

RECURSIÓN ULTIMATE

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

Algoritmos y Estructuras de Datos II

TAD DICCIONARIO(κ, σ)

géneros $\text{dicc}(\kappa, \sigma)$

observadores básicos

$\text{def?} \quad : \kappa \times \text{dicc}(\kappa, \sigma) \longrightarrow \text{bool}$

$\text{obtener} \quad : \kappa \times \text{dicc}(\kappa, \sigma) \times d \longrightarrow \sigma \quad \{ \text{def?}(c, d) \}$

generadores

$\text{vacío} \quad : \longrightarrow \text{dicc}(\kappa, \sigma)$

$\text{definir} \quad : \kappa \times \sigma \times \text{dicc}(\kappa, \sigma) \longrightarrow \text{dicc}(\kappa, \sigma)$

otras operaciones

$\text{borrar} \quad : \kappa \times \text{dicc}(\kappa, \sigma) \times d \longrightarrow \text{dicc}(\kappa, \sigma) \quad \{ \text{def?}(c, d) \}$

$\text{claves} \quad : \text{dicc}(\kappa, \sigma) \longrightarrow \text{conj}(\kappa)$

axiomas $\forall d: \text{dicc}(\kappa, \sigma), \forall c, k: \kappa, \forall s: \sigma$

$\text{def?}(c, \text{vacío}) \equiv \text{false}$

$\text{def?}(c, \text{definir}(k, s, d)) \equiv c = k \vee \text{def?}(c, d)$

$\text{obtener}(c, \text{definir}(k, s, d)) \equiv \text{if } c = k \text{ then } s \text{ else obtener}(c, d) \text{ fi}$

Fin TAD

EJERCICIO 1

Dados dos diccionarios donde el significado de uno es la clave del otro, crear un nuevo diccionario que tenga las claves del primero y los significados del segundo, estando estos relacionados por el significado/clave en común.

$$\text{juntar} : \text{dicc}(k \times s_1) d_1 \times \text{dicc}(s_1 \times s_2) \longrightarrow \text{dicc}(k, s_2)$$

$$\{(\forall c: k) \text{ def?}(c, d_1) \Rightarrow_L \text{ def?}(\text{obtener}(c, d_1), d_2)\}$$

TAD DICCIONARIO

observadores básicos

$$\text{def?} : \kappa \times \text{dicc}(\kappa, \sigma) \longrightarrow \text{bool}$$

$$\text{obtener} : \kappa \times \text{dicc}(\kappa, \sigma) \longrightarrow \sigma \quad \{\text{def?}(c, d)\}$$

generadores

$$\text{vacío} : \longrightarrow \text{dicc}(\kappa, \sigma)$$

$$\text{definir} : \kappa \times \sigma \times \text{dicc}(\kappa, \sigma) \longrightarrow \text{dicc}(\kappa, \sigma)$$

otras operaciones

$$\text{borrar} : \kappa \times \text{dicc}(\kappa, \sigma) \longrightarrow \text{dicc}(\kappa, \sigma) \quad \{\text{def?}(c, d)\}$$

$$\text{claves} : \text{dicc}(\kappa, \sigma) \longrightarrow \text{conj}(\kappa)$$

$\text{juntar} : \text{dicc}(k \times s_1)d_1 \times \text{dicc}(s_1 \times s_2) \longrightarrow \text{dicc}(k, s_2)$

$\{(\forall c: k) \text{ def?}(c, d_1) \Rightarrow_L \text{def?}(\text{obtener}(c, d_1), d_2)\}$

```
juntar( $d_1, d_2$ )  $\equiv$  if  $\emptyset?$ (claves( $d_1$ )) then
    vacío
else
    definir(
        dameUno(claves( $d_1$ )),
        obtener(obtener(dameUno(claves( $d_1$ )),  $d_1$ ),  $d_2$ ),
        juntar(borrar(dameUno(claves( $d_1$ )),  $d_1$ ),  $d_2$ )
    )
fi
```

EJERCICIO 2

Dado el TAD CARALIBRO que representa una red social, se nos pide axiomatizar la función *islaDeAmigos*(u, c) que dado un miembro de la red social nos devuelva el conjunto de usuarios que son alcanzables con la relación de amistad.

TAD CARALIBRO

géneros cl

observadores básicos

$miembros : cl \longrightarrow conj(usuario)$

$amigos : usuario\ u \times cl\ c \longrightarrow conj(usuario)$

$\{u \in miembros(cl)\}$

otras operaciones

$islaDeAmigos : usuario\ u \times cl\ c \longrightarrow conj(usuario)$

$\{u \in miembros(c)\}$

Fin TAD

EJERCICIO 2

islaDeAmigos : usuario $u \times$ cl $c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us} \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$

$\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{amigosDeAmigos}(\text{us}, c) \equiv \text{if } \emptyset?(us) \text{ then}$
 \emptyset
 else
 $\text{amigos}(\text{dameUno}(\text{us}), c) \cup \{\text{dameUno}(\text{us})\} \cup$
 $\text{amigosDeAmigos}(\text{sinUno}(\text{us}), c)$
 fi

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuario})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuario})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos}(\text{us}, c) \equiv$ **if** ?? **then**
 us
 else
 islaDeVariosAmigos(amigosDeAmigos(us, c),
 c)
 fi

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuario})$
 $\{\text{us} \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos}(\text{us}, c) \equiv \text{if } ?? \text{ then}$
 us
 else
 $\text{islaDeVariosAmigos}(\text{amigosDeAmigos}(\text{us}, c),$
 $c)$
 fi

$\text{islaDeAmigos}(u, c) \equiv \text{islaDeVariosAmigos}(\{u\}, c)$

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{us \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos} : \text{conj}(\text{usuario}) \text{ us } \times \text{cl } c \longrightarrow \text{conj}(\text{usuario})$
 $\{us \subseteq \text{miembros}(c)\}$
 $\text{islaDeVariosAmigos}(us, c) \equiv \text{if } \text{amigosDeAmigos}(us, c) \subseteq us \text{ then}$
 us
 else
 $\text{islaDeVariosAmigos}(\text{amigosDeAmigos}(us, c),$
 $c)$
 fi

EJERCICIO 2

$\text{islaDeAmigos} : \text{usuario } u \times \text{cl } c \longrightarrow \text{conj}(\text{usuario}) \quad \{u \in \text{miembros}(c)\}$

$\text{amigosDeAmigos} : \text{conj}(\text{usuario}) \text{ } us \times \text{cl } c \longrightarrow \text{conj}(\text{usuarios})$
 $\{us \subseteq \text{miembros}(c)\}$

$\text{islaDeVariosAmigos} : \text{conj}(\text{usuario}) \text{ } us \times \text{cl } c \longrightarrow \text{conj}(\text{usuario})$
 $\{us \subseteq \text{miembros}(c)\}$

$\text{islaDeAmigos}(u, c) \equiv \text{islaDeVariosAmigos}(\{u\}, c)$

EJERCICIO 3

Ahora que sabemos cómo se componen las islas de amigos, nos piden especificar todas las cadenas de menor cantidad de saltos entre dos usuarios en una misma isla. Esto es, aquellas cadenas de amistad entre dos usuarios con largo equivalente al grado de separación entre ellos (Ver el *numero de bacon*).

EJERCICIO 3

Ahora que sabemos cómo se componen las islas de amigos, nos piden especificar todas las cadenas de menor cantidad de saltos entre dos usuarios en una misma isla. Esto es, aquellas cadenas de amistad entre dos usuarios con largo equivalente al grado de separación entre ellos (Ver el *numero de bacon*).



EJERCICIO 3

gradosDeSeparación : usuario $a \times$ usuario $b \times$ cl $c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\{\{a, b\} \subseteq \text{miembros}(c)\}$

EJERCICIO 3

$\text{gradosDeSeparación} : \text{usuario } a \times \text{usuario } b \times \text{cl } c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\{\{a, b\} \subseteq \text{miembros}(c)\}$

$\text{seguirCadenaA} : \text{secu}(\text{usuario}) s \times \text{usuario } b \times \text{cl } c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\{\neg \emptyset?(s) \wedge_{\text{L}} \text{ult}(s) \in \text{miembros}(c) \wedge b \in \text{miembros}(c)\}$

EJERCICIO 3

$\text{gradosDeSeparación} : \text{usuario } a \times \text{usuario } b \times \text{cl } c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\{\{a, b\} \subseteq \text{miembros}(c)\}$

$\text{seguirCadenaA} : \text{secu}(\text{usuario}) s \times \text{usuario } b \times \text{cl } c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\{\neg \emptyset?(s) \wedge_L \text{ult}(s) \in \text{miembros}(c) \wedge b \in \text{miembros}(c)\}$

$\text{cadenasExtendidas} : \text{secu}(\text{usuario}) s \times \text{conj}(\text{usuario}) us \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\times \text{usuario } b \times \text{cl } c$
 $\{b \in \text{miembros}(c) \wedge us \subseteq \text{miembros}(c)\}$

EJERCICIO 3

```
seguirCadenaA : secu(usuario)  $s \times$  usuario  $b \times$  cl  $c \longrightarrow$  conj(secu(usuario))  
seguirCadenaA(s, b, c)  $\equiv$  if ult(s) = b then  
    {s}  
    else  
        cadenasExtendidas(s, amigos(ult(s), c), b, c)  
    fi
```

EJERCICIO 3

```
seguirCadenaA : secu(usuario)  $s \times$  usuario  $b \times$  cl  $c \longrightarrow$  conj(secu(usuario))
seguirCadenaA(s, b, c)  $\equiv$  if ult(s) = b then
                        {s}
                        else
                        cadenasExtendidas(s, amigos(ult(s), c), b, c)
                        fi
```

```
cadenasExtendidas : secu(usuario)  $s \times$  conj(usuario)  $us \longrightarrow$  conj(secu(usuario))
                   $\times$  usuario  $b \times$  cl  $c$ 
```

EJERCICIO 3

```
seguirCadenaA : secu(usuario)  $s \times$  usuario  $b \times$  cl  $c \longrightarrow$  conj(secu(usuario))  
seguirCadenaA(s, b, c)  $\equiv$  if ult(s) = b then  
    {s}  
    else  
        cadenasExtendidas(s, amigos(ult(s), c), b, c)  
    fi
```

```
cadenasExtendidas : secu(usuario)  $s \times$  conj(usuario)  $us \longrightarrow$  conj(secu(usuario))  
     $\times$  usuario  $b \times$  cl  $c$   
cadenasExtendidas(s,us,b,c)  $\equiv$  if  $\emptyset?(us)$  then  
     $\emptyset$   
    else  
        if  $\neg$  está?(dameUno(us), s) then  
            seguirCadenaA(s  $\circ$  dameUno(us), b, c)  
             $\cup$  cadenasExtendidas(s, sinUno(us), b,  
                c)  
        else  
            cadenasExtendidas(s, sinUno(us), b, c)  
        fi  
    fi
```

EJERCICIO 3

$\text{seguirCadenaA} : \text{secu}(\text{usuario})\ s \times \text{usuario}\ b \times \text{cl}\ c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$

$$\{\neg \emptyset?(s) \wedge_{\text{L}} \text{ult}(s) \in \text{miembros}(c) \wedge b \in \text{miembros}(c)\}$$

$\text{cadenasExtendidas} : \text{secu}(\text{usuario})\ s \times \text{conj}(\text{usuario})\ us \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\times \text{usuario}\ b \times \text{cl}\ c$

$$\{b \in \text{miembros}(c) \wedge us \subseteq \text{miembros}(c)\}$$

$\text{gradosDeSeparación} : \text{usuario}\ a \times \text{usuario}\ b \times \text{cl}\ c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$

$$\{\{a, b\} \subseteq \text{miembros}(c)\}$$

EJERCICIO 3

$\text{seguirCadenaA} : \text{secu}(\text{usuario})\ s \times \text{usuario}\ b \times \text{cl}\ c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$

$$\{\neg \emptyset?(s) \wedge_{\text{L}} \text{ult}(s) \in \text{miembros}(c) \wedge b \in \text{miembros}(c)\}$$

$\text{cadenasExtendidas} : \text{secu}(\text{usuario})\ s \times \text{conj}(\text{usuario})\ us \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$
 $\times \text{usuario}\ b \times \text{cl}\ c$

$$\{b \in \text{miembros}(c) \wedge us \subseteq \text{miembros}(c)\}$$

$\text{gradosDeSeparación} : \text{usuario}\ a \times \text{usuario}\ b \times \text{cl}\ c \longrightarrow \text{conj}(\text{secu}(\text{usuario}))$

$$\{\{a, b\} \subseteq \text{miembros}(c)\}$$

$\text{gradosDeSeparación}(a, b, c) \equiv \text{minLargo}(\text{seguirCadenaA}(a \bullet \langle \rangle, b, c))$