DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1 - Especificación de TADs

Criptomonedas

Algoritmos y Estructura de Datos 2

Grupo Pythonisbetter

Integrante	LU	Correo electrónico
Perez Marzo, Jordan Alexis	738/24	jordanpm30@gmail.com
Nuñez Geronimo, Sebastian Javier	986/24	sebustos2394@gmail.com
Camacho Gomez, Lucero Belen	667/23	camacholu14@gmail.com
Suarez, Ricardo Javier	127/20	ricardojaviersuarezz@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

1. Definición de Tipos

type *Bloque* ES $tupla\langle \mathbb{Z}, seq\langle Transaccion\rangle\rangle$
type *idBloque* ES $\mathbb{Z}$
type *Transaccion* ES $tupla\langle idTransaccion, idComprador, idVendedor, monto\rangle$
type *idTransaccion, idComprador, idVendedor, monto* ES $\mathbb{Z}$
type *Usuario, Saldo* ES $\mathbb{Z}$
type *Cotizacion* ES $\mathbb{Z}$

2. Especificación

TAD $BerretaCoin {

obs $bloques = seq\langle Bloque\rangle$
obs $dineroUsuariosTotal = dict\langle Usuario, Saldo\rangle$

2.1. Ejercicio 1

`proc $BerretaCoinVacio` () : $ Berretacoin {
`asegura` $\{res.bloque =<>\}$
`asegura` $\{res.dineroUsuarioTotal = \{\}\}$
}

`proc agregarBloque` (`inout` BCoin : $ Berretacoin , `in` b : Bloque) : {
`requiere` $\{BCoin = BCoin_0\}$
`requiere` $\{0 < |b_1| \leq 50\}$
`requiere` $\{|BCoin_0.bloques| \leq 2999 \longrightarrow (bloqueValidoConTransaccionEspecial(b) \wedge$
$vendedorDeCreacionSiempreDistinto(BCoin_0.bloques, b))\}$
`requiere` $\{|BCoin_0.bloques| > 2999 \longrightarrow bloqueValidoSinTransaccionEspecial(b)\}$
`requiere` $\{nadieGastaMasDeLoQueTiene(BCoin_0.dineroUsuarioTotal, b)\}$
`requiere` $\{esBloqueConsecutivoConLosDemasPorId(BCoin_0.bloques, b)\}$
`asegura` $\{BCoin.bloques = BCoin_0.bloques + + < b >\}$
`asegura` $\{(\forall j : Z)((esUsuarioDelBloque(j, b)) \longrightarrow_L$
$BCoin.dineroUsuariosTotal[j] =$ `IfThenElse`$(j \notin BCoin_0.dineroUsuarioTotal,$
$montoComprado(b_1, j) - montoVendido(b_1, j),$
$BCoin_0.dineroUsuarioTotal[j] + montoComprado(b_1, j) - montoVendido(b_1, j))\}$
`asegura` $\{(\forall j : Z)((j \in BCoin_0.dineroUsuarioTotal \wedge \neg esUsuarioDelBloque(j, b)) \longrightarrow_L$
$BCoin.dineroUsuariosTotal[j] = BCoin_0.dineroUsuarioTotal[j]))\}$
`asegura` $\{(\forall j : Z)((j \notin BCoin_0.dineroUsuarioTotal \wedge \neg esUsuarioDelBloque(j, b)) \longrightarrow_L$
$j \notin BCoin.dineroUsuariosTotal)\}$
}

`pred bloqueValidoConTransaccionEspecial` (b: Bloque) {
$(b_1[0]_1 = 0 \wedge b_1[0]_2 > 0 \wedge b_1[0]_3 = 1 \wedge b_1[0]_0 > 0) \wedge (\forall i : Z)(1 \leq i < |b_1| \longrightarrow_L$
$(idTransaccionYUsuariosYMontosValidos(b_1[i]))) \wedge tieneTransaccionesConsecutivas(b)$
}
`pred bloqueValidoSinTransaccionEspecial` (b: Bloque) {
$(\forall i : Z)(0 \leq i < |b_1| \longrightarrow_L (idTransaccionYUsuariosYMontosValidos(b_1[i]))) \wedge tieneTransaccionesConsecutivas(b)$
}
`pred idTransaccionYUsuariosYMontosValidos` (t: Transaccion) {
$t_1 > 0 \wedge_L t_2 > 0 \wedge_L t_1 \neq t_2 \wedge_L t_3 > 0 \wedge_L t_0 > 0)$
}
`pred tieneTransaccionesConsecutivas` (b: Bloque) {
$(\forall j : Z)(0 \leq j < |b_1| - 1) \longrightarrow_L b_1[j]_0 = b_1[j + 1]_0 - 1)$
}
`pred vendedorDeCreacionSiempreDistinto` (s:$seq\langle Bloque\rangle$,b: Bloque) {
$(\forall i : Z)(0 \leq i < |s| \longrightarrow_L BCoin_0.bloques[i]_1[0]_2 \neq b_1[0]_2)$
}
`pred nadieGastaMasDeLoQueTiene` (d: dict $\langle Usuario, Saldo\rangle, b : Bloque$){

$(\forall i : Z)(0 \leq i < |b_1| \longrightarrow_L$
$(\forall j : Z)(j \notin d \wedge_L [(montoComprado(subseq(b_1, 0, i+1), j) - (montoVendido(subseq(b_1, 0, i+1), j)))] \geq 0)$
$\wedge$
$(\forall n : Z)(0 \leq n < |b_1| \longrightarrow_L$
$(\forall m : Z)(m \in d \wedge_L d[m] + [(montoComprado(subseq(b_1, 0, n+1), m) - (montoVendido(subseq(b_1, 0, n+1), m)))] \geq 0)$
}

`pred esBloqueConsecutivoConLosDemasPorId` (s:$seq\langle Bloque\rangle$,b: Bloque) {

$|s| > 0 \wedge s[|s| - 1]_0 = b_0 - 1$

}

`aux montoComprado` (t:$seq\langle Transaccion\rangle$, u: Usuario) : Z $=$
$\sum_{i=0}^{|t|-1}$ `IfThenElse`$(t[i]_1 = u, t[i]_3, 0)$;

`aux montoVendido` (t:$seq\langle Transaccion\rangle$, u: Usuario) : Z $=$
$\sum_{i=0}^{|t|-1}$ `IfThenElse`$(t[i]_2 = u, t[i]_3, 0)$;

`pred esUsuarioDelBloque` (u: Usuario , b: Bloque) {

$(\exists i : Z)(0 \leq i < |b_n| \wedge_L (b_{1_{[i]_1}} = u \vee_L b_{1_{[i]_2}} = u))$

}

2.2. Ejercicio 2

`proc maximoTenedores` (in BCoin : $BerretaCoin) : $seq\langle\mathbb{Z}\rangle${

`requiere` $\{True\}$

`asegura` $\{(\forall i : Usuario)\ (0 \leq i < |res| \longrightarrow_L usuarioConMasBCoin(res[i], BCoin.dineroUsuarioTotal))\}$

`asegura` $\{noHayRepetidos(res)\}$

`asegura` $\{(\forall j : Usuario)(j \in Bcoin.dineroUsuarioTotal$
$\wedge_L usuarioConMasBCoin(j, BCoin.dinerousuarioTotal)) \longrightarrow_L j \in res\}$

}

`pred usuarioConMasBCoin` (u : Usuario, d : *dict* $\langle$Usuario, Saldo$\rangle$){

$(\forall j : Usuario)\big((j \in d \wedge u \in d \wedge j \neq u) \longrightarrow_L d[j] \leq d[u]\big)$

}

`pred noHayRepetidos` (s:$seq\langle T\rangle$) {

$(\forall i : Z)(0 \leq i < |s| \longrightarrow_L \neg(\exists j : Z)((0 \leq j < |s| \wedge j \neq i) \wedge_L s[i] = s[j])$

}

2.3. Ejercicio 3

`proc montoMedio` (in BCoin : $ BerretaCoin) : R {

`asegura` $\{|BCoin.bloques| = 0 \longrightarrow res = 0\}$

`asegura` $\{|BCoin.bloques| > 0 \longrightarrow res = \frac{montoTotalTransacciones(BCoin.bloques)}{cantidadTransacciones(BCoin.bloques)}\}$

}

`aux montoTotalTransacciones` (s:$seq\langle bloques\rangle$) : R $=$
$\sum_{i=0}^{|s|-1} montoPorBloque(s[i])$;

`aux montoPorBloque` (s:bloque) : R $=$
$\sum_{i=0}^{|b|-1}$ `IfThenElse`$(b_1[j]_1 \neq 0, b_1[j]_3, 0)$;

`aux cantidadTransacciones` (s:$seq\langle$bloque$\rangle$) : R $=$
$\sum_{i=0}^{|s|-1}$ `IfThenElse`$(s[i]_{1_{0_1}}, |s[i]_1| - 1, |s[i]_1|)$;

2.4. Ejercicio 4

`proc cotizacionAPesos` (in BCoin : $ BerretaCoin , in c : $seq\langle Cotizacion\rangle$) : $seq\langle Z\rangle$\{

`requiere` $\{|BCoin.bloques| = |c|\}$

`requiere` $\{(\forall i : Z)(0 \leq i < |c| \longrightarrow_L c[i] > 0)\}$

`asegura` $\{|res| = |BCoin.bloques| \wedge |res| = |c|\}$

`asegura` $\{(\forall i : Z)(0 \leq i < |res|) \longrightarrow_L res[i] = montoPorBloque(BCoin.bloques[i]) * c[i]\}$

}