Algoritmos y Estructuras de Datos II, DC, UBA.
Grupo Arboloneta
TP 1: Especificación TADs Lollapatuza

Integrante	LU	Correo electrónico
Rivero, Bárbara	1206/22	barbara.m.rivero@gmail.com
Sanguinetti, Iván	331/22	ivan.sanguinetti18@gmail.com
Neville, Matu	88/22	nevillematias@gmail.com
Ruz, Luciano	589/22	luciruzveloso@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Índice

TAD cantidad es nat
TAD porcentaje es nat
TAD persona es nat
TAD venta es multiconj(tupla(item, cantidad))

1. TAD LOLLAPATUZA

TAD LOLLAPATUZA

géneros lolla

exporta lolla, generadores, observadores, otras operaciones

usa MULTICONJ, CONJ, ITEM, PERSONA, PUESTO, NAT, CANTIDAD, VENTA

igualdad observacional

$$(\forall l, l' : \text{lolla})\ \left(l =_{\text{obs}} l' \iff \begin{pmatrix}\text{puestos(l)} =_{\text{obs}} \text{puestos(l')} \wedge \text{personasHabilitadas(l)} =_{\text{obs}} \text{perso-}\\ \text{nasHabilitadas(l')}\end{pmatrix}\right)$$

observadores básicos

puestos	: lolla	$\longrightarrow$	multiconj(puesto)
personasHabilitadas	: lolla	$\longrightarrow$	conj(persona)

generadores

nuevoLolla	: multiconj(puesto) cq $\times$ conj(persona) cp	$\longrightarrow$	lolla
	$\left\{\begin{array}{l}(\forall p\text{: persona})(\forall q, q'\text{: puesto})\ (p \in \text{cp} \wedge q \in \text{cq} \wedge q' \in \text{cq}) \ \Rightarrow\ _L \text{def(p, historialVentas(q))} \wedge_L \text{cp} = \\ \text{claves(historialVentas(q))} \wedge_L \text{obtener(p, historialVentas(q)} = \emptyset \wedge_L \neg(\exists i, i'\text{: item})(\text{i} \in \text{menu(q)} \wedge \\ \text{i'} \in \text{menu(q')} \wedge \text{q} \neq \text{q'} \wedge \text{i} = \text{i'} \wedge \text{precio(i)} \neq \text{precio(i')}\end{array}\right\}$		
personaCompra	: lolla l $\times$ puesto q $\times$ persona p $\times$ venta v	$\longrightarrow$	lolla
	$\left\{\begin{array}{l}\text{p} \in \text{personasHabilitadas(l)} \wedge \text{q} \in \text{puestos(l)} \wedge (\forall tv, tv'\text{: tupla(item, cantidad)}\ \text{tv} \in \text{v} \wedge \pi_1(\text{tv}) \in \text{menu(q)} \\ \wedge_L\ \pi_2(\text{tv}) \leq \text{obtener}(\pi_1(\text{tv}), \text{stock(q)}) \wedge (\text{tv} \neq \text{tv'} \ \Rightarrow\ \pi_1(\text{tv}) \neq \pi_1(\text{tv'}))\end{array}\right\}$		
hackeo	: lolla l $\times$ persona p $\times$ item i	$\longrightarrow$	lolla
			{(∃q: puesto) q∈puestos(l) ∧ esHackeable?(q, p, i)}

otras operaciones

puestoHackeado	: lolla l $\times$ item i $\times$ persona p	$\longrightarrow$	puesto
			{(∃q: puesto) q∈puestos(l) ∧ esHackeable?(q, p, i)}
buscoPuestoAHackear	: multiconj(puestos) cq $\times$ persona p $\times$ item i	$\longrightarrow$	puesto
			$\{(\exists q: : \textit{puesto})(q \in \text{cq} \wedge \text{esHackeable(q,p,i)}\}$
maxQueSeGasto	: lolla l	$\longrightarrow$	nat
buscoMontoMax	: conj(puesto) $\times$ conj(persona) cp	$\longrightarrow$	nat $\quad \{\#(\text{cp}) > 0\}$
cuantoGastoEnLolla	: conj(puesto) cq $\times$ persona p	$\longrightarrow$	nat
			$\{\#(\text{cq}) > 0 \wedge (\forall q\text{: puesto})\ \text{q} \in \text{cq} \ \Rightarrow\ _L \text{def?(p, historialVentas(q))}\}$
cuantoGastoEn	: puesto q $\times$ persona p	$\longrightarrow$	nat
			{def?(p, historialVentas(q))}
sumoTodasLasVentas	: venta $\times$ puesto	$\longrightarrow$	nat
precioTotal	: tupla(item $\times$ cantidad) $\times$ conj(tupla(cantidad $\times$ porcentaje))	$\longrightarrow$	nat
buscoCant	: conj(tupla(cantidad $\times$ porcentaje)) $\times$ nat	$\longrightarrow$	nat
buscoDescDeEsaCant	: n $\times$ conj(tupla(cantidad $\times$ porcentaje)) cd	$\longrightarrow$	nat
			{(∃t: tupla(cantidad, porcentaje) (t ∈ cd ∧ n = π_1(dameUno(cd)))}
mayorGastador	: lolla l	$\longrightarrow$	persona
veoQuienGastoMax	: conj(persona) cp $\times$ conj(puesto) $\times$ nat n	$\longrightarrow$	persona $\quad \{\#\text{cp} > 0\}$

axiomas ($\forall cq$: conj(puesto)), ($\forall cp$: conj(persona)), ($\forall l$: lola), ($\forall q$: puesto), ($\forall v$: venta), ($\forall p$: persona), ($\forall i$: item), ($\forall q$: puesto), ($\forall n$: nat), ($\forall cd$: conj(tupla(cantidad,porcentaje)))), ($\forall tv$: tupla(item,cantidad))

puestos(nuevoLolla(cq,cp)) $\equiv$ cq
puestos(personaCompra(l,q,v)) $\equiv$ Ag(vender(q,v), puestos(l)-q)

puestos(hackeo(l,p,i)) ≡ Ag(sufreHackeoPuesto(puestoHackeado(l,p,i),p,i), puestos(l) - puestoHackeado(l,p,i))

personasHabilitadas(nuevoLolla(cq, cp)) ≡ cp
personasHabilitadas(personaCompra(l, q, p, v)) ≡ personasHabilitadas(l)
personasHabilitadas(hackeo(l,p,i)) ≡ personasHabilitadas(l)

puestoHackeado(l, p, i) ≡ buscoPuestoAHackear(puestos(l), p, i)
buscoPuestoAHackear(cq, p, i) ≡ **if** esHackeable?(dameUno(cq), p, i) **then**
dameUno(cq)
else
buscoPuestoAHackear(sinUno(cq), p, i)
fi
maxQueSeGasto(l) ≡ buscoMontoMax(puestos(l), personas(l))
buscoMontoMax(cq, cp) ≡ **if** #(cp) = 1 **then**
cuantoGastoEnLolla(cq, dameUno(cp))
else
max(cuantoGastoEnLolla(cq, dameUno(cp)), buscoMontoMax(cq, sinUno(cp)))
fi
cuantoGastoEnLolla(cq, p) ≡ **if** #(cq) = 1 **then**
cuantoGastoEn(dameUno(cq), p)
else
cuantoGastoEn(dameUno(cq), p) + cuantoGastoEnLolla(sinUno(cq), p)
fi
cuantoGastoEn(q, p) ≡ sumoTodasLasVentas(q, obtener(p, historialVentas(q)))
sumoTodasLasVentas(v, q) ≡ **if** ∅?(v) **then**
0
else
precioTotal(dameUno(v), obtener(π_1(dameUno(v)), descuentos(q)) + sumoTodasLasVentas(sinUno(v), q)
fi
precioTotal(tv, cd) ≡ aplicarDescuento(precio(π_1(tv))*π_2(tv), buscoDescDeEsaCant(buscoCant(cd, π_2(tv)), cd)
buscoCant(cd, n) ≡ **if** ∅?(cd) **then**
0
else
if π_1(dameUno(cd)) ≤ n **then**
max(π_1(dameUno(cd)), buscoCant(sinUno(cd), n))
else
buscoCant(sinUno(cd), n)
fi
fi
buscoDescDeEsaCant(n, cd) ≡ **if** n = π_1(dameUno(cd)) **then**
π_2(dameUno(cd))
else
buscoDescDeEsaCanta(n, sinUno(cd))
fi
mayorGastador(l) ≡ veoQuienGastoMax(personasHabilitadas(l), puestos(l), maxQueSeGasto(l)
veoQuienGastoMax(cp, cq, n) ≡ **if** #(cp) = 1 **then**
dameUno(p)
else
if cuantoGastoEnLolla(cq, dameUno(p)) = n **then**
dameUno(p)
else
veoQuienGastoMax(sinUno(cp), cq, n)
fi
fi

Fin TAD

2. TAD PUESTO DE COMIDA

TAD PUESTO DE COMIDA

géneros puesto

exporta puesto, generadores, observadores, otras operaciones

usa MULTICONJ, CONJ, ITEM, PERSONA, NAT, DICC, TUPLA, PORCENTAJE, CANTIDAD, VENTA

igualdad observacional

$$(\forall p, p' : \text{item}) \left(p =_{\text{obs}} p' \iff \begin{pmatrix} \text{menú(p)} =_{\text{obs}} \text{menú(p')} \land \text{stock(p)} =_{\text{obs}} \text{stock(p')} \\ \land \text{descuentos(p)} =_{\text{obs}} \text{descuentos(p')} \land \\ \text{historialVentas(p)} =_{\text{obs}} \text{historialVentas(p')} \end{pmatrix} \right)$$

observadores básicos

menú	:	puesto	$\longrightarrow$	conj(item)
stock	:	puesto	$\longrightarrow$	dicc(item,cantidad)
descuentos	:	puesto	$\longrightarrow$	dicc(item,conj(tupla(cantidad, porcentaje)))
historialVentas	:	puesto	$\longrightarrow$	dicc(persona, venta)

generadores

abrirPuesto : conj(item) m × dicc(item × cantidad) s × dicc(item × conj(tupla(cantidad × porcentaje))) dd $\longrightarrow$ puesto

$$\left\{ \begin{array}{l} \text{claves(s)} = \text{m} \land \text{claves(dd)} = \text{m} \land (\forall i\text{: item})(\text{def?(s)} \rightarrow (\forall t\text{: tupla(cantidad, porcentaje)}(\text{t} \in \\ \text{obtener(i,s)} \rightarrow \pi_2 < 100 \land \neg(\exists t\text{'}\text{: tupla(cantidad, porcentaje)}(\text{t'} \in \text{obtener(i,s)} \land \text{t} \neq \text{t'} \land \pi_1(\text{t'}) < \\ \pi_1(\text{t}) \land \pi_2(\text{t'}) > \pi_2(\text{t}))))) \end{array} \right\}$$

vender : puesto q × persona p × venta v $\longrightarrow$ puesto $\{(\forall tv\text{: tupla(item,cantidad)})(\text{tv} \in \text{v} \land \pi_2(\text{tv}) \leq \text{obtener}(\pi_1(\text{tv}),\text{stock(p)}))\}$

sufreHackeoPuesto : puesto q × persona p × item i $\longrightarrow$ puesto $\{\text{esHackeable?(q, p, i)}\}$

otras operaciones

esHackeable? : puesto q × persona p × item i $\longrightarrow$ bool $\{\text{def?(p, historialVentas(q))} \land \text{def?(i, descuentos(q))}\}$

existeCompraSinDesc? : venta × nat $\longrightarrow$ bool

modificarStock : dicc(item × cantidad) s × persona × venta v $\longrightarrow$ puesto $\{(\forall tv\text{: tupla(item,cantidad)})(\text{tv} \in \text{v} \land \pi_2(\text{tv}) \leq \text{obtener}(\pi_1(\text{tv}),\text{s}))\}$

buscaVentaHackeada : dicc(persona × venta) dv × dicc(item × conj(tupla(cantidad × porcentaje)) dd × persona p × item i $\longrightarrow$ puesto

$$\left\{ \begin{array}{l} (\exists p\text{: persona})(\text{def?(p,dv)} \land (\exists tv\text{: tupla(item,cantidad) (tv} \in \text{obtener(i,dv)} \land (\forall i\text{: item})(\text{def?(i,dd)}) \\ \rightarrow (\forall tc\text{: tupla(cantidad,porcentaje))}(\text{tc} \in \text{obtener(i,dd)} \rightarrow \pi_1(\text{tv}) = \text{i} \land \pi_2 < \pi_1(\text{tc}) \ )))) \end{array} \right\}$$

cantidadMinDesc : dicc(item × conj(tupla(cantidad × porcentaje)) dd × item i $\longrightarrow$ nat $\{\text{def?(i,dd)} \land \#(\text{obtener(i,dd)}) > 0)\}$

buscoElMinimo : conj(tupla(cantidad × porcentaje)) d $\longrightarrow$ nat $\{\#\text{d} > 0)\}$

axiomas $(\forall m\text{: conj(item)})$, $(\forall s\text{: dicc(item,cantidad)}$, $(\forall dd\text{: dicc(item,conj(tupla(cantidad, porcentaje))))}$, $(\forall q\text{: puesto})$, $(\forall p\text{: persona})$, $(\forall i\text{: item})$, $(\forall v\text{: venta})$, $(\forall dv\text{: dicc(persona, venta)}$, $(\forall dd\text{: conj(tupla(cantidad, porcentaje)))}$

menú(abrirPuesto(m, s, dd)) $\equiv$ m
menú(vender(q, p, v)) $\equiv$ menú(q)
menú(sufreHackeoPuesto(q, p, i)) $\equiv$ menú(q)

stock(abrirPuesto(m, s, dd)) $\equiv$ s
stock(vender(q, p, v)) $\equiv$ modificarStock(stock(q), v)
stock(sufreHackeoPuesto(q, p, i)) $\equiv$ definir(i,obtener(i,stock(q)) + π_2(buscaVentaHackeada(historialVentas(q), descuentos(q),p,i), stock(q))

descuentos(abrirPuesto(m, s, dd)) $\equiv$ dd
descuentos(vender(q, p, v)) $\equiv$ descuentos(q)

descuentos(sufreHackeoPuesto(q, p, i)) ≡ descuentos(q)

historialVentas(abrirPuesto(m, s, dd)) ≡ ∅
historialVentas(vender(q, p, v)) ≡ definir(p, v ∪ obtener(p,historialVentas(q)), historialVentas(q))
historialVentas(sufreHackeoPuesto(q, p, i)) ≡ definir(p, obtener(p, historialVentas(q)) - buscarVentaHackeada(historialVentas(q), descuentos(q), p, i), historialVentas(q))
esHackeable?(q, p, i) ≡ existeCompraSinDesc?(obtener(p, historialVentas(q)), buscoElMinimo(obtener(i, descuentos(q))))
existeCompraSinDesc?(v, n) ≡ **if** ∅?(dameUno(v)) **then**
false
else
if π_2(dameUno(v)) < n **then**
true
else
existeCompraSinDesc?(sinUno(v), n)
fi
fi
modificarStock(s, v) ≡ **if** $\#v = 1$ **then**
definir(π_1(dameUno(v)), obtener(π_1(dameUno(v)), s) −π_2(dameUno(v)), s)
else
modificarStock(definir(π_1(dameUno(v)), obtener(π_1(dameUno(v)), s) − π_2(dameUno(v)), s), sinUno(v))
fi
buscaVentaHackeada(dv, dd, p, i) ≡ buscoVentaSinDesc(obtener(p, dv), cantidadMinDesc(dd, i))
buscoVentaSinDesc(v,n) ≡ **if** #v= 1 **then**
dameUno(v)
else
if π_2(dameUno(v)) < n **then**
dameUno(v)
else
buscoVentaSinDesc(sinUno(v),n)
fi
fi
cantidadMinDesc(dd,i) ≡ buscoElMinimo(obtener(i, dd))
buscoElMinimo(cd) ≡ **if** #(cd)= 1 **then**
π_1(dameUno(cd))
else
min(π_1(dameUno(cd)), buscoElminimo(sinUno(cd)))
fi

Fin TAD

3. TAD ITEMS

TAD ITEMS

géneros item

exporta item, generadores, observadores

usa STRING, NAT

igualdad observacional

$$(\forall i, i' : \text{item}) \left(i =_{\text{obs}} i' \iff \begin{pmatrix} \text{precio(i)} =_{\text{obs}} \text{precio(i)} \wedge \\ \text{nombre(i)} =_{\text{obs}} \text{nombre(i')} \end{pmatrix} \right)$$

observadores básicos

precio	: item	$\longrightarrow$	nat
nombre	: item	$\longrightarrow$	string

generadores

prepararItem	: string s $\times$ nat n	$\longrightarrow$	item

axiomas ($\forall n$: nat)($\forall s$: string)

precio(prepararItem(s, n)) $\equiv$ n

nombre(prepararItem(s, n)) $\equiv$ s

Fin TAD