Algoritmos y Estructuras de Datos II
Recuperatorio del primer parcial – 7 de julio de 2014

Aclaraciones

- El parcial es **a libro abierto**.
- Cada ejercicio debe entregarse **en hojas separadas**.
- Incluir en cada hoja el número de orden asignado, número de hoja, apellido y nombre.
- Al entregar el parcial, completar el resto de las columnas en la planilla.
- Cada ejercicio se calificará con **MB**, **B**, **R** o **M**, y podrá recuperarse independientemente de los demás. Podrán aprobarse los parciales de la materia con hasta 2 (dos) ejercicios **R** (regular) **entre ambos exámenes**, siempre que ninguno de ellos sea un ejercicio 1. Para más detalles, ver *Información sobre la cursada* en el sitio Web.

Ej. 1. Especificación

La empresa de transporte aéreo AeroBox necesita crear un sistema que le permita controlar sus operaciones en el mundo. AeroBox se dedica al transporte de *containers* entre distintas ciudades. Puede suponerse que la empresa sólo dará servicio a un conjunto fijo de ciudades.

La ruta aérea que utiliza AeroBox es peculiar: desde cada ciudad sólo se puede volar a una única ciudad, y a cada ciudad sólo se puede llegar volando desde una única ciudad, formando una suerte de anillo unidireccional de ciudades. En cualquier ciudad, en cualquier momento pueden agregarse nuevos containers, indicando a qué ciudad de destino deben ser transportados.

Cada vez que AeroBox adquiere un nuevo avión, lo coloca en alguna de sus ciudades. Cada avión tiene una capacidad límite (en cantidad de containers) que no debe excederse.

Cuando un avión aterriza (o cuando es colocado) en una ciudad, inmediatamente carga en su bodega tantos containers como sea posible, respetando su límite de capacidad. Una vez cargado, el avión se queda esperando que le den la orden de despegue rumbo a la siguiente ciudad. Al llegar a una ciudad, en caso de haber containers en su bodega que correspondan a ese destino, los descarga antes de volver a llenarla.

Se pide especificar usando tipos abstractos de datos el comportamiento de la empresa transportista AeroBox y sus aviones. Interesa saber en todo momento qué avión transportó más containers en total. En caso de haber más de uno, puede devolverse cualquiera de ellos.

Ej. 2. Inducción estructural

El TAD BolsaDeGatos modela un curioso contenedor urbano, suerte de conjunto felino al que pueden agregarse tanto gatos negros como gatos blancos. Toda bolsa de gatos debe contener al menos un gato negro.

TAD GatoNegro **es** Nat

TAD GatoBlanco **es** Nat

TAD BolsaDeGatos

generadores

nueva	: gatoNegro	$\longrightarrow$	bdg
agNegro	: gatoNegro $\times$ bdg	$\longrightarrow$	bdg
agBlanco	: gatoBlanco $\times$ bdg	$\longrightarrow$	bdg

observadores básicos

$\bullet \in_N \bullet$	: gatoNegro $\times$ bdg	$\longrightarrow$	bool
$\bullet \in_B \bullet$	: gatoBlanco $\times$ bdg	$\longrightarrow$	bool

otras operaciones

unir	: bdg $\times$ bdg	$\longrightarrow$	bdg

axiomas $\forall gn$: gatoNegro $\forall gb$: gatoBlanco $\forall B, C$: bdg

$N0$)	$gn \in_N$ nueva(gn')	$\equiv$	$gn = gn'$
$N1$)	$gn \in_N$ agNegro(gn', B)	$\equiv$	$gn = gn' \vee gn \in_N B$
$N2$)	$gn \in_N$ agBlanco(gb, B)	$\equiv$	$gn \in_N B$
$B0$)	$gb \in_B$ nueva(gn)	$\equiv$	false
$B1$)	$gb \in_B$ agNegro(gn, B)	$\equiv$	$gb \in_B B$
$B2$)	$gb \in_B$ agBlanco(gb', B)	$\equiv$	$gb = gb' \vee gb \in_B B$
$U0$)	unir(nueva(gn), B)	$\equiv$	agNegro(gn, B)
$U1$)	unir(agNegro(gn, B), C)	$\equiv$	unir(B, agNegro(gn, C))
$U2$)	unir(agBlanco(gb, B), C)	$\equiv$	unir(B, agBlanco(gb, C))

Fin TAD

Se pide demostrar por inducción estructural la siguiente propiedad:

$$\big(\forall B, C : \text{bdg}\big)\big(\forall gn : \text{gatoNegro}\big)\big(\, gn \in_N \text{unir}(B, C) \Longrightarrow (gn \in_N B) \vee (gn \in_N C)\,\big)$$

a) Escribir el **predicado unario**. Luego escribir el **esquema completo de inducción** a utilizar. En el esquema, marcar **claramente** CB(s), PI(s), HI(s), TI(s) y alcance de cada cuantificador.

b) Plantear el/los caso(s) base y resolverlo(s), justificando cada paso de la demostración.

c) Plantear el/los paso(s) inductivo(s) y resolverlo(s), justificando cada paso de la demostración.

Ej. 3. Diseño

El almacén de Manolo comercializa diversos productos alimenticios. Manolo repone su stock comprando productos que luego vende a sus clientes. Cada producto tiene un nombre, un precio de compra y uno de venta (no necesariamente mayor). A Manolo le obsesiona calcular, dado un producto, cuánto dinero obtuvo hasta el momento con el mismo (es decir, la ganancia o pérdida obtenida por ese producto). A veces también le interesa averiguar cuánto compró y vendió de cada producto. También desea conocer su ganancia (o pérdida) total.

TAD Manolo

generadores

nuevo : $\longrightarrow$ manolo

comercializarProducto : manolo m $\times$ string p $\times$ nat *compra* $\times$ nat *venta* $\longrightarrow$ manolo $\quad \{p \notin productos(m)\}$

comprar : manolo m $\times$ string p $\times$ nat *cantidad* $\longrightarrow$ manolo $\quad \{p \in productos(m)\}$

vender : manolo m $\times$ string p $\times$ nat *cantidad* $\longrightarrow$ manolo $\quad \{p \in productos(m) \wedge_{\text{L}} stock(m, p) \geq cantidad\}$

observadores básicos

productos : manolo $\longrightarrow$ conj(string)

precioCompra : manolo m $\times$ string *prod* $\longrightarrow$ nat $\quad \{prod \in productos(m)\}$

precioVenta : manolo m $\times$ string *prod* $\longrightarrow$ nat $\quad \{prod \in productos(m)\}$

compras : manolo m $\times$ string *prod* $\longrightarrow$ nat $\quad \{prod \in productos(m)\}$

gananciaProducto : manolo m $\times$ string *prod* $\longrightarrow$ int $\quad \{prod \in productos(m)\}$

otras operaciones

ventas : manolo m $\times$ string *prod* $\longrightarrow$ nat $\quad \{prod \in productos(m)\}$

stock : manolo m $\times$ string *prod* $\longrightarrow$ nat $\quad \{prod \in productos(m)\}$

gananciaTotal : manolo $\longrightarrow$ int

Fin TAD

A la hora del diseño se elige la siguiente estructura de representación:

manolo **se representa con** estr, donde

estr es tupla $\langle$ *productos*: conj(string),
stock: dicc(string, nat),
precios: dicc(string, tupla(nat, nat)),
compras: dicc(string, nat),
ventas: dicc(string, nat),
gananciaTotal: int $\rangle$

en la cual:

- *productos* son los nombres de los productos que comercializa Manolo,
- *stock* indica la cantidad disponible de cada producto,
- *precios* indica los precios de compra y venta de cada producto,
- *compras* y *ventas* indican la cantidad comprada y vendida, respectivamente, de un producto dado,
- *gananciaTotal* indica el valor de la ganancia o pérdida total de Manolo hasta el momento.

a) Escribir el invariante de representación en castellano.

b,c) Escribir formalmente *b)* el invariante de representación y *c)* la función de abstracción.