DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1

Especificación de TADs

18 de abril de 2025 — Algoritmos y Estructuras de Datos

Grupo 423:59

Integrante	LU	Correo electrónico
Cestau, Nicolás	834/23	nicocestau@gmail.com
Ricci, Fabrizio Bruno	532/22	fabrizioricci819@gmail.com
Romero, Santiago	272/21	santiagooromero1234@gmail.com
Davila Bustamante, Jasson Aldayr	59/22	jason.davila001@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Tipos de datos creados:

Usuario ES $\mathbb{Z}$
Usuarios ES $seq\langle Usuario\rangle$
Transaccion ES struct<id : $\mathbb{Z}$, monto : $\mathbb{Z}$, comprador : Usuario, vendedor : Usuario>
Bloque ES struct<id : $\mathbb{Z}$, transacciones : $seq\langle Transaccion\rangle$>
CadenaBloques ES $seq\langle Bloque\rangle$

`TAD Berretacoin` {

obs cb : CadenaBloques

`proc crearCoin` () : Berretacoin {

`requiere` {true}

`asegura` {$res.cb =<>$}

}

`proc agregarBloque` (`inout` b : Berretacoin, `in` bl : Bloque) : {

`requiere` {$b = b_0$}

`requiere` {$bloqueValido(bl, b.cb)$}

`asegura` {$|b.cb| = |b_0.cb| + 1$}

`asegura` {$bloqueAgregado(bl, b.cb) \wedge_L$
$viejosBloquesMantenidos(b.cb, b_0.cb) \wedge_L$
$estaOrdenado(b.cb)$}

(NOTA 1: Predicados del Requiere y sus Auxiliares)

`pred bloqueValido` (bl : Bloque, cb : CadenaBloques) {

$idBloqueMayorEnUno(bl, cb) \wedge_L$
$(1 \leq |bl.transacciones| \leq 50) \wedge_L$
$transaccionDeCreacionValida(bl.transacciones[0], cb) \wedge_L$
$restoDeTransaccionesValidas(bl.transacciones, cb)$

}

`pred idBloqueMayorEnUno` (bl : Bloque, cb : CadenaBloques) {

$bl.id = (cb[|cb| - 1].id) + 1$

}

`pred transaccionDeCreacionValida` (t : Transaccion, cb : CadenaBloques) {

$(|cb| \leq 3000 \longrightarrow t.comprador = 0 \wedge t.vendedor > 0 \wedge$
$vendedoresDeCreacionDistintos(t, cb) \wedge t.monto = 1 \wedge t.id > 0)$

}

`pred vendedoresDeCreacionDistintos` (t : Transaccion, cb : CadenaBloques) {

$(\forall i : \mathbb{Z})$ (

$0 \leq i < |cb| \longrightarrow_L t.vendedor \neq cb.[i].transacciones[0].vendedor$

)

}

`pred restoDeTransaccionesValidas` (t : $seq\langle Transaccion\rangle$, cb : CadenaBloques) {

$(|cb| \leq 3000 \longrightarrow transaccionesConEmision(t, cb)) \wedge$
$(|cb| > 3000 \longrightarrow transaccionesSinEmision(t, cb))$

}

`pred transaccionesConEmision` (t : $seq\langle Transaccion\rangle$, cb : CadenaBloques) {

$(\forall i : \mathbb{Z})\ (1 \leq i < |t| \longrightarrow_L t[i].id > 0) \wedge_L (\forall i : \mathbb{Z})\ (1 \leq i < |t| \longrightarrow_L t[i].comprador \neq t[i].vendedor) \wedge_L$
$(\forall i : \mathbb{Z})\ (1 \leq i < |t| \longrightarrow_L montoValido(cb, t, t[i], i))$

}

pred transaccionesSinEmision (t : $seq\langle Transaccion\rangle$, cb : CadenaBloques) {

$(\forall i : \mathbb{Z})\ (0 \leq i<|t| \longrightarrow_L t[i].id>0) \wedge_L (\forall i : \mathbb{Z})\ (0 \leq i<|t| \longrightarrow_L t[i].comprador \neq t[i].vendedor) \wedge_L$
$(\forall i : \mathbb{Z})\ (0 \leq i<|t| \longrightarrow_L montoValido(cb, t, t[i], i))$

}

pred montoValido (cb : CadenaBloques, ts : $seq\langle Transaccion\rangle$, t : Transaccion, i : $\mathbb{Z}$) {

$patrimonio(ventasEnLaCadenaBloques(cb, t.comprador),$
$comprasEnLaCadenaBloques(cb, t.comprador), ventasEnElMismoBloque(ts, t.comprador, i),$
$comprasEnElMismoBloque(ts, t.comprador, i)) \geq t.monto$

}

aux ventasEnLaCadenaBloques (cb : CadenaBloques, id : Usuario) : $\mathbb{Z}$ =
$\sum_{i=0}^{|cb|}(\sum_{j=0}^{|cb[i]|}(IfThenElse(id = cb[i].transacciones[j].vendedor,cb[i].transacciones[j].monto,0)))$;

aux comprasEnLaCadenaBloques (cb : CadenaBloques, id : Usuario) : $\mathbb{Z}$ =
$\sum_{i=0}^{|cb|}(\sum_{j=0}^{|cb[i]|}(IfThenElse(id = cb[i].transacciones[j].comprador,cb[i].transacciones[j].monto,0)))$;

aux ventasEnElMismoBloque (t : $seq\langle Transaccion\rangle$, id : Usuario, i : $\mathbb{Z}$) : $\mathbb{Z}$ =
$\sum_{j=0}^{i-1}(IfThenElse(id = t[j].vendedor,t[j].monto,0))$;

aux comprasEnElMismoBloque (t : $seq\langle Transaccion\rangle$, id : Usuario, i : $\mathbb{Z}$) : $\mathbb{Z}$ =
$\sum_{j=0}^{i-1}(IfThenElse(id = t[j].comprador,t[j].monto,0))$;

aux patrimonio (VELCB: $\mathbb{Z}$, CELCB : $\mathbb{Z}$, VEEMB : $\mathbb{Z}$, CEEMB : $\mathbb{Z}$) : $\mathbb{Z}$ =
$(VEEMB + VELCB) - (CEEMB + CELCB)$;

(NOTA 2: Predicados del Asegura)

pred bloqueAgregado (bl : Bloque, cb : CadenaBloques) {

$bl \in cb$

}

pred viejosBloquesMantenidos (cb : CadenaBloques, cb_0 : CadenaBloques) {

$(\forall B : Bloque)$ (

$B \in cb\_0 \longrightarrow_L B \in cb$

)

}

pred estaOrdenado (cb : CadenaBloque) {

$(\forall i : \mathbb{Z})$ (

$0 \leq i<|cb| - 1 \longrightarrow_L cb[i].id<cb[i + 1].id$

)

}

}

proc maximosTenedores (in b : Berretacion) : $seq\langle Usuario\rangle$

requiere $\{|b.cb| > 0\}$

asegura $\{sonMaximosTenedores(b.cb, res)\}$

pred sonMaximosTenedores (cb : CadenaBloques, res : $seq\langle Usuario\rangle$) {

$(\forall u : Usuario)\ (u \in res \longrightarrow usuarioDeBloque(cb, u) \wedge$
$(\forall v : Usuario)\ (usuarioDeBloque(cb, v) \longrightarrow$
$(ventasEnLaCadenaBloques(cb, v) - comprasEnLaCadenaBloques(cb, v)) \leq$
$(ventasEnLaCadenaBloques(cb, u) - comprasEnLaCadenaBloques(cb, u))))$

}

pred usuarioDeBloque (cb : CadenaBloques, u : Usuario) {

$(\exists b : Bloque)\ (b \in cb \wedge (\exists t : Transaccion)\ (t \in b \wedge (u = t_2 \vee u = t_3)))$

}

proc montoMedio (in b : Berretacoin) : $\mathbb{R}$

requiere $\{(\exists u : bloque)\ (u \in b.cb \wedge |u.transacciones| \geq 2)\}$

asegura $\{res = \frac{(sumaMontoTotales(b.cb) - sumaMontoTransaccionesCreacion(b.cb))}{(cantidadTransaccionesTotales(b.cb) - cantidadTransaccionesCreacion(b.cb))}\}$

aux sumaMontoTotales (c : CadenaBloques) : $\mathbb{Z}$ =
$\sum_{i=0}^{|c|-1} (sumaMontoDeBloque[i])$;

aux SumaMontoDeBloque (b : Bloque) : $\mathbb{Z}$ =
$\sum_{i=0}^{|b.transacciones|-1} (b.transacciones[i]_1)$;

aux sumaMontoTransaccionesCreacion (c : CadenaBloques) : $\mathbb{Z}$ =
$\sum_{i=0}^{|c|-1} (IfThenElseFi(i < 3000, c[i].transacciones[0]_1, 0))$;

aux cantidadTransaccionesTotales (c : CadenaBloques) : $\mathbb{Z}$ =
$\sum_{i=0}^{|c|-1} (|c[i].transacciones|)$;

aux CantidadTransaccionesCreacion (c : CadenaBloques) : $\mathbb{Z}$ =
$IfThenElseFi(|c| < 3000, |c|, 3000)$;

(NOTA 3: Aquí en el procediminto de cotizacionAPesos reusamos el auxiliar sumaMontoDeBloque)
(NOTA 4: El -1 que resta al auxiliar sumaMontoDeBloque es por los montos de las transacciones de creación)

proc cotizacionAPesos (in b : Berretacion, in P : $seq\langle\mathbb{Z}\rangle$) : $seq\langle\mathbb{Z}\rangle$

requiere $\{|b.cb| = |P|\}$

asegura $\{|res| = |P|\}$

asegura $\{((\forall i : \mathbb{Z})\ (0 \leq i < |P| \longrightarrow_L res[i] = (sumaMontoDeBloque(b.cb[i]) - 1) * P[i]))\}$

}