Algoritmos y Estructuras de Datos II
Primer parcial – Sábado 30 de septiembre de 2017

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, nombre, LU y **turno**.
- Cada ejercicio se calificará con **P**romocionado, **A**probado, **R**egular, o **I**nsuficiente.
- El parcial estará aprobado si: (el ejercicio 1 tiene al menos **A**) y (el ejercicio 3 tiene al menos una **A** o (el ejercicio 3 tiene una **R** y el 2 al menos una **A**)).

Ej. 1. Especificación

Técnicos a Domicilio (o simplemente "TaD"), es una empresa que provee servicio técnico para problemas de electricidad en hogares y empresas. TaD cuenta con un grupo de técnicos altamente capacitados para atender la demanda de sus clientes y tiene una estrategia de trabajo algo particular. Cuando alguien solicita un técnico, la central de TaD verifica si alguno de sus técnicos se encuentra en la empresa y de ser así envía inmediatamente un técnico al domicilio de la persona. En caso de no haber técnicos disponibles en ese momento (i.e., todos se encuentran atendiendo algún pedido), el pedido queda *pendiente de asignación* a la espera de que algún técnico se desocupe.

Por otro lado, cuando un técnico termina de resolver un problema, y antes de retirarse de ese domicilio, el técnico avisa por radio a la central que quedó disponible para otro trabajo. Si existiesen en ese momento pedidos *pendientes de asignación*, la central le asigna al técnico el más cercano al domicilio en el que éste se encuentra y el técnico se dirige automáticamente hacia allí (si hay más de un pendiente a la misma distancia mínima, se asignará al pedido entre éstos que lleve más tiempo esperando). Por el contrario, de no haber trabajos pendientes, el técnico regresa a la central y queda disponible para futuros trabajos.

Modelar con un TAD la empresa *Técnicos a Domicilio* descripta teniendo en cuenta además que interesa saber, dada una dirección, quiénes fueron los técnicos que la visitaron la mayor cantidad de veces (aun si todavía no resolvieron el inconveniente técnico).

Observación: Se puede asumir como dado el TAD Dirección que exporta el género *dirección* y la operación *dist(d, d′)* que devuelve un *nat* que representa la distancia entre las direcciones d y d'.

Ej. 2. Complejidad

Discutir la veracidad de las siguientes afirmaciones, justificando adecuadamente en cada caso:

1. Si $f \in O(n)$ y $g(n) = n^2$, entonces $f \circ g \in O(g)$.
2. Si $f \in O(n)$ y $g(n) = n^2$, entonces $g \circ f \in O(f)$.
3. La complejidad temporal del *peor caso* del Algoritmo 1 es $\Theta(n^2)$.
4. La complejidad temporal del *mejor caso* del Algoritmo 1 es $O(n^2)$.

Algoritmo 1 Muestra para cada número, cuántos menores consecutivos hay entre el inicio y su posición

```
1: function MostrarMenoresConsecutivos(arreglo de enteros A)
2:     for i := 0 ... Long(A) - 1 do
3:         j := 0;
4:         while j < i && A[j] < A[i] do
5:             j := j + 1;
6:         end while
7:         imprimir j;
8:     end for
9: end function
```

Observación: Consideramos que las operaciones de suma, resta y comparación entre enteros son operaciones elementales, así también como la impresión de la línea 7.

Ej. 3. Diseño

La organización Redes en Paralelo (o simplemente "ReP") agrupa varias redes sociales en un mismo sitio. El objetivo del sitio es que sus usuarios puedan comunicarse con sus "amistades" de distintas redes sociales, utilizando un único portal. Para ello, ReP registra a qué redes sociales (dentro de un conjunto predeterminado de redes) pertenece cada uno de sus usuarios y monitorea las relaciones de amistad presentes en estas redes. Dos usuarios serán "amigos" en ReP si y sólo si son amigos en alguna de las redes monitoreadas. ReP brinda además algunos servicios extra como videollamadas, compartir archivos, etc., pero sólo ofrece estos servicios a aquellas relaciones de amistad que sean suficientemente fuertes. Para ello se define que dos usuarios son *super amigos* si son amigos en 3 o más redes sociales. El siguiente TAD modela el sitio de remates descripto (aunque se omiten las axiomatizaciones).

TAD usuario es nat y TAD red es string

TAD ReP

observadores básicos

usuarios	: rep	$\longrightarrow$	conj(usuario)	
redes	: rep	$\longrightarrow$	conj(red)	
miembro	: rep r $\times$ usuario u $\times$ red t	$\longrightarrow$	bool	$\{u \in \text{usuarios}(r) \land t \in \text{redes}(r)\}$
amigosEn	: rep r $\times$ usuario u $\times$ usuario u' $\times$ red t	$\longrightarrow$	bool	$\{\{u, u'\} \subseteq \text{usuarios}(r) \land t \in \text{redes}(r)\}$

generadores

iniciar	: conj(red)	$\longrightarrow$	rep	
altaUsuario	: rep r $\times$ usuario u	$\longrightarrow$	rep	$\{u \notin \text{usuarios}(r)\}$
altaEnRed	: rep r $\times$ usuario u $\times$ red t	$\longrightarrow$	rep	$\{u \in \text{usuarios}(r) \land t \in \text{redes}(r)\}$
amistadEnRed	: rep r $\times$ usuario u $\times$ usuario u' $\times$ red t	$\longrightarrow$	rep	

$$\left\{ \begin{array}{l} u \neq u' \land \{u, u'\} \subseteq \text{usuarios}(r) \land t \in \text{redes}(r) \land_{\text{L}} \text{miembro}(r, u, t) \land \text{miembro}(r, u', t) \land \\ \neg\text{amigosEn}(r, u, u', t) \end{array} \right\}$$

otras operaciones

superAmigos	: rep r $\times$ usuario u $\times$ usuario u'	$\longrightarrow$	bool	$\{\{u, u'\} \subseteq \text{usuarios}(r)\}$

axiomas

$\ldots \equiv \ldots$

Fin TAD

Se decidió utilizar la siguiente estructura para representar el TAD:

ReP **se representa con** estr

donde estr es tupla $\langle$ *usuarios*: conj(usuario),
redes: conj(red),
membresías: dicc(usuario,conj(red)),
amigos: dicc(usuario, conj(tupla$\langle$red,usuario$\rangle$))),
superAmigos: dicc(usuario,conj(usuario))$\rangle$)

En esta estructura, *usuarios* y *redes* almacenan los usuarios y las redes registradas en el ReP y *membresías* registra las redes a las que está suscripto cada usuario. Por otro lado para cada usuario u, *amigos* guarda las relaciones de amistad entre u y otros usuarios en diferentes redes sociales y *superAmigos* registra el conjunto de usuarios que son amigos de u en al menos 3 redes diferentes.

Teniendo en cuenta el TAD ReP y la estructura elegida para su representación se pide:

a) Escribir en castellano el invariante de representación.

b) Escribir formalmente el invariante de representación.

c) Escribir formalmente la función de abstracción.

Solución Ej 1

TAD TAD

igualdad observacional

$$(\forall t, t' : \text{tad}) \left(t =_{\text{obs}} t' \iff \left(\begin{array}{l} \text{pendientes}(t) =_{\text{obs}} \text{pendientes}(t') \wedge \\ \text{técnicos}(t) =_{\text{obs}} \text{técnicos}(t') \wedge \\ \text{libres}(t) =_{\text{obs}} \text{libres}(t') \wedge_{\text{L}} (\\ (\forall\ p : \text{técnico})\ (p \in \text{técnicos}(t) \Rightarrow_{\text{L}} \\ (\forall\ d : \text{dirección})\ (\text{visitas}(t, p, d) =_{\text{obs}} \text{visitas}(t', p, d)) \\ \wedge\ (p \in \text{libres}(t) \Rightarrow_{\text{L}} \\ \text{trabajandoEn}(t, p) =_{\text{obs}} \text{trabajandoEn}(t', p))))) \end{array} \right) \right)$$

géneros tad

observadores básicos

pendientes : tad $\longrightarrow$ secu(dirección)

técnicos : tad $\longrightarrow$ conj(técnico)

libres : tad $\longrightarrow$ conj(técnico)

trabajandoEn : tad t × técnico p $\longrightarrow$ dirección { $p \in (\text{técnicos}(t) \setminus \text{libres}(p))$ }

visitas : tad t × técnico p × dirección d $\longrightarrow$ nat { $p \in \text{técnicos}(t)$ }

generadores

iniciar : conj(técnico) $\longrightarrow$ tad

solicitar : tad × dirección $\longrightarrow$ tad

finalizar : tad t × técnico p $\longrightarrow$ tad { $p \in (\text{técnicos}(t) \setminus \text{libres}(p))$ }

otras operaciones

másLaVisitaron : tad × dirección $\longrightarrow$ conj(técnico)

próximo : secu(dirección) s × dirección d $\longrightarrow$ dirección { $\neg$ vacía?(s) }

soloMaximos : tad t × conj(técnico) ps × dirección $\longrightarrow$ conj(técnico) { $ps \subseteq \text{técnicos}(t)$ }

maxVisitas : tad t × conj(técnico) ps × dirección $\longrightarrow$ nat { $\emptyset \subset ps \subseteq \text{técnicos}(t)$ }

axiomas ($\forall$ t : tad) , ($\forall$ p, p' : técnico) , ($\forall$ d, d' : dirección)

técnicos(iniciar(ts)) $\equiv$ ts

técnicos(solicitar(t,d)) $\equiv$ técnicos(t)

técnicos(finalizar(t,p)) $\equiv$ técnicos(t)

pendientes(iniciar(ts)) $\equiv$ <>

pendientes(solicitar(t,d)) $\equiv$ **if** $\emptyset$?(libres(t)) **then** pendientes(t) $\circ$ d **else** <> **fi**

pendientes(finalizar(t,p)) $\equiv$ **if** vacía?(pendientes(t)) **then**
<>
else
borrar(próximo(pendientes(t), trabajandoEn(t, p)), pendientes(t))
fi

libres(iniciar(ts)) $\equiv$ ts

libres(solicitar(t,d)) $\equiv$ **if** $\emptyset$?(libres(t)) **then** $\emptyset$ **else** sinUno(libres(t)) **fi**

libres(finalizar(t,p)) $\equiv$ **if** vacía?(pendientes(t)) **then** Ag(p,libres(t)) **else** $\emptyset$ **fi**

TrabajandoEn(solicitar(t,d), p) $\equiv$ **if** $p \in$ libres(t) **then** d **else** TrabajandoEn(t,p) **fi**

TrabajandoEn(finalizar(t,p'), p) $\equiv$ **if** $p = p'$ **then**
próximo(pendientes(t), trabajandoEn(t, p))
else
TrabajandoEn(t,p)
fi

visitas(iniciar(ts), p, d) $\equiv$ 0

visitas(solicitar(t,d'), p, d) $\equiv$ **if** $d = d' \wedge \neg$vacío?(libres(t)) $\wedge_{\text{L}}$ $p =$ dameUno(libres(t)) **then**
visitas(t, p, d) + 1
else
visitas(t, p, d)
fi

visitas(finalizar(t,p'), p, d) $\equiv$ **if** $p = p' \wedge \neg$vacía?(pendientes(t)) $\wedge_{\text{L}}$ $d =$ próximo(pendientes(t), trabajandoEn(t, p)) **then**
visitas(t, p, d) + 1
else
visitas(t, p, d)
fi

próximo(ds,d) $\equiv$ **if** long(ds) = 1 **then**
prim(ds)
else
if dist(d, prim(ds)) $\leq$ dist(d, próximo(fin(ds), d)) **then**
prim(ds)
else
próximo(fin(ds), d)
fi
fi

másLaVisitaron(t,d) $\equiv$ soloMaximos(t,técnicos(t), d)

soloMaximos(t,ps,d) $\equiv$ **if** #(ps) $\leq$ 1 **then**
ps
else
if visitas(t, dameUno(ps), d) > maxVisitas(t, sinUno(ps),d)) **then**
{ dameUno(ps) }
else
if visitas(t, dameUno(ps), d) = maxVisitas(t, sinUno(ps),d)) **then**
Ag(dameUno(ps), soloMaximos(t,sinUno(ps),d))
else
soloMaximos(t,sinUno(ps),d)
fi
fi
fi

maxVisitas(t,ps,d) $\equiv$ visitas(t, dameUno(soloMaximos(t,ps,d)), d)

Fin TAD

Solución Ej 2

1. La afirmación es **verdadera**: Debemos hallar c, n_0, tales que $f(g(n)) \leq cn^2$, para todo $n \geq n_0$. Sabemos que existen c_1, n_1 tales que $f(n) \leq c_1 n$, para todo $n \geq n_1$. Como $g(n) \geq n_1$ para todo $n \geq n_1$, entonces $f(g(n)) \leq c_1 g(n) = c_1 n^2$, para todo $n \geq n_1$. Así, usando $c := c_1$ y $n_0 := n_1$, probamos que $f \circ g \in O(g)$.
2. La afirmación es **falsa**: Si tomamos $f(n) = n$, lo cual cumple con $f \in O(n)$, entonces $(g \circ f)(n) = g(f(n)) = g(n) = n^2$ y es fácil ver que $n^2 \notin O(n)$.
3. La afirmación es **cierta**: El peor caso del algoritmo es cuando todos los pares de posiciones están ordenados y en ese caso la línea 6 se ejecuta $(\sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1)$ veces, lo cual es igual a $n(n-1)/2$. Que claramente pertenece a $\theta(n^2)$ (Falta probarlo!).
4. La afirmación es **cierta**: El mejor caso del algoritmo es cuando el primer elemento es mayor que todos los demás, con lo cual nunca se entra al while y en ese caso la cantidad de operaciones es lineal en el tamaño del arreglo, lo cual claramente pertenece a $O(n)$ y esto está dentro de $O(n^2)$ (Falta probarlo!).

Solución Ej 3

a y b) Tener en cuenta que esto no está escrito de la manera pedida en el parcial. Primero se debería escribir exclusivamente en castellano, y luego formalmente en lógica.

- claves(membresías) = usuarios
- Para cada $u \in$ usuarios, obtener(u, membresías) $\subseteq$ redes.
- claves(amigos) = usuarios

- Para cada $u \in$ usuarios, para cada tupla $< r, u' >$, si $< r, u' > \in$ obtener(u, amigos) $\Rightarrow_{\text{L}}$ ($r \in$ obtener(u, membresías) $\wedge$ $r \in$ obtener(u', membresías) $\wedge$ $u' \in$ usuarios $\wedge_{\text{L}} < r, u > \in$ obtener(u', amigos)).
- claves(superamigos) = usuarios y para cada $u \in$ usuarios, obtener(u, superamigos) $\subseteq$ usuarios.
- Para cada $u, u' \in$ usuarios, $u \in$ obtener(u', superamigos) si y solo si existen r_1, r_2, r_3 distintas tales que $\{< r1, u >, < r2, u >, < r3, u >\} \subseteq$ obtener(u',amigos).

c) *Abs*: estr $e \rightarrow$ REP { *Rep*(e) }

Abs(e) = r /

usuarios(r) = e.usuarios $\wedge$

redes(r) = e.redes $\wedge_{\text{L}}$

($\forall$ u : usuario) ($\forall$ t : red) **(**$u \in$ usuarios(r) $\wedge$ $t \in$ redes(r) $\Rightarrow_{\text{L}}$

miembro(r, u, t) $\Leftrightarrow$ $t \in$ obtener(u, e.membresías) $\wedge$

($\forall$ u' : usuario) (amigosEn(r, u, u', t) $\Leftrightarrow$ $< r, u' > \in$ obtener(u, e.amigos)) **)**