Algoritmos y Estructuras de Datos II
Primer parcial – Sábado 5 de mayo

Aclaraciones

- El parcial es a **libro abierto**.
- Incluir en cada hoja el número de orden asignado, apellido y nombre y número de hoja.
- Al entregar el parcial completar los datos faltantes en la planilla.
- Cada ejercicio se calificará con A, R o I, y se puede recuperar de forma independiente.
- Para aprobar el parcial se deberá obtener al menos una A en el primer ejercicio y en los ejercicios 2 y 3 se deberá obtener al menos una A y una R.
- Para considerar un parcial promocionado se debe obtener A en todos los ejercicios y **además** haber realizado una resolución ampliamente satisfactoria de todos ellos. En estos casos se agregará una P y una nota numérica.
- Se debe entregar cada ejercicio en hojas separadas.

Ej. 1. Especificación

Se desea especificar un sistema que controla las ventanillas de atención al cliente de cierta organización. La atención se organiza mediante distintos tipos de trámite. Los tipos de trámite se identifican por un nombre único en forma de un string, y son fijos. Se cuenta con n ventanillas de atención, identificadas con naturales entre 0 y $n-1$, cada una de las cuales tiene la capacidad de manejar uno o varios tipos de trámite. La cantidad de ventanillas y los tipos de trámite que puede manejar cada una también se mantienen fijos.

Cuando llega un cliente se identifica por su DNI y el tipo de trámite que desea realizar. Si hay una ventanilla libre capaz de manejarlo, es atendido inmediatamente. En caso de haber varias posibilidades, se usa siempre la ventanilla de número mas bajo. Si no hay ventanillas libres, queda esperando. Cuando un cliente que está siendo atendido se retira, la ventanilla que fue liberada pasa a atender a la persona que está esperando hace más tiempo de las que tienen un tipo de trámite dentro de sus capacidades. Si ninguno de los clientes en la cola de espera puede ser atendido, la ventanilla queda libre.

Una persona no puede volver a ponerse en la cola para el mismo u otro trámite mientras está siendo atendida o se encuentra esperando en la cola. Las personas nunca se retiran sin ser atendidas.

Se desea saber en todo momento, para cada ventanilla, si está libre u ocupada, y para las ocupadas, a quién se está atendiendo.

Ej. 2. Inducción Estructural

Dadas las siguientes funciones sobre secuencias:

$$
\begin{array}{lll}
\# : \text{secu}(\alpha) & \longrightarrow & \text{nat} \\
\#_1) \quad \#(<>) & \equiv & 0 \\
\#_2) \quad \#(a \bullet s) & \equiv & 1+\#(s)
\end{array}
\qquad
\begin{array}{lll}
\text{máximos} : \text{secu(nat)}\ s \times \text{secu(nat)}\ t & \longrightarrow & \text{secu(nat)} \quad \{\#(s) = \#(t)\} \\
m_1) \quad \text{máximos}(<>, t) & \equiv & <> \\
m_2) \quad \text{máximos}(a \bullet s, t) & \equiv & \text{máx}(a, \text{prim}(t)) \bullet \text{máximos}(s, \text{fin}(t))
\end{array}
$$

Se quiere probar por inducción estructural la siguiente propiedad:

$$(\forall s\text{: secu(nat)})(\forall t\text{: secu(nat)})[(\#(s) \equiv \#(t)) \Rightarrow_{\text{L}} (\text{máximos}(s,t) \equiv \text{máximos}(t,s))]$$

Los símbolos lógicos que aparecen son de lógica de primer orden. El TAD Bool no aparece en este ejercicio. Se puede asumir sin demostración los siguientes lemas:

$$
\begin{array}{ll}
\forall x : \text{secu}(\alpha) & \\
pf) & \#(x) > 0 \Rightarrow_{\text{L}} (\text{prim}(x) \bullet \text{fin}(x) \ \equiv\ x) \\
v0) & \#(x) = 0 \Rightarrow_{\text{L}} (x \ \equiv\ <>)
\end{array}
\qquad
\begin{array}{ll}
\forall n, m : \text{nat} & \\
mx) & \text{máx}(n, m) \ \equiv\ \text{máx}(m, n)
\end{array}
$$

a) Escribir el predicado unario y dar el esquema de inducción a utilizar.
b) Plantear el/los caso/s base y resolverlo/s, justificando cada paso de la demostración.
c) Plantear el/los paso/s inductivo/s, marcando claramente la hipótesis, tesis inductiva y el alcance de los cuantificadores. Resolver justificando cada paso de la demostración.

Ej. 3. Diseño

Considerar la siguiente especificación que modela el programa de TV Insoportables. Cada tanto se agregan famosos a los considerados por el programa y se registran las peleas y reconciliaciones entre famosos registrados.

Para cada famoso se tiene en cuenta con qué otros famosos está peleado actualmente y cuantas veces peleó en total.

TAD INSOPORTABLES

observadores básicos

famosos : programa $\longrightarrow$ conj(famoso)

enemigos : programa p × famoso f $\longrightarrow$ conj(famoso) $\{f \in \text{famosos}(p)\}$

#peleas : programa p × famoso f $\longrightarrow$ nat $\{f \in \text{famosos}(p)\}$

generadores

salirAlAire : $\longrightarrow$ programa

nuevoFamoso : programa p × famoso f $\longrightarrow$ programa $\{f \notin \text{famosos}(p)\}$

pelear : programa p × famoso f × famoso f' $\longrightarrow$ programa
$\{\{f, f'\} \subseteq \text{famosos}(p) \wedge_L f \neq f' \wedge f \notin \text{enemigos}(p, f')\}$

reconciliar : programa p × famoso f × famoso f' $\longrightarrow$ programa
$\{\{f, f'\} \subseteq \text{famosos}(p) \wedge_L f \in \text{enemigos}(p, f')\}$

axiomas ...

famosos(salirAlAire) $\equiv \emptyset$

famosos(nuevoFamoso(p, f)) $\equiv$ Ag(p, f)

famosos(pelear(p, f, f')) $\equiv$ famosos(p)

famosos(reconciliar(p, f, f')) $\equiv$ famosos(p)

enemigos(nuevoFamoso(p, g), f) $\equiv$ **if** $g = f$ **then** $\emptyset$ **else** enemigos(p, f) **fi**

enemigos(pelear(p, g, g'), f) $\equiv$ **if** $f \in \{g, g'\}$ **then** $\{g, g'\} \setminus \{f\}$ **else** $\emptyset$ **fi** $\cup$ enemigos(p, f)

enemigos(reconciliar(p, g, g'), f) $\equiv$ enemigos(p, f) $\setminus$ **if** $f \in \{g, g'\}$ **then** $\{g, g'\}$ **else** $\emptyset$ **fi**

#peleas(nuevoFamoso(p, g), f) $\equiv$ **if** $g = f$ **then** 0 **else** #peleas(p, f) **fi**

#peleas(pelear(p, g, g'), f) $\equiv$ **if** $f \in \{g, g'\}$ **then** 1 **else** 0 **fi** + #peleas(p, f)

#peleas(reconciliar(p, g, g'), f) $\equiv$ #peleas(p, f)

Fin TAD

Se decidió utilizar la siguiente estructura para representar el TAD.

insoportables **se representa con** estr, donde

estr es tupla ⟨*enemigosActuales*: dicc(famoso,conj(famoso)),
por#Peleas: dicc(nat,conj(famoso)),
historialPeleas: dicc(famoso,secu(famoso))⟩

donde *enemigosAcutales* dice para cada famoso su conjunto de enemigos (posiblemente vacío), *historialPeleas* dice para cada famoso la secuencia de rivales con los cuales se peleó (posiblemente varias veces con el mismo, sin registrar las reconciliaciones) y *por#Peleas* dice para cada entero n el conjunto de famosos que participó en exactamente n peleas. *por#Peleas* solo tiene como claves los naturales n para los cuales al menos un famoso tiene exactamente n peleas en su haber.

a) Escribir en castellano el invariante de representación.
b) Escribir formalmente el invariante de representación.
c) Escribir formalmente la función de abstracción.