# Algoritmos y Estructuras de Datos II

# Trabajo Práctico 1

Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires

Segundo Cuatrimestre de 2012

# Grupo 9

Apellido y Nombre	LU	E-mail	
María Candela Capra Coarasa	234/11	canduh_27@hotmail.com	
Leandro Lovisolo	645/11	leandro@leandro.me	
Gastón de Orta	244/11	gaston.deorta@hotmail.com	
Lautaro José Petaccio	443/11	lausuper@gmail.com	

#### Reservado para la cátedra

Instancia	Docente que corrigió	Calificación
Primera Entrega		
Recuperatorio		

## Índice

1.	TAD	CATEGORIA	2
2.	TAD I	Link	2
3.	TAD I	DÍA	2
4.	TAD .	ÁRBOLDECATEGORÍAS	2
<b>5.</b>	TAD I	LinkLinkIt	3

#### 1. TAD CATEGORÍA

#### TAD CATEGORÍA es STRING

#### 2. TAD LINK

TAD LINK es STRING

#### 3. TAD DÍA

TAD DÍA es NAT

### 4. TAD ÁRBOLDECATEGORÍAS

### TAD ÁRBOLDECATEGORÍAS

buscarNodo(c, a)

```
géneros
               árbolDeCategorías
               árbolDeCategorías, generadores, observadores básicos, contiene?, buscarNodo, aplanar
exporta
               BOOL, NAT, CATEGORÍA, CONJUNTO(CATEGORÍA), CONJUNTO(ÁRBOLDECATEGORÍAS),
usa
               SECUENCIA (CATEGORÍA), SECUENCIA (ÁRBOLDECATEGORÍAS)
igualdad observacional
               (\forall a, a' : \text{árbolDeCategorías}) (a =_{\text{obs}} a' \iff (\text{raíz}(a) =_{\text{obs}} \text{raíz}(a') \land \text{hijos}(a) =_{\text{obs}} \text{hijos}(a')))
generadores
                               : categoría c \times \text{conj}(\text{árbolDeCategorías}) as \longrightarrow árbolDeCategorías
  árbol
                   \{(\forall c': \text{categoria}) \ (\#\text{Ocurrencias}(c', c \bullet \text{aplanarSecuencia}(\text{transformarEnSecuencia}(as))) \le 1\} \}
observadores básicos
  raíz
                               : árbolDeCategorías
                                                                                   → categoría
  hijos
                               : árbolDeCategorías
                                                                                   → conj(árbolDeCategorías)
otras operaciones
  contiene?
                               : categoría \times árbol
De<br/>Categorías
                                                                                   \longrightarrow bool
  buscarNodo
                               : categoría c \times árbol
De<br/>Categorías a
                                                                                   → árbolDeCategorías
                                                                                                           \{\text{contiene}?(c, a)\}
  buscarNodoEnConjunto : categoría × conj(árbolDeCategorías)
                                                                                  → árbolDeCategorías
  aplanar
                               : árbolDeCategorías
                                                                                   → conj(categoría)
  aplanarConjunto
                               : conj(árbolDeCategorías)
                                                                                  → conj(categoría)
                                                                                  → secu(árbolDeCategorías)
  transformarEnSecuencia : conj(árbolDeCategorías)
                                                                                  → secu(categoría)
  aplanarSecuencia
                               : secu(árbolDeCategorías)
  \#Ocurrencias
                               : categoría × secu(categoría)
                                                                                  \longrightarrow nat
               \forall c: categoría, \forall a: árbolDeCategorías, \forall as: conj(árbolDeCategorías),
axiomas
               \forall scs: secu(categoría), \forall sas: secu(árbolDeCategorías)
  raíz(árbol(c, as))
                                       \equiv c
  hijos(árbol(c, as))
                                       \equiv as
  contiene?(c, a)
                                       \equiv c \in \operatorname{aplanar}(a)
```

 $\equiv$  buscarNodoEnConjunto(c, Ag(a,  $\emptyset$ ))

```
buscarNodoEnConjunto(c, as) \equiv \mathbf{if} \ c = \text{raiz}(\text{dameUno}(as)) then
                                           dameUno(as)
                                       else
                                           buscarNodoEnConjunto(c, hijos(dameUno(as)) \cup sinUno(as))
aplanar(a)
                                    \equiv aplanarConjunto(Ag(a, \emptyset))
aplanarConjunto(as)
                                    \equiv if \emptyset?(as) then
                                       else
                                           Ag(raiz(dameUno(as)),
                                              \operatorname{aplanarConjunto(hijos(dameUno(as))} \cup \sinUno(as)))
transformarEnSecuencia(as)
                                    \equiv if \emptyset?(as) then
                                           <>
                                       else
                                           dameUno(as) \bullet transformarEnSecuencia(sinUno(as))
aplanarSecuencia(sas)
                                    \equiv if vacía?(sas) then
                                           <>
                                       else
                                           raíz(prim(sas)) \bullet aplanarSecuencia(
                                              transformarEnSecuencia(hijos(prim(sas))) \& fin(sas))
\#Ocurrencias(c, scs)
                                    \equiv if vacía?(scs) then
                                       else
                                           (if c = \text{prim}(scs) then 1 else 0 fi) + #Ocurrencias(c, fin(scs))
```

Fin TAD

#### 5. TAD LINKLINKIT

#### TAD LINKLINKIT

géneros linkLinkIt

exporta linkLinkIt, generadores, observadores básicos, #LinksEnCategoría, #AccesosMásRecientes

usa Bool, Nat, Categoría, Link, Día, ÁrbolDeCategorías, Conjunto(Categoría),

CONJUNTO(LINK)

#### igualdad observacional

```
 (\forall lli, lli': linkLinkIt) \left( \begin{aligned} & \operatorname{dar \acute{A}rbolDeCategor \acute{a}s}(lli) =_{\mathrm{obs}} \\ & \operatorname{dar \acute{A}rbolDeCategor \acute{a}s}(lli') \wedge \\ & \operatorname{links}(lli) =_{\mathrm{obs}} \operatorname{links}(lli') \wedge_{\mathrm{L}} \\ & (\forall l: link) \; (l \in \operatorname{links}(lli) \Rightarrow_{\mathrm{L}} \\ & (\operatorname{categor \acute{a}Link}(l, lli) =_{\mathrm{obs}} \\ & \operatorname{categor \acute{a}Link}(l, lli') \wedge \\ & (\forall d: \operatorname{d\'{a}}) \; (\#\operatorname{AccesosPorD \acute{a}}(l, d, lli) =_{\mathrm{obs}} \\ & \#\operatorname{AccesosPorD \acute{a}}(l, d, lli')))) \end{aligned} \right)
```

#### generadores

iniciar : árbol De<br/>Categorías  $\longrightarrow$  link Link It registrar Link : link <br/>  $l \times$  categoría  $c \times$  link Link It <br/>  $lli \longrightarrow$  link Link It

 $\{\neg(l \in links(lli)) \land contiene?(c, darÁrbolDeCategorías(lli))\}$ 

registrar Acceso : link  $l \times \text{día } d \times \text{linkLinkIt}$   $lii \longrightarrow \text{linkLinkIt}$ 

```
\{l \in links(lli) \land d \ge dia\'UltimoAcceso(lli)\}
observadores básicos
  darÁrbolDeCategorías : linkLinkIt
                                                                                  → árbolDeCategorías
  links
                               : linkLinkIt
                                                                                  \rightarrow \text{conj(link)}
                               : link l \times linkLinkIt lli
  categoríaLink
                                                                                  → categoría
                                                                                                                      \{l \in links(lli)\}
  #AccesosPorDía
                               : link l \times \text{día } d \times \text{linkLinkIt } lli
                                                                                    nat
                                                                                                                      \{l \in links(lli)\}
otras operaciones
  díaÚltimoAcceso
                                            : linkLinkIt
                                                                                                                 \rightarrow nat
  #LinksEnCategoría
                                            : categoría c \times linkLinkIt lli
                                                                                                               \rightarrow nat
                                                                                      \{\text{contiene?}(c, \text{dar}\text{\'ArbolDeCategor\'ias}(lli)\}
  filtrar
YContar<br/>Links
PorCategoría : conj(links) ls \times conj(categoría) cs \times link<br/>LinkIt lli \longrightarrow bool
                                                                                                                    \{ls \subseteq links(lli)\}
                                            : link l \times linkLinkIt lli
                                                                                                              \longrightarrow nat
  #AccesosMásRecientes
                                                                                                                      \{l \in links(lli)\}
                \forall c: categoría, \forall cs: conj(categoría), \forall l, l': link, \forall ls: conj(link), \forall d, d': día,
axiomas
                \forall a: \text{ arbolDeCategorias}, \forall lli: \text{linkLinkIt}
  dar Arbol De Categorías (iniciar(a))
  dar Arbol De Categorías (registrar Link (l, c, lli))
                                                               \equiv \operatorname{dar} ArbolDeCategorias(lli)
  darÁrbolDeCategorías(registrarAcceso(l, d, lli))
                                                               \equiv \operatorname{darArbolDeCategorias}(lli)
  links(iniciar(a))
  links(registrarLink(l, c, lli))
                                                               \equiv Ag(l, links(lli))
  links(registrarAcceso(l, d, lli))
                                                               \equiv links(lli)
  categoríaLink(l, registrarLink(l', c, lli))
                                                               \equiv if l = l' then c else categoríaLink(l, lli) fi
  categoríaLink(l, registrarAcceso(l', d, lli))
                                                               \equiv categoríaLink(l, lli)
  \#AccesosPorDía(l, d, iniciar(a))
                                                               \equiv 0
  \#AccesosPorDía(l, d, registrarLink(l', c, lli))
                                                               \equiv \#AccesosPorDía(l, d, lli)
  \#AccesosPorDía(l, d, registrarAcceso(l', d', lli))
                                                               \equiv (if l = l' \wedge d = d' then 1 else 0 fi) +
                                                                       \#AccesosPorDía(l, d, lli)
  diaUltimoAcceso(iniciar(a))
                                                               \equiv 0
  dia\dot{U}ltimoAcceso(registrarLink(l, c, lli))
                                                               \equiv díaÚltimoAcceso(lli)
  diaUltimoAcceso(registrarAcceso(l, d, lli))
  \#LinksEnCategoría(c, lli)
                                                               ≡ filtrarYContarLinksPorCategoría(
                                                                       aplanar(buscarNodo(c, darÁrbolDeCategorías(lli))),
                                                                       lli)
  filtrarYContarLinksPorCategoría(ls, cs, lli)
                                                               \equiv if \emptyset?(ls) then
                                                                       0
                                                                   else
                                                                       (if categoríaLink(dameUno(ls), lli) \in cs then
                                                                       else
                                                                           0
                                                                       \mathbf{fi}) +
                                                                       filtrarYContarLinksPorCategoría(sinUno(ls), cs, lli)
                                                                   fi
```

```
\# Accesos M \'as Recientes(l, lli) \\ \equiv \# Accesos Por D\'ia(l, d\'ia\'Ultimo Acceso(lli), lli) + \\ (\textbf{if } d\'ia\'Ultimo Acceso(lli) \geq 1 \textbf{ then} \\ \# Accesos Por D\'ia(l, d\'ia\'Ultimo Acceso(lli) - 1, lli) \\ \textbf{else} \\ 0 \\ \textbf{fi}) + \\ (\textbf{if } d\'ia\'Ultimo Acceso(lli) \geq 2 \textbf{ then} \\ \# Accesos Por D\'ia(l, d\'ia\'Ultimo Acceso(lli) - 2, lli) \\ \textbf{else} \\ 0 \\ \textbf{fi}) \\ \end{pmatrix}
```

Fin TAD