



# Trabajo práctico 1

## Especificación de TADs

19 de abril de 2025

Algoritmos y Estructuras de Datos

### Grupo 23:59

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  $struct\langle id : \mathbb{Z}, monto : \mathbb{Z}, comprador : Usuario, vendedor : Usuario \rangle$

Bloque ES  $struct\langle id : \mathbb{Z}, transacciones : seq\langle Transaccion \rangle \rangle$

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<sub>0</sub>}

**requiere** {bloqueValido(bl, b.cb)}

**asegura** {|b.cb| = |b<sub>0</sub>.cb| + 1}

**asegura** {bloqueAgregado(bl, b.cb)  $\wedge_L$   
viejosBloquesMantenidos(b.cb, b<sub>0</sub>.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 ≤ |bl.transacciones| ≤ 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| ≤ 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) {

    (∀i :  $\mathbb{Z}$ ) (

      0 ≤ i < |cb|  $\longrightarrow_L$  t.vendedor ≠ cb[i].transacciones[0].vendedor

    )

  }

**pred** restoDeTransaccionesValidas (t : seq⟨Transaccion⟩, cb : CadenaBloques) {

    (|cb| ≤ 3000  $\longrightarrow$  transaccionesConEmision(t, cb))  $\wedge$

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

  }

**pred** transaccionesConEmision (t : seq⟨Transaccion⟩, cb : CadenaBloques) {

    (∀i :  $\mathbb{Z}$ ) (1 ≤ i < |t|  $\longrightarrow_L$  t[i].id > 0)  $\wedge_L$  (∀i :  $\mathbb{Z}$ ) (1 ≤ i < |t|  $\longrightarrow_L$  t[i].comprador ≠ t[i].vendedor)  $\wedge_L$

    (∀i :  $\mathbb{Z}$ ) (1 ≤ i < |t|  $\longrightarrow_L$  montoValido(cb, t, t[i], i))

  }

```

pred transaccionesSinEmision (t : seq⟨Transaccion⟩, cb : CadenaBloques) {
  (∀i : ℤ) (0 ≤ i < |t| ⟶L t[i].id > 0) ∧L (∀i : ℤ) (0 ≤ i < |t| ⟶L t[i].comprador ≠ t[i].vendedor) ∧L
  (∀i : ℤ) (0 ≤ i < |t| ⟶L montoValido(cb, t, t[i], i))
}

pred montoValido (cb : CadenaBloques, ts : seq⟨Transaccion⟩, t : Transaccion, i : ℤ) {
  patrimonio(ventasEnLaCadenaBloques(cb, t.comprador),
  comprasEnLaCadenaBloques(cb, t.comprador), ventasEnElMismoBloque(ts, t.comprador, i),
  comprasEnElMismoBloque(ts, t.comprador, i)) ≥ t.monto
}

aux ventasEnLaCadenaBloques (cb : CadenaBloques, id : Usuario) : ℤ =
  ∑i=0|cb| ( ∑j=0|cb[i]| (IfThenElse(id = cb[i].transacciones[j].vendedor, cb[i].transacciones[j].monto, 0))) ;

aux comprasEnLaCadenaBloques (cb : CadenaBloques, id : Usuario) : ℤ =
  ∑i=0|cb| ( ∑j=0|cb[i]| (IfThenElse(id = cb[i].transacciones[j].comprador, cb[i].transacciones[j].monto, 0))) ;

aux ventasEnElMismoBloque (t : seq⟨Transaccion⟩, id : Usuario, i : ℤ) : ℤ =
  ∑j=0i-1 (IfThenElse(id = t[j].vendedor, t[j].monto, 0)) ;

aux comprasEnElMismoBloque (t : seq⟨Transaccion⟩, id : Usuario, i : ℤ) : ℤ =
  ∑j=0i-1 (IfThenElse(id = t[j].comprador, t[j].monto, 0)) ;

aux patrimonio (VELCB : ℤ, CELCB : ℤ, VEEMB : ℤ, CEEMB : ℤ) : ℤ =
  (VEEMB + VELCB) - (CEEMB + CELCB) ;
(NOTA 2: Predicados del Asegura )

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

pred viejosBloquesMantenidos (cb : CadenaBloques, cb_0 : CadenaBloques) {
  (∀B : Bloque) (
    B ∈ cb_0 ⟶L B ∈ cb
  )
}

pred estaOrdenado (cb : CadenaBloques) {
  (∀i : ℤ) (
    0 ≤ i < |cb| - 1 ⟶L cb[i].id < cb[i + 1].id
  )
}

}

proc maximosTenedores (in b : Berretacion) : Usuarios
  requiere {|b.cb| > 0}
  asegura {sonMaximosTenedores(b.cb, res)}
  pred sonMaximosTenedores (cb : CadenaBloques, res : Usuarios) {
    (∀u : Usuario) (u ∈ res ⟶ usuarioDeCoin(cb, u) ∧
    (∀v : Usuario) (usuarioDeCoin(cb, v) ⟶
    (ventasEnLaCadenaBloques(cb, v) - comprasEnLaCadenaBloques(cb, v)) ≤
    (ventasEnLaCadenaBloques(cb, u) - comprasEnLaCadenaBloques(cb, u))))
  }

  pred usuarioDeCoin (cb : CadenaBloques, u : Usuario) {
    (∃b : Bloque) (b ∈ cb ∧ (∃t : Transaccion) (t ∈ b ∧ (u = t2 ∨ u = t3)))
  }
}

```

```

proc montoMedio (in b : Berretacoin) :  $\mathbb{R}$ 
  requiere  $\{(\exists u : \text{bloque}) (u \in b.cb \wedge |u.transacciones| \geq 2)\}$ 
  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) :  $\mathbb{Z}$  =
     $\sum_{i=0}^{|b.transacciones|-1} (b.transacciones[i].monto);$ 
  aux sumaMontoTransaccionesCreacion (c : CadenaBloques) :  $\mathbb{Z}$  =
     $\sum_{i=0}^{|c|-1} (IfThenElseFi(i < 3000, c[i].transacciones[0].monto, 0));$ 
  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 )

(NOTA 5: P representa las cotizaciones)

```

proc cotizacionAPesos (in b : Berretacion, in P : seq< $\mathbb{Z}$ >) : seq< $\mathbb{Z}$ >
  requiere  $\{|b.cb| = |P|\}$ 
  asegura  $\{|res| = |P|\}$ 
  asegura  $\{((\forall i : \mathbb{Z}) (0 \leq i < |P| \longrightarrow_L res[i] = (sumaMontoDeBloque(b.cb[i]) - 1) * P[i]))\}$ 
}

```