c \leq \#casilleros(t)\}$

conectados? : casillero c $\times$ casillero c' $\times$ tab t $\longrightarrow$ bool $\quad \{c, c' \leq \#casilleros(t)\}$

axiomas $\forall\, t, t'$: tablero

$\#$casilleros(crearTablero(k, k', m, m')) $\equiv$ 2

$\#$casilleros(agregarCasillero(c, k, m, m', t)) $\equiv$ suc($\#$casilleros(t))

$\#$casilleros(conectar(c, c', m, m', t)) $\equiv$ $\#$casilleros(t)

$\#$casilleros(agregarFlecha(c, c', m, t)) $\equiv$ $\#$casilleros(t)

cont(c,crearTablero(k, k', m, m')) $\equiv$ **if** $c = 1$ **then** k **else** k' **fi**

cont(c,agregarCasillero$(\tilde{c}, k, m, m', t)$) $\equiv$ **if** $c =$ suc($\#$casilleros(t)) **then** k **else** cont(c,t) **fi**

cont(c,conectar$(\tilde{c}, \tilde{c}', m, m', t)$) $\equiv$ cont(c,t)

cont(c,agregarFlecha$(\tilde{c}, \tilde{c}', m, t)$) $\equiv$ cont(c,t)

movsDeA(c, c',crearTablero(k, k', m, m')) $\equiv$ **if** $c = 1 \wedge c' = 2$ **then**
    $\{m\}$
else
    if $c = 2 \wedge c' = 1$ **then** $\{m'\}$ **else** $\emptyset$ **fi**
fi

$$
\begin{array}{l}
\text{movsDeA}(c, c', \text{agregarCasillero}(\tilde{c}, k, m, m', t)) \equiv \textbf{if } c = \text{suc}(\#\text{casilleros}(t)) \textbf{ then} \\
\quad \textbf{if } c' = \tilde{c} \textbf{ then } \{m\} \textbf{ else } \emptyset \textbf{ fi} \\
\textbf{else} \\
\quad \textbf{if } c = \tilde{c} \wedge c' = \text{suc}(\#\text{casilleros}(t)) \textbf{ then} \\
\qquad \{m'\} \\
\quad \textbf{else} \\
\qquad \text{movsDeA}(c, c', t) \\
\quad \textbf{fi} \\
\textbf{fi}
\end{array}
$$

$$
\begin{array}{l}
\text{movsDeA}(c, c', \text{conectar}(\tilde{c}, \tilde{c}', m, m', t)) \equiv \textbf{if } c = \tilde{c} \wedge c' = \tilde{c}' \textbf{ then} \\
\quad \{m\} \cup \text{movsDeA}(c, c', t) \\
\textbf{else} \\
\quad \textbf{if } c = \tilde{c}' \wedge c' = \tilde{c} \textbf{ then} \\
\qquad \{m'\} \cup \text{movsDeA}(c, c', t) \\
\quad \textbf{else} \\
\qquad \text{movsDeA}(c, c', t) \\
\quad \textbf{fi} \\
\textbf{fi}
\end{array}
$$

$$
\begin{array}{l}
\text{movsDeA}(c, c', \text{agregarFlecha}(\tilde{c}, \tilde{c}', m, t)) \equiv \textbf{if } c = \tilde{c} \wedge c' = \tilde{c}' \textbf{ then} \\
\quad \{m\} \cup \text{movsDeA}(c, c', t) \\
\textbf{else} \\
\quad \text{movsDeA}(c, c', t) \\
\textbf{fi}
\end{array}
$$

Fin TAD

2. TAD Partida

TAD Partida

géneros partida

exporta nat, generadores, observadores, $+$, $-$, $\times$, $<$, $\leq$, mín, máx

usa Bool

igualdad observacional

$$
(\forall p, p' : \text{partida}) \left(p =_{\text{obs}} p' \iff \left(\begin{array}{l}
(\text{tablero}(p) =_{\text{obs}} \text{tablero}(p') \wedge \\
\#\text{jugadores}(p) =_{\text{obs}} \#\text{jugadores}(p') \wedge \\
((\forall c : nat) c \leq \#\text{casilleros}(\text{tablero}(p)) \Rightarrow_{\text{L}} \\
\text{fichasEnCasillero}(c, p) =_{\text{obs}} \text{fichasEnCasillero}(c, p')) \wedge \\
((\forall j : nat) j \leq \#\text{jugadores}(p) \Rightarrow_{\text{L}} \\
(\text{misión}(j, p) =_{\text{obs}} \text{misión}(j, p)) \wedge \\
\text{fichasPuestas}(j, p) =_{\text{obs}} \text{fichasPuestas}(j, p'))
\end{array} \right) \right)
$$

observadores básicos

tablero : partida $\longrightarrow$ tablero

#jugadores : partida $\longrightarrow$ nat

fichasEnCasillero : casillero c $\times$ partida p $\longrightarrow$ multiconj(jugador) $\{c \leq \#\text{casilleros}(\text{tablero}(p))\}$

misión : jugador j $\times$ partida p $\longrightarrow$ continente $\{j \leq \#\text{jugadores}(p)\}$

fichasPuestas : jugador j $\times$ partida p $\longrightarrow$ nat $\{j \leq \#\text{jugadores}(p)\}$

generadores

crearPartida : tablero t $\times$ nat js $\times$ secu(casillero) cs $\times$ secu(continente) ks $\longrightarrow$ partida

$$
\left\{ \begin{array}{l}
2 \leq js \wedge \text{long}(cs) = \text{long}(ks) = js \wedge \text{sinRepetidos}(cs) \wedge ((\forall k : ks) k \in \text{dameConts}(t)) \wedge \\
((\forall c : cs) c \leq \#\text{casilleros}(t))
\end{array} \right\}
$$

agregarFicha : jugador j × casillero c × partida p ⟶ partida
$\left\{\begin{array}{l} j \leq \#\text{jugadores}(p) \wedge_L \text{estáActivo?}(j,p) \wedge \neg\text{terminada?}(p) \wedge c \leq \#\text{casilleros}(\text{tablero}(p)) \wedge_L \\ ((\forall j' : nat)(1 \leq j' \leq \#\text{jugadores}(p) \wedge j' \neq j) \Rightarrow_L \text{fichasEnCasillero}(j,c,p) = 0) \end{array}\right\}$

mover : jugador j × movimiento m × nat n × partida p ⟶ partida
$\{j \leq \#\text{jugadores}(p) \wedge \text{estáActivo?}(j,p) \wedge \neg\text{terminada?}(p)\}$

otras operaciones

estáActivo? : jugador j × partida p ⟶ bool $\{j \leq \#\text{jugadores}(p)\}$

jugadoresActivos : partida ⟶ conj(jugador)

jugadoresEliminados : partida ⟶ conj(jugador)

tieneFichasEnAlgúnCasillero? : jugador j × nat n × partida p ⟶ bool $\{j \leq \#\text{jugadores}(p)\}$

jugadoresActivos : partida × nat ⟶ conj(jugador)

terminada? : partida ⟶ bool

algunoCompletóLaMisión? : partida × nat ⟶ bool

axiomas ∀ p: partida

tablero(crearPartida(t, js, cs, ks)) ≡ t

tablero(agregarFicha(j, c, p)) ≡ tablero(p)

tablero(mover(j, m, n, p)) ≡ tablero(p)

#jugadores(crearPartida(t, js, cs, ks)) ≡ js

#jugadores(agregarFicha(j, c, p)) ≡ #jugadores(p)

#jugadores(mover(j, m, n, p)) ≡ #jugadores(p)

misión(j,crearPartida(t, js, cs, ks)) ≡ $ks[j-1]$

misión(j,agregarFicha(j, c, p)) ≡ misión(j, p)

misión(j,mover(j, m, n, p)) ≡ misión(j, p)

estáActivo?(j) ≡ tieneFichasEnAlgúnCasillero(j,#casilleros(tablero(p)),p)

tieneFichasEnAlgúnCasillero(j, n, p) ≡ **if** $n = 0$ **then**
  false
else
  $0 <$ fichasEnCasillero(j, n, p)∨
  tieneFichasEnAlgúnCasillero($j, n-1, p$)
fi

jugadoresActivos(p) ≡ losActivos(p,#jugadores(p))

losActivos(p, n) ≡ **if** $n = 0$ **then**
  ∅
else
  if estáActivo?(n, p) **then** Ag(n,losActivos($p, n-1$)) **else** losActivos($p, n-1$) **fi**
fi

jugadoresEliminados(p) ≡ losEliminados(p,#jugadores(p))

losEliminados(p, n) ≡ **if** $n = 0$ **then**
  ∅
else
  if ¬ estáActivo?(n, p) **then**
    Ag(n,losEliminados($p, n-1$))
  else
    losEliminados($p, n-1$)
  fi
fi

//culo

terminada?(p) $\equiv$ #(jugadoresActivos(p))= 1 $\vee$ algunoCompletóLaMisión?(p,#jugadores(p))

Fin TAD

3. TAD Tupla(α_1, ..., α_n)

TAD Tupla(α_1, ..., α_n)

igualdad observacional

$(\forall t, t' : \text{tupla}(\alpha_1, \ldots, \alpha_n))\ \ (t =_{\text{obs}} t' \iff (\Pi_1(t) =_{\text{obs}} \Pi_1(t') \wedge \ldots \wedge \Pi_n(t) =_{\text{obs}} \Pi_n(t')))$

parámetros formales

géneros α_1, ..., α_n

géneros tupla($\alpha_1, \ldots, \alpha_n$)

exporta tupla, generadores, observadores

observadores básicos

Π_1 : tupla($\alpha_1, \ldots, \alpha_n$) $\longrightarrow$ α_1

$\vdots$

Π_n : tupla($\alpha_1, \ldots, \alpha_n$) $\longrightarrow$ α_n

generadores

$\langle \bullet, \ldots, \bullet \rangle$: $\alpha_1 \times \ldots \times \alpha_n$ $\longrightarrow$ tupla($\alpha_1, \ldots, \alpha_n$)

axiomas $\forall\ a_1$: α_1 ... $\forall\ a_n$: α_n

$\Pi_1(\langle a_1, \ldots, a_n \rangle) \equiv a_1$

$\vdots \equiv \vdots$

$\Pi_n(\langle a_1, \ldots, a_n \rangle) \equiv a_n$

Fin TAD

4. TAD Secuencia(α)

TAD Secuencia(α)

igualdad observacional

$$(\forall s, s' : \text{secu}(\alpha))\ \left(s =_{\text{obs}} s' \iff \begin{pmatrix} \text{vacía?}(s) =_{\text{obs}} \text{vacía?}(s') \wedge_{\text{L}} \\ (\neg\ \text{vacía?}(s) \Rightarrow_{\text{L}} (\text{prim}(s) =_{\text{obs}} \text{prim}(s') \wedge \text{fin}(s) =_{\text{obs}} \\ \text{fin}(s')) \end{pmatrix}\right)$$

parámetros formales

géneros α

géneros secu(α)

exporta secu(α), generadores, observadores, &, $\circ$, ult, com, long, está?

usa Bool, Nat

observadores básicos

vacía? : secu(α) $\longrightarrow$ bool

prim	: secu(α) s	$\longrightarrow$	α	$\{\neg$ vacía?(s)$\}$
fin	: secu(α) s	$\longrightarrow$	secu(α)	$\{\neg$ vacía?(s)$\}$

generadores

<>	:	$\longrightarrow$	secu(α)
$\bullet \bullet \bullet$	: $\alpha \times$ secu(α)	$\longrightarrow$	secu(α)

otras operaciones

$\bullet \circ \bullet$	: secu(α) $\times\ \alpha$	$\longrightarrow$	secu(α)	
$\bullet\ \&\ \bullet$	: secu(α) $\times$ secu(α)	$\longrightarrow$	secu(α)	
ult	: secu(α) s	$\longrightarrow$	α	$\{\neg$ vacía?(s)$\}$
com	: secu(α) s	$\longrightarrow$	secu(α)	$\{\neg$ vacía?(s)$\}$
long	: secu(α)	$\longrightarrow$	nat	
está?	: $\alpha \times$ secu(α)	$\longrightarrow$	bool	

axiomas $\forall\ s, t$: secu(α), $\forall\ e$: α

vacía?(<>)	$\equiv$	true
vacía?(e $\bullet$ s)	$\equiv$	

false

prim(e $\bullet$ s)	$\equiv$	e
fin(e $\bullet$ s)	$\equiv$	s
$s \circ e$	$\equiv$	**if** vacía?(s) **then** e $\bullet$ <> **else** prim(s) $\bullet$ (fin(s) $\circ$ e) **fi**
$s\ \&\ t$	$\equiv$	**if** vacía?(s) **then** t **else** prim(s) $\bullet$ (fin(s) & t) **fi**
ult(s)	$\equiv$	**if** vacía?(fin(s)) **then** prim(s) **else** ult(fin(s)) **fi**
com(s)	$\equiv$	**if** vacía?(fin(s)) **then** <> **else** prim(s) $\bullet$ com(fin(s)) **fi**
long(s)	$\equiv$	**if** vacía?(s) **then** 0 **else** 1 + long(fin(s)) **fi**
está?(e, s)	$\equiv$	$\neg$ vacía?(s) $\wedge_{\text{L}}$ (e = prim(s) $\vee$ está?(e, fin(s)))

Fin TAD

5. TAD Conjunto(α)

TAD Conjunto(α)

igualdad observacional

$$(\forall c, c' : \text{conj}(\alpha))\ \ (c =_{\text{obs}} c' \Longleftrightarrow ((\forall a : \alpha)(a \in c =_{\text{obs}} a \in c')))$$

parámetros formales

géneros α

géneros conj(α)

exporta conj(α), generadores, observadores, $\emptyset$?, $\cup$, $\cap$, #, $\bullet - \{\bullet\}$, dameUno, sinUno, $\subseteq$, $\bullet - \bullet$

usa Bool, Nat

observadores básicos

$\bullet \in \bullet$	: $\alpha \times$ conj(α)	$\longrightarrow$	bool

generadores

$\emptyset$	:		$\longrightarrow$ conj(α)
Ag	:	$\alpha \times$ conj(α)	$\longrightarrow$ conj(α)

otras operaciones

$\emptyset$?	:	conj(α)	$\longrightarrow$ bool	
$\#$	:	conj(α)	$\longrightarrow$ nat	
$\bullet - \{\bullet\}$	:	conj(α) $\times\ \alpha$	$\longrightarrow$ conj(α)	
$\bullet \cup \bullet$	:	conj(α) $\times$ conj(α)	$\longrightarrow$ conj(α)	
$\bullet \cap \bullet$	:	conj(α) $\times$ conj(α)	$\longrightarrow$ conj(α)	
dameUno	:	conj(α) c	$\longrightarrow \alpha$	$\{\neg\emptyset?(c)\}$
sinUno	:	conj(α) c	$\longrightarrow$ conj(α)	$\{\neg\emptyset?(c)\}$
$\bullet \subseteq \bullet$	:	conj(α) $\times$ conj(α)	$\longrightarrow$ bool	
$\bullet - \bullet$	:	conj(α) $\times$ conj(α)	$\longrightarrow$ conj(α)	

axiomas $\forall\ c, d$: conj(α), $\forall\ a, b$: α

$a \in \emptyset$	$\equiv$ false
$a \in \mathrm{Ag}(b,\ c)$	$\equiv (a = b) \vee (a \in c)$
$\emptyset?(\emptyset)$	$\equiv$ true
$\emptyset?(\mathrm{Ag}(b,\ c))$	$\equiv$ false
$\#(\emptyset)$	$\equiv 0$
$\#(\mathrm{Ag}(a,\ c))$	$\equiv 1 + \#(c$ - $\{\ a\ \}\ )$
$c - \{a\}$	$\equiv c - \mathrm{Ag}(a,\ \emptyset)$
$\emptyset \cup c$	$\equiv c$
$\mathrm{Ag}(a,\ c) \cup d$	$\equiv \mathrm{Ag}(a,\ c \cup d)$
$\emptyset \cap c$	$\equiv \emptyset$
$\mathrm{Ag}(a,\ c) \cap d$	$\equiv$ **if** $a \in d$ **then** $\mathrm{Ag}(a,\ c \cap d)$ **else** $c \cap d$ **fi**
$\mathrm{dameUno}(c) \in c$	$\equiv$ true
$\mathrm{sinUno}(c)$	$\equiv c - \{\mathrm{dameUno}(c)\}$
$c \subseteq d$	$\equiv c \cap d = c$
$\emptyset - c$	$\equiv \emptyset$
$\mathrm{Ag}(a,\ c) - d$	$\equiv$ **if** $a \in d$ **then** $c - d$ **else** $\mathrm{Ag}(a,\ c - d)$ **fi**

Fin TAD

6. TAD Multiconjunto(α)

TAD Multiconjunto(α)

igualdad observacional

$$(\forall c, c' : \mathrm{multiconj}(\alpha))\ \ (c =_{\mathrm{obs}} c' \iff ((\forall a : \alpha)(\#(a, c) =_{\mathrm{obs}} \#(a, c'))))$$

parámetros formales

géneros α

géneros multiconj(α)

exporta multiconj(α), generadores, observadores, $\in$, $\emptyset$?, #, $\cup$, $\cap$, $\in$, $\bullet - \{\bullet\}$, dameUno, sinUno

usa Bool, Nat

observadores básicos

$\#$: $\alpha \times$ multiconj(α) $\longrightarrow$ nat

generadores

$\emptyset$: $\longrightarrow$ multiconj(α)

Ag : $\alpha \times$ multiconj(α) $\longrightarrow$ multiconj(α)

otras operaciones

$\bullet \in \bullet$: $\alpha \times$ multiconj(α) $\longrightarrow$ bool

$\emptyset$? : multiconj(α) $\longrightarrow$ bool

$\#$: multiconj(α) $\longrightarrow$ nat

$\bullet - \{\bullet\}$: multiconj(α) $\times\ \alpha$ $\longrightarrow$ multiconj(α)

$\bullet \cup \bullet$: multiconj(α) $\times$ multiconj(α) $\longrightarrow$ multiconj(α)

$\bullet \cap \bullet$: multiconj(α) $\times$ multiconj(α) $\longrightarrow$ multiconj(α)

dameUno : multiconj(α) c $\longrightarrow$ α $\{\neg\emptyset?(c)\}$

sinUno : multiconj(α) c $\longrightarrow$ multiconj(α) $\{\neg\emptyset?(c)\}$

axiomas $\forall\ c, d$: multiconj(α), $\forall\ a, b$: α

$\#(a, \emptyset) \equiv 0$

$\#(a, \text{Ag}(b, c)) \equiv$ **if** $a = b$ **then** 1 **else** 0 **fi** $+ \#(a, c)$

$a \in c \equiv \#(a, c) > 0$

$\emptyset?(\emptyset) \equiv$ true

$\emptyset?(\text{Ag}(a, c)) \equiv$ false

$\#(\emptyset) \equiv 0$

$\#(\text{Ag}(a, c)) \equiv 1 + \#(c)$

$\emptyset - \{a\} \equiv \emptyset$

$\text{Ag}(a, c) - \{b\} \equiv$ **if** $a = b$ **then** c **else** $\text{Ag}(a, c - \{b\})$ **fi**

$\emptyset \cup c \equiv c$

$\text{Ag}(a, c) \cup d \equiv \text{Ag}(a, c \cup d)$

$\emptyset \cap c \equiv \emptyset$

$\text{Ag}(a, c) \cap d \equiv$ **if** $a \in d$ **then** $\text{Ag}(a, c \cap (d - \{a\}))$ **else** $c \cap d$ **fi**

$\text{dameUno}(c) \in c \equiv$ true

$\text{sinUno}(c) \equiv c - \{\text{dameUno}(c)\}$

Fin TAD

7. TAD Arreglo dimensionable(α)

TAD Arreglo dimensionable(α)

igualdad observacional

$$(\forall a, a' : \mathrm{ad}(\alpha)) \left(a =_{\mathrm{obs}} a' \iff \begin{pmatrix} \mathrm{tam}(a) =_{\mathrm{obs}} \mathrm{tam}(a') \wedge \\ (\forall n : \mathrm{nat})(\mathrm{definido?}(a, n) =_{\mathrm{obs}} \mathrm{definido?}(a', n) \wedge \\ (\mathrm{definido?}(a, n) \Rightarrow a[n] =_{\mathrm{obs}} a'[n])) \end{pmatrix} \right)$$

parámetros formales

géneros α

géneros ad(α)

exporta ad(α), generadores, observadores

usa Bool, Nat

observadores básicos

tam : ad(α) $\longrightarrow$ nat

definido? : ad(α) $\times$ nat $\longrightarrow$ bool

$\bullet\,[\,\bullet\,]$: ad(α) a $\times$ nat n $\longrightarrow$ α {definido?(a, n)}

generadores

crearArreglo : nat $\longrightarrow$ ad(α)

$\bullet\,[\,\bullet\,] \leftarrow \bullet$: ad(α) a $\times$ nat n $\times$ α $\longrightarrow$ ad(α) {$n <$ tam(a)}

axiomas $\forall$ a: ad(α), $\forall$ e: α, $\forall$ n, m: nat

tam(crearArreglo(n)) $\equiv$ n

tam(a [n] $\leftarrow$ e) $\equiv$ tam(a)

definido(crearArreglo(n), m)) $\equiv$ false

definido(a [n] $\leftarrow$ e, m) $\equiv$ $n = m$ $\vee$ definido?(a, m)

(a [n] $\leftarrow$ e) [m] $\equiv$ **if** $n = m$ **then** e **else** a [m] **fi**

Fin TAD

8. TAD Pila(α)

TAD Pila(α)

igualdad observacional

$$(\forall p, p' : \mathrm{pila}(\alpha)) \left(p =_{\mathrm{obs}} p' \iff \begin{pmatrix} \mathrm{vacía?}(p) =_{\mathrm{obs}} \mathrm{vacía?}(p')) \wedge_{\mathrm{L}} (\neg\ \mathrm{vacía?}(p) \Rightarrow_{\mathrm{L}} \\ (\mathrm{tope}(p) =_{\mathrm{obs}} \mathrm{tope}(p') \wedge \mathrm{desapilar}(p) =_{\mathrm{obs}} \mathrm{desapilar}(p')) \end{pmatrix} \right)$$

parámetros formales

géneros α

géneros pila(α)

exporta pila(α), generadores, observadores, tamaño

usa Bool, Nat

observadores básicos

vacía?	: pila(α)	$\longrightarrow$	bool	
tope	: pila(α) p	$\longrightarrow$	α	$\{\neg$ vacía?(p)$\}$
desapilar	: pila(α) p	$\longrightarrow$	pila(α)	$\{\neg$ vacía?(p)$\}$

generadores

vacía	:	$\longrightarrow$	pila(α)
apilar	: $\alpha \times$ pila(α)	$\longrightarrow$	pila(α)

otras operaciones

tamaño	: pila(α)	$\longrightarrow$	nat

axiomas $\forall$ p: pila(α), $\forall$ e: α

vacía?(vacía)	$\equiv$	true
vacía?(apilar(e,p))	$\equiv$	false
tope(apilar(e,p))	$\equiv$	e
desapilar(apilar(e,p))	$\equiv$	p
tamaño(p)	$\equiv$	**if** vacía?(p) **then** 0 **else** 1 + tamaño(desapilar(p)) **fi**

Fin TAD

9. TAD Cola(α)

TAD Cola(α)

igualdad observacional

$$(\forall c, c' : \text{cola}(\alpha)) \left(c =_{\text{obs}} c' \iff \left(\begin{array}{l} \text{vacía?}(c) =_{\text{obs}} \text{vacía?}(c') \wedge_{\text{L}} \\ (\neg\text{vacía?}(c) \Rightarrow_{\text{L}} (\text{próximo}(c) =_{\text{obs}} \text{próximo}(c') \wedge \\ \qquad\qquad \text{desencolar}(c) =_{\text{obs}} \text{desencolar}(c')) \end{array} \right) \right)$$

parámetros formales
géneros α

géneros cola(α)

exporta cola(α), generadores, observadores, tamaño

usa Bool, Nat

observadores básicos

vacía?	: cola(α)	$\longrightarrow$	bool	
próximo	: cola(α) c	$\longrightarrow$	α	$\{\neg$ vacía?(c)$\}$
desencolar	: cola(α) c	$\longrightarrow$	cola(α)	$\{\neg$ vacía?(c)$\}$

generadores

vacía	:	$\longrightarrow$	cola(α)
encolar	: $\alpha \times$ cola(α)	$\longrightarrow$	cola(α)

otras operaciones

tamaño	: cola(α)	$\longrightarrow$	nat

axiomas $\forall$ c: cola(α), $\forall$ e: α

vacía?(vacía)	$\equiv$	true

vacía?(encolar(e,c)) $\equiv$ false
próximo(encolar(e,c)) $\equiv$ **if** vacia?(c) **then** e **else** próximo(c) **fi**
desencolar(encolar(e,c)) $\equiv$ **if** vacía?(c) **then** vacía **else** encolar(e, desencolar(c)) **fi**
tamaño(c) $\equiv$ **if** vacía?(c) **then** 0 **else** 1 + tamaño(desencolar(c)) **fi**

Fin TAD

10. TAD Árbol binario(α)

TAD Árbol binario(α)

igualdad observacional

$$(\forall a, a' : \text{ab}(\alpha)) \left(a =_{\text{obs}} a' \iff \begin{pmatrix} \text{nil?}(a) =_{\text{obs}} \text{nil?}(a') \wedge_{\text{L}} (\neg\, \text{nil?}(a) \Rightarrow_{\text{L}} (\text{raiz}(a) =_{\text{obs}} \text{raiz}(a') \\ \wedge\, \text{izq}(a) =_{\text{obs}} \text{izq}(a') \wedge \text{der}(a) =_{\text{obs}} \text{der}(a')) \end{pmatrix} \right)$$

parámetros formales
géneros α

géneros ab(α)

exporta ab(α), generadores, observadores, tamaño, altura, tamaño, inorder, preorder, postorder

usa Bool, Nat, Secuencia(α)

observadores básicos

nil? : ab(α) $\longrightarrow$ bool
raiz : ab(α) a $\longrightarrow$ α $\{\neg\, \text{nil?}(a)\}$
izq : ab(α) a $\longrightarrow$ ab(α) $\{\neg\, \text{nil?}(a)\}$
der : ab(α) a $\longrightarrow$ ab(α) $\{\neg\, \text{nil?}(a)\}$

generadores

nil : $\longrightarrow$ ab(α)
bin : ab(α) $\times$ α $\times$ ab(α) $\longrightarrow$ ab(α)

otras operaciones

altura : ab(α) $\longrightarrow$ nat
tamaño : ab(α) $\longrightarrow$ nat
inorder : ab(α) $\longrightarrow$ secu(α)
preorder : ab(α) $\longrightarrow$ secu(α)
postorder : ab(α) $\longrightarrow$ secu(α)

axiomas $\forall\, a, b$: ab(α), $\forall\, e$: α

nil?(nil) $\equiv$ true
nil?(bin(a,e,b)) $\equiv$ false
raiz(bin(a,e,b)) $\equiv$ e
izq(bin(a,e,b)) $\equiv$ a
der(bin(a,e,b)) $\equiv$ b
altura(a) $\equiv$ **if** nil?(a) **then** 0 **else** 1 + máx(altura(izq(a)), altura(der(a))) **fi**
tamaño(a) $\equiv$ **if** nil?(a) **then** 0 **else** 1 + tamaño(izq(a)) + tamaño(der(a)) **fi**

inorder(a) $\equiv$ **if** nil?(a) **then** <> **else** inorder(izq(a)) & (raiz(a) $\bullet$ inorder(der(a))) **fi**
preorder(a) $\equiv$ **if** nil?(a) **then** <> **else** (raiz(a) $\bullet$ preorder(izq(a))) & preorder(der(a)) **fi**
postorder(a) $\equiv$ **if** nil?(a) **then** <> **else** postorder(izq(a)) & (postorder(der(a)) $\circ$ raiz(a)) **fi**

Fin TAD

11. TAD Diccionario(clave, significado)

TAD Diccionario(clave, significado)

igualdad observacional

$$(\forall d, d' : \text{dicc}(\kappa, \sigma)) \left(d =_{\text{obs}} d' \iff \begin{pmatrix} (\forall c : \kappa)(\text{def?}(c, d) =_{\text{obs}} \text{def?}(c, d') \wedge_{\text{L}} \\ (\text{def?}(c, d) \Rightarrow_{\text{L}} \text{obtener}(c, d) =_{\text{obs}} \text{obtener}(c, d'))) \end{pmatrix} \right)$$

parámetros formales
géneros clave, significado

géneros dicc(clave, significado)

exporta dicc(clave, significado), generadores, observadores, borrar, claves

usa Bool, Nat, Conjunto(clave)

observadores básicos

def? : clave × dicc(clave, significado) ⟶ bool
obtener : clave c × dicc(clave, significado) d ⟶ significado {def?(c, d)}

generadores

vacío : ⟶ dicc(clave, significado)
definir : clave × significado × dicc(clave, significado) ⟶ dicc(clave, significado)

otras operaciones

borrar : clave c × dicc(clave, significado) d ⟶ dicc(clave, significado) {def?(c,d)}
claves : dicc(clave, significado) ⟶ conj(clave)

axiomas $\forall$ d: dicc(clave, significado), $\forall$ c, k: clave, $\forall$ s: significado

def?(c,vacío) $\equiv$ false
def?(c, definir(k, s, d) $\equiv$ $c = k \vee$ def?(c, d)
obtener(c, definir(k, s, d)) $\equiv$ **if** $c = k$ **then** s **else** obtener(c, d) **fi**
borrar(c, definir(k, s, d)) $\equiv$ **if** $c = k$ **then**
  if def?(c,d) **then** borrar(c,d) **else** d **fi**
else
  definir(k, s, borrar(c, d))
fi
claves(vacío) $\equiv$ $\emptyset$
claves(definir(c,s,d)) $\equiv$ Ag(c, claves(d))

Fin TAD

12. TAD Cola de prioridad(α)

TAD Cola de prioridad(α)

igualdad observacional

$$(\forall c, c' : \text{colaPrior}(\alpha)) \left(c =_{\text{obs}} c' \iff \left(\begin{array}{l} \text{vacía?}(c) =_{\text{obs}} \text{vacía?}(c') \wedge_{\text{L}} \\ (\neg\text{vacía?}(c) \Rightarrow_{\text{L}} (\text{próximo}(c) =_{\text{obs}} \text{próximo}(c') \wedge \\ \qquad \text{desencolar}(c) =_{\text{obs}} \text{desencolar}(c')) \end{array} \right) \right)$$

parámetros formales

géneros α

operaciones $\bullet < \bullet : \alpha \times \alpha \longrightarrow$ bool — Relación de orden total estricto[1]

géneros colaPrior(α)

exporta colaPrior(α), generadores, observadores

usa Bool

observadores básicos

vacía?	: colaPrior(α)	$\longrightarrow$ bool	
próximo	: colaPrior(α) c	$\longrightarrow$ α	$\{\neg$ vacía?(c)$\}$
desencolar	: colaPrior(α) c	$\longrightarrow$ colaPrior(α)	$\{\neg$ vacía?(c)$\}$

generadores

vacía	:	$\longrightarrow$ colaPrior(α)
encolar	: $\alpha \times$ colaPrior(α)	$\longrightarrow$ colaPrior(α)

axiomas $\forall$ c: colaPrior(α), $\forall$ e: α

vacía?(vacía)	$\equiv$ true
vacía?(encolar(e, c))	$\equiv$ false
próximo(encolar(e, c))	$\equiv$ **if** vacía?(c) $\vee_{\text{L}}$ proximo(c) $< e$ **then** e **else** próximo(c) **fi**
desencolar(encolar(e, c))	$\equiv$ **if** vacía?(c) $\vee_{\text{L}}$ proximo(c) $< e$ **then** c **else** encolar(e, desencolar(c)) **fi**

Fin TAD

[1] Una relación es un orden total estricto cuando se cumple:

Antirreflexividad: $\neg\ a < a$ para todo $a : \alpha$

Antisimetría: $(a < b \ \Rightarrow\ \neg\ b < a)$ para todo $a, b : \alpha$, $a \neq b$

Transitividad: $((a < b \wedge b < c) \ \Rightarrow\ a < c)$ para todo $a, b, c : \alpha$

Totalidad: $(a < b \vee b < a)$ para todo $a, b : \alpha$