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
5.	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

 $\textbf{exporta} \qquad \text{linkLinkIt, generadores, observadores básicos, } \# Links En Categoría, \# Accesos Más Recientes$

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

CONJUNTO(LINK)

igualdad observacional

```
 \left( \forall lli, lli': \text{linkLinkIt} \right) \left( \begin{aligned} &\text{darÁrbolDeCategorías}(lli) =_{\text{obs}} \\ &\text{darÁrbolDeCategorías}(lli') \land \\ &\text{links}(lli) =_{\text{obs}} \text{links}(lli') \land_{\text{L}} \\ &(\forall l: \text{link}) \; (l \in \text{links}(lli) \Rightarrow_{\text{L}} \\ &(\text{categoríaLink}(l, lli) =_{\text{obs}} \\ &\text{categoríaLink}(l, lli') \land \\ &(\forall d: \text{día}) \; (d \in \text{díasAccesosRecientes}(lli) \Rightarrow_{\text{L}} \\ &(\text{\#AccesosPorDía}(l, d, lli') =_{\text{obs}} \\ &\#\text{AccesosPorDía}(l, d, lli'))))) \right)
```

generadores

iniciar : árbol DeCategorías \longrightarrow linkLinkIt registrar Link : link $l \times$ categoría $c \times$ linkLinkIt $lli \longrightarrow$ linkLinkIt $\{\neg(l \in links(lli)) \land contiene?(c, darÁrbol$ $DeCategorías(lli))\}$

```
registrarAcceso
                                : link l \times día d \times linkLinkIt lli
                                                                                   \longrightarrow linkLinkIt
                                                                                      \{l \in links(lli) \land d \ge diaÚltimoAcceso(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
                                                                                    \rightarrow nat
                                                                                 \{l \in links(lli) \land d \in diasAccesosRecientes(lli)\}
otras operaciones
  díaÚltimoAcceso
                                             : linkLinkIt
                                                                                                                  \longrightarrow día
   díasAccesosRecientes
                                             : linkLinkIt
                                                                                                                  \longrightarrow \text{conj}(\text{día})
   #LinksEnCategoría
                                             : categoría c \times \text{linkLinkIt } lli
                                                                                                                  \longrightarrow nat
                                                                                         \{\text{contiene?}(c, \text{darArbolDeCategorias}(lli)\}
  filtrarYContarLinksPorCategoría : conj(links) ls \times \text{conj}(\text{categoría}) \ cs \times \text{linkLinkIt} \ lli \longrightarrow \text{bool}
                                                                                                                        \{ls \subseteq links(lli)\}
   #AccesosMásRecientes
                                             : link l \times linkLinkIt lli
                                                                                                                    \rightarrow nat
                                                                                                                          \{l \in links(lli)\}
axiomas
                 \forall c: categoría, \forall cs: conj(categoría), \forall l, l': link, \forall ls: conj(link), \forall d, d': día,
                 \forall a: \text{ \'arbolDeCategor\'as}, \forall lli: linkLinkIt}
  dar Arbol De Categorías (iniciar(a))
                                                                 \equiv a
  darÁrbolDeCategorías(registrarLink(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)
                                                                 \equiv 0
   \#AccesosPorDía(l, d, iniciar(a))
   \#AccesosPorDía(l, d, registrarLink(l', c, lli))
                                                                 \equiv \# AccesosPorDía(l, d, lli)
                                                                 \equiv (if l = l' \wedge d = d' then 1 else 0 fi) +
   \#AccesosPorDía(l, d, registrarAcceso(l', d', lli))
                                                                         \#AccesosPorDía(l, d, lli)
  diaUltimoAcceso(iniciar(a))
                                                                 \equiv 0
  diaUltimoAcceso(registrarLink(l, c, lli))
                                                                 \equiv díaÚltimoAcceso(lli)
  diaUltimoAcceso(registrarAcceso(l, d, lli))
                                                                 \equiv d
  díasAccesosRecientes(iniciar(a))
                                                                 \equiv \emptyset
  díasAccesosRecientes(registrarLink(l, c, lli))
                                                                 \equiv díasAccesosRecientes(lli)
  díasAccesosRecientes(registrarAcceso(l, d, lli))
                                                                 \equiv if d=0 then
                                                                         Ag(0, \emptyset)
                                                                     else
                                                                         (if d = 1 then
                                                                             Ag(1, Ag(0, \emptyset))
                                                                         else
                                                                              Ag(d, Ag(d-1, Ag(d-2, \emptyset)))
                                                                         fi)
                                                                     fi
```

```
\#LinksEnCategoría(c, lli)
                                                        ≡ filtrarYContarLinksPorCategoría(
                                                               links(lli),
                                                               aplanar(buscarNodo(c, darÁrbolDeCategorías(lli))),
                                                               lli)
filtrar
Y<br/>Contar
Links
Por
Categoría<br/>(ls,\ cs,\ lli)
                                                        \equiv if \emptyset?(ls) then
                                                               0
                                                           else
                                                               (if categoríaLink(dameUno(ls), lli) \in cs then
                                                               \mathbf{else}
                                                               fi) +
                                                               filtrarYContarLinksPorCategoría(sinUno(ls), cs, lli)
                                                        \equiv \# AccesosPorDía(l, díaÚltimoAcceso(lli), lli) +
\#AccesosMásRecientes(l, lli)
                                                           (if díaÚltimoAcceso(lli) \geq 1 then
                                                               #Accesos
Por<br/>Día(l, día
Último
Acceso(lli) - 1, lli)
                                                           else
                                                               0
                                                           \mathbf{fi}) +
                                                           (if díaÚltimoAcceso(lli) \geq 2 then
                                                               \#AccesosPorDía(l, díaÚltimoAcceso(lli) - 2, lli)
                                                           else
                                                           fi
```

Fin TAD