

Grupo: Ameri

Integrante	LU	Correo electrónico
Rankov, Jorge	714/23	jrankov@dc.uba.ar
Falbo, Tiziana	nnn/nn	nnn@gmail.com
Facundo	nnn/nn	nnn@gmail.com
Bautista	nnn/nn	nnn@gmail.com



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

## TAD \$BerretaCoin {

**obs blockchain:** Seq<bloques>

```
proc nuevoBerretaCoin(): BerretaCoin {  
  asegura { res.blockchain = <> }  
}
```

```
proc agregarBloque (inout B: BerretaCoin, in S: bloque) {  
  requiere {  $B = B_0 \wedge (0 < |S.transacciones| \leq 50) \wedge \text{bloqueValido}(B, S) \wedge \text{sonTransaccionesValidas}(S, B)$  }  
  
  asegura { B.blockchain = B0.blockchain ++ {S} }  
}
```

```
proc maximosTenedores (in B: BerretaCoin): Seq<Z> {  
  asegura {  $(\forall i: \mathbb{Z})(0 \leq i < |res|) \rightarrow \text{esUsuario}(res_{[i]}, B)$  }  
  
  asegura {  $(\forall id: \mathbb{Z})(id \in res) \rightarrow \neg (\exists \text{otro: } \mathbb{Z})(\text{esUsuario}(\text{otro}, B) \wedge$   
     $(\text{montoDeUsuario}(\text{otro}, B) \geq \text{montoDeUsuario}(id, B)))$  }  
  
  asegura {  $(\forall i: \mathbb{Z})(\forall j: \mathbb{Z})(0 \leq i < |res|) \wedge (0 \leq j < |res|) \wedge (i \neq j) \rightarrow res_{[i]} \neq res_{[j]}$  }  
}
```

```
proc montoMedio (in B: BerretaCoin): Float {  
  requiere {  $|B.blockchain| \neq 0$  }  
  
  asegura { res = promedio (B.blockchain) }  
}
```

```
proc cotizacionAPesos (in cotizaciones: Seq<Z>, in B: BerretaCoin): Seq<Z> {  
  requiere {  $(\forall C \in \text{cotizaciones})(C > 0)$  }  
  
  requiere {  $|\text{cotizaciones}| = |B.blockchain|$  }  
  
  asegura {  $|res| = |\text{cotizaciones}|$  }  
  
  asegura {  $(\forall i: \mathbb{Z})(0 \leq i < |res|) \rightarrow res_{[i]} = \text{cotizarBloque}(i, \text{cotizaciones}, B.blockchain)$  }  
}
```

# Predicados

```
pred bloqueValido (B: BerretaCoin, S: bloque) {  
  ( $|B.blockchain| < 3000$ )  $\rightarrow S.transacciones_{[0]}.idComprador = 0 \wedge$   
  ( $|B.blockchain| \geq 3000$ )  $\rightarrow S.transacciones_{[0]}.idComprador \neq 0 \wedge$   
  ( $|S.transacciones| \leq 50$ )  
   $\wedge$   
  [ $(\forall transaccion \in S.transacciones) \rightarrow [(transaccion.idComprador \neq transaccion.idVendedor) \wedge$   
    ( $transaccion.id > 0 \wedge transaccion.idComprador \geq 0 \wedge transaccion.idVendedor > 0 \wedge transaccion.monto > 0$ )]  $\wedge$   
    estaOrdenada(S)  $\wedge$  esIdDeBloqueConsecutivo(B, S)]  
}
```

```
pred estaOrdenada (S: bloque) {  
  ( $\forall i: \mathbb{Z}$ ) ( $0 \leq i < |S.transacciones|$ )  $\rightarrow (S.transacciones_{[i]}.idTransaccion = i)$   
}
```

```
pred esIdDeBloqueConsecutivo (B: BerretaCoin, S: bloque) {  
  ( $|B.blockchain| = 0$ )  $\rightarrow (S.idBloque = 0) \wedge$   
  ( $|B.blockchain| \neq 0$ )  $\rightarrow (S.idBloque = B.blockchain[|B.blockchain|-1].idBloque + 1)$   
}
```

```
pred sonTransaccionesValidas (S: bloque, B: blockchain) {  
  ( $\forall id: \mathbb{Z}$ ) ( $\forall j: \mathbb{Z}$ ) ( $0 \leq j < |S.transacciones|$ ) [ $esUsuario(id, B) \vee esUsuarioDeBloque(id, S)$ ]  
     $\rightarrow montoDeUsuarioHastaTransaccion(id, S, B, j) \geq 0$   
}
```

```
pred esUsuario (id:  $\mathbb{Z}$ , B: BerretaCoin) {  
  ( $\exists i: \mathbb{Z}$ ) ( $\exists j: \mathbb{Z}$ ) ( $0 \leq i < |b.blockchain| \wedge 0 \leq j < |b.blockchain_{[i]}.transacciones|$ )  $\wedge$   
  ( $id = b.blockchain_{[i]}.transacciones_{[j]}.idVendedor$ )  
}
```

## Auxiliares

**aux MontoDeUsuario** (id:  $\mathbb{Z}$ ; B: BerretaCoin):  $\mathbb{Z}$ =

$$\sum_{j=0}^{|bloques|-1} [ \sum_{i=0}^{|bloques[j].transacciones|-1} (ifThenElse(bloques[j].transacciones[i].idComprador = id, \\ bloques[j].transacciones[i].monto, 0)) + \\ \sum_{i=0}^{|bloques[j].transacciones|-1} (ifThenElse(bloques[j].transacciones[i].idVendedor = id, \\ bloques[j].transacciones[i].monto, 0)) ]$$

**aux montoDeUsuarioHastaTransaccion** (id:  $\mathbb{Z}$ , S: bloque, B: BerretaCoin, pos:  $\mathbb{Z}$ ) =

$$\text{montoDeUsuario}(\text{id}, \text{B}) + \\ [ \sum_{i=0}^{pos} (ifThenElse(S.transacciones[i].idComprador = id, bloques[j].transacciones[i].monto, 0)) \\ + \\ \sum_{i=0}^{pos} (ifThenElse(S.transacciones[i].idVendedor = id, bloques[j].transacciones[i].monto, 0)) ]$$

**aux promedio** (bloques: seq<bloques>): Float =

$$\frac{\sum_{j=0}^{|bloques|-1} ( \sum_{i=0}^{|bloques[j].transacciones|-1} (ifThenElse(bloques[j].transacciones[i].idComprador \neq 0, \\ bloques[j].transacciones[i].monto, 0)) )}{\sum_{j=0}^{|bloques|-1} \left( \sum_{i=0}^{|bloques[j].transacciones|-1} (ifThenElse(bloques[j].transacciones[i].idComprador \neq 0, 1, 0)) \right)}$$

**aux cotizarBloque** (posicion:  $\mathbb{Z}$ , cotizaciones: Seq< $\mathbb{Z}$ >, blockchain: Seq<bloques>):  $\mathbb{Z}$ =

$$\left( \sum_{i=0}^{|blockchain[posicion]|-1} blockchain[posicion].transacciones[i].montos \right) * cotizaciones[posicion]$$

## Anotaciones y Cosas a reubicar después

cadenaDeBloques = Seq<Seq< $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ >>

bloque = struct(transacciones: seq<struct (idTransaccion: $\mathbb{Z}$ , idComprador: $\mathbb{Z}$ , idVendedor: $\mathbb{Z}$ , monto: $\mathbb{Z}$ )>, idBloque: $\mathbb{Z}$ )

### Procs

```
proc montosDeUsuarios {
  asegura {  $\forall id \in \text{sinRepetidos (Usuarios (Cripto.blockchain))} \rightarrow id \in \text{res}$ 
            $\longleftrightarrow (\text{esMaximo (MontoDeUsuario (Cripto.blockchain, id)); Montos (Usuarios (Cripto.blockchain))})$  }
}
```

```
proc Usuarios (in bc: BerretaCoin): Seq< $\mathbb{Z}$ > {
  asegura {  $(\forall i: \mathbb{Z}) (0 \leq i < |bc.blockchain|)$  }

  asegura {  $(\forall j: \mathbb{Z}) (0 \leq i < |bc.blockchain_{[i]}|)$  }

  asegura {  $(bc.blockchain_{[i][j][1]} \wedge bc.blockchain_{[i][j][2]}) \in res \wedge \text{ningunOtroElem} \in res$  }
}
```

### Preds

```
pred esTransaccionCreacion (t: Seq< $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ >) {
   $t_{[1]} = 0$ 
}
```

```
pred sinRepetirId (ids: Seq< $\mathbb{Z} \times \mathbb{Z}$ >) {
   $(\forall i, j: \mathbb{N}) ((0 \leq i < |ids| \wedge_L (0 \leq j < |ids| \wedge_L (j \neq i)) \rightarrow_L id_{[i][0]} \neq id_{[j][0]}))$ 
}
```

```
pred esMaximo (Monto:  $\mathbb{Z}$ , Montos: Seq< $\mathbb{Z}$ >) {
   $(\forall i \in \text{Montos}) \rightarrow_L \text{Monto} \geq i$ 
}
```

```
pred esTransaccionValida (b: Seq< $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ >, ids: Seq< $\mathbb{Z} \times \mathbb{Z}$ >) {
   $(\forall i, j: \mathbb{N}) (j \leq i < |b| \wedge_L (0 \leq j < |ids|) \wedge_L (b_{[i][2]} = ids_{[j][0]}) \rightarrow_L (b_{[i][3]} \leq ids_{[j][1]})$ 
}
```

### Auxs

aux Montos (S: Seq< $\mathbb{Z}$ >): Seq< $\mathbb{Z}$ >=

$$\sum_{j=0}^{|s|-1} (\text{MontoDeUsuario}(S_{[i]}))$$

aux sinRepetidos (S: Seq< $\mathbb{Z}$ >): Seq< $\mathbb{Z}$ >=

$$[S_{[0]}] + \sum_{i=1}^{|s|-1} \text{ifThenElse}(S_{[i]} \in \text{SubSeq}(S, 0, i-1); \emptyset; [S_{[i]}])$$

aux Usuarios (S:Seq<Seq< $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ >>): Seq< $\mathbb{Z}$ >=

$$\sum_{i=0}^{|s|-i} \sum_{j=0}^{|s_{[i]}|-1} (S_{[i][j][1]}, S_{[i][j][2]})$$