

Trabajo Práctico Número 2

Algoritmos y Estructuras de Datos II

Grupo: 21

Integrante	LU	Correo electrónico
Langberg, Andrés	249/14	andreslangberg@gmail.com
Walter, Nicolás	272/14	nicowalter25@gmail.com
Sticco, Patricio Bernardo	337/14	pbsticco@hotmail.com
Len, Julián	467/14	julianlen@gmail.com



Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2160 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (54 11) 4576-3359

<http://www.fcen.uba.ar>

1. Diseño del Tipo RASTRILLAJE

1.1. Especificación

Se usa el TAD CAMPUSSEGURO especificado por la cátedra.

1.2. Aspectos de la interfaz

1.2.1. Interfaz

Se explica con especificación de CAMPUSSEGURO

Género *rastr*

Operaciones básicas de Rastrillaje

CAMPUS(**in** *r: rastr*) \rightarrow *res: campus*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} campus(r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el campus.

ESTUDIANTES(**in** *r: rastr*) \rightarrow *res: conj(nombre)*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} estudiantes(r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el conjunto de estudiantes presentes en el campus.

HIPPIES(**in** *r: rastr*) \rightarrow *res: conj(nombre)*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} hippies(r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el conjunto de hippies presentes en el campus.

AGENTES(**in** *r: rastr*) \rightarrow *res: conj(agente)*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} agentes(r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el conjunto de agentes presentes en el campus.

POSESTUDIANTEYHIPPIE(**in** *r: rastr*, **in** *id: nombre*) \rightarrow *res: posicion*

Pre $\equiv \{ id \in (estudiantes(r) \cup hippies(cs)) \}$

Post $\equiv \{ res =_{\text{obs}} posEstudianteYHippie(id, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve la posición del estudiante/hippie pasado como parámetro.

POSAGENTE(**in** *r: rastr*, **in** *a: agente*) \rightarrow *res: posicion*

Pre $\equiv \{ a \in posAgente(a, r) \}$

Post $\equiv \{ res =_{\text{obs}} posAgente(a, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve la posición del agente pasado como parámetro.

CANTSANCIONES(**in** *r: rastr*, **in** *a: agente*) \rightarrow *res: nat*

Pre $\equiv \{ a \in cantSanciones(a, r) \}$

Post $\equiv \{ res =_{\text{obs}} cantSanciones(a, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve la cantidad de sanciones recibidas por el agente pasado como parámetro.

CANTHIPPIESATRAPADOS(**in** *r: rastr*, **in** *a: agente*) \rightarrow *res: nat*

Pre $\equiv \{ a \in agentes(r) \}$

Post $\equiv \{ res =_{\text{obs}} \text{cantHippiesAtrapados}(a, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve la cantidad de hippies atrapados por el agente pasado como parámetro.

COMENZARRASTRILLAJE(**in** c : *campus*, **in** d : *dicc(agente, posicion)*) $\longrightarrow res$: *rastr*

Pre $\equiv \{ (\forall a : \text{agente})(\text{def?}(a, d) \Rightarrow_L (\text{posValida?}(\text{obtener}(a, d))) \wedge \neg \text{ocupada?}(\text{obtener}(a, d), c)) \wedge (\forall a, a_2 : \text{agente})((\text{def?}(a, d) \wedge \text{def?}(a_2, d) \wedge a \neq a_2) \Rightarrow_L \text{obtener}(a, d) \neq \text{obtener}(a_2, d)) \}$

Post $\equiv \{ res =_{\text{obs}} \text{comenzarRastrillaje}(c, d) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Crea un Rastrillaje.

INGRESARESTUDIANTE(**in/out** r : *rastr*, **in** e : *nombre*, **in** p : *posicion*) \longrightarrow

Pre $\equiv \{ r = r_0 \wedge e \notin (\text{estudiantes}(r) \cup \text{hippies}(r)) \wedge \text{esIngreso?}(p, \text{campus}(r)) \wedge \neg \text{estaOcupada?}(p, r) \}$

Post $\equiv \{ r =_{\text{obs}} \text{ingresarEstudiante}(e, p, r_0) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Modifica el rastrillaje, ingresando un estudiante al campus.

INGRESARHIPPIE(**in/out** r : *rastr*, **in** h : *nombre*, **in** p : *posicion*) \longrightarrow

Pre $\equiv \{ r = r_0 \wedge h \notin (\text{estudiantes}(r) \cup \text{hippies}(r)) \wedge \text{esIngreso?}(p, \text{campus}(r)) \wedge \neg \text{estaOcupada?}(p, r) \}$

Post $\equiv \{ r =_{\text{obs}} \text{ingresarHippie}(h, p, r_0) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Modifica el rastrillaje, ingresando un hippie al campus.

MOVERESTUDIANTE(**in/out** r : *rastr*, **in** e : *nombre*, **in** dir : *direccion*) \longrightarrow

Pre $\equiv \{ r = r_0 \wedge e \in \text{estudiantes}(r) \wedge (\text{seRetira}(e, dir, r) \vee (\text{posValida?}(\text{proxPosicion}(\text{posEstudianteYHippie}(e, r), dir, \text{campus}(r)), \text{campus}(r)) \wedge \neg \text{estaOcupada?}(\text{proxPosicion}(\text{posEstudianteYHippie}(e, r), dir, \text{campus}(r)), r))) \}$

Post $\equiv \{ r =_{\text{obs}} \text{moverEstudiante}(e, d, r_0) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Modifica el rastrillaje, al mover un estudiante del campus.

MOVERHIPPIE(**in/out** r : *rastr*, **in** h : *nombre*) \longrightarrow

Pre $\equiv \{ r = r_0 \wedge h \in \text{hippies}(r) \wedge \neg \text{todasOcupadas?}(\text{vecinos}(\text{posEstudianteYHippie}(h, r), \text{campus}(r)), r) \}$

Post $\equiv \{ r =_{\text{obs}} \text{moverHippie}(r, r_0) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Modifica el rastrillaje, al mover un hippie del campus.

MOVERAGENTE(**in/out** r : *rastr*, **in** a : *agente*) \longrightarrow

Pre $\equiv \{ r = r_0 \wedge a \in \text{agentes}(r) \wedge_L \text{cantSanciones}(a, r) \leq 3 \wedge \neg \text{todasOcupadas?}(\text{vecinos}(\text{posAgente}(a, r), \text{campus}(r)), r) \}$

Post $\equiv \{ r =_{\text{obs}} \text{moverAgente}(a, r_0) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Modifica el rastrillaje, al mover un agente del campus.

MASVIGILANTE(**in** r : *rastr*) $\longrightarrow res$: *agente*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} \text{masVigilante}(r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el agente con mas capturas.

CONKSANCIONES(**in** r : *rastr*, **in** k : *nat*) $\longrightarrow res$: *conj(agente)*

Pre $\equiv \{ true \}$

Post $\equiv \{ res =_{\text{obs}} \text{conKSanciones}(k, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el agente con mas capturas.

CONMISMASANCIONES(**in** r : *rastr*, **in** a : *agente*) $\longrightarrow res$: *conj(agente)*

Pre $\equiv \{ a \in \text{agentes}(r) \}$

Post $\equiv \{ res =_{\text{obs}} \text{conMismasSanciones}(a, r) \}$

Complejidad: $\mathcal{O}(1)$

Descripción: Devuelve el conjunto de agentes con la misma cantidad de sanciones que a.

1.3. Pautas de implementación

1.3.1. Estructura de representación

campus se representa con *estr*

donde *estr* es

tupla(

campo: *campus* \times

agentes: *diccPromedio*(*agente* ; *datosAg*) \times

hippies: *conjLineal*(*datosHoE*) \times

estudiantes: *conjLineal*(*datosHoE*) \times

posCiviles: *diccString*(*nombre*; *posicion*) \times

posRapida: *diccLineal*(*nombre*; *posicion*) \times

quienOcupa: *vector*(*vector*(*datosPos*)) \times

masVigilante: *itConj*(*agente*) \times

agregoEn1: *lista*(*datosK*) \times

buscoEnLog: *vector*(*datosK*)

)

donde *datosAg* es

tupla(

QSanciones: *nat* \times

premios: *nat* \times

posActual: *posicion* \times

grupoSanciones: *itConj*(*agente*) \times

verK: *itLista*(*nat*)

)

donde *datosHoE* es

tupla(

ID: *nombre* \times

itDicc(*nombre*; *posicion*): *posActual*

)

donde *datosPos* es

tupla(

ocupada?: *bool* \times

queHay: *clases* \times

hayCana: *itDicc*(*agente*) \times

hayHoE: *itConj*(*nombre*)

)

donde *clases* es $\text{enum}\{ \text{"agente"}, \text{"estudiante"}, \text{"hippie"}, \text{"obstaculo"}, \text{"nada"} \}$

donde *datosK* es

tupla(

K: *nat* \times

grupoK: *conjLineal*(*agente*)

)

1.3.2. Justificación

1.3.3. Invariante de Representación**Informal**

1. El mapa debe tener tantas filas como indica la estructura, lo mismo con las columnas.

Formal

$\text{Rep} : \text{estr} \longrightarrow \text{boolean}$

$(\forall e : \text{estr})$

$\text{Rep}(e) \equiv (\text{true} \iff$

$(1) \text{ e.filas} = \text{longitud}(\text{e.mapa}) \wedge_L (\forall i : \text{nat})(i \leq \text{e.filas} \Rightarrow \text{longitud}(\text{e.mapa}[i]) = \text{e.columnas}))$

1.3.4. Función de Abstracción

$\text{Abs} : \text{estr } e \longrightarrow \text{campus}$

$\{\text{Rep}(e)\}$

$(\forall e : \text{estr}) \text{ Abs}(e) =_{\text{obs}} c : \text{campus} /$

$\left(\text{filas}(c) = \text{e.filas} \wedge \text{columnas}(c) = \text{e.columnas} \wedge_L (\forall p : \text{posicion})(p.X \leq \text{e.filas} \wedge \right.$
 $\left. p.Y \leq \text{e.columnas} \Rightarrow_L \text{ocupada?}(p, c) \Leftrightarrow (\text{e.mapa}[p.X])[p.Y]) \right)$

1.3.5. Algoritmos

```
1: function iCAMPUS(in e: estr)  $\longrightarrow$  res : campus  $\triangleright \mathcal{O}(1)$   
2:   res  $\leftarrow$  e.campo  
3: end function
```

```
1: function iESTUDIANTES(in e: estr)  $\longrightarrow$  res : itConj(nombre)  $\triangleright \mathcal{O}(1)$   
2:   res  $\leftarrow$  crearIt (e.estudiantes)  
3: end function
```

```
1: function iHIPPIES(in e: estr)  $\longrightarrow$  res : itConj(nombre)  $\triangleright \mathcal{O}(1)$   
2:   res  $\leftarrow$  crearIt (e.hippies)  
3: end function
```

```
1: function iAGENTES(in e: estr)  $\longrightarrow$  res : itConj(agente)  $\triangleright \mathcal{O}(1)$   
2:   res  $\leftarrow$  crearIt (e.agentes)  
3: end function
```

```
1: function iPOSESTUDIANTESYHIPPIE(in n: nombre in e: estr)  $\longrightarrow$  res : posicion  $\triangleright \mathcal{O}(n_m)$   
2:   res  $\leftarrow$  obtener(n,e.posCiviles)  
3: end function
```

```
1: function iPOSAGENTE(in a: agente in e: estr)  $\longrightarrow$  res : posicion  $\triangleright \mathcal{O}(1)(promedio)$   
2:   res  $\leftarrow$  obtener(a,e.agentes).posActual  
3: end function
```

```
1: function iCANTSANCIONES(in a: agente in e: estr)  $\longrightarrow$  res : nat  $\triangleright \mathcal{O}(1)(promedio)$   
2:   res  $\leftarrow$  obtener(a,e.agentes).Qsanciones  
3: end function
```

```
1: function iCANTHIPPIESATRAPADOS(in a: agente in e: estr)  $\longrightarrow$  res : nat  $\triangleright \mathcal{O}(1)(promedio)$   
2:   res  $\leftarrow$  obtener(a,e.agentes).premios  
3: end function
```
