

1. Módulo Campus

Interfaz

se explica con: CAMPUS

géneros: campus

Operaciones básicas de campus

NUEVOCAMPUS(**in** $al : \text{nat}$, **in** $an : \text{nat}$) $\rightarrow res : \text{campus}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{crearCampus}(al, an)\}$

Complejidad: $\Theta(1)$

Descripción: Crea un nuevo campus vacío de alto al x ancho an .

AGREGAROBSTACULO(**in** $p : \text{pos}$), **in/out** $c : \text{campus}$)

Pre $\equiv \{\text{posValida}(p, c) \wedge \neg \text{ocupada?}(p, c) \wedge c =_{\text{obs}} c_0\}$

Post $\equiv \{c =_{\text{obs}} \text{agregarObstaculo}(p, c_0)\}$

Descripción: Agrega un obstáculo al campus c en la posición p .

FILAS(**in** $c : \text{campus}$) $\rightarrow res : \text{nat}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{filas}(c)\}$

Descripción: Devuelve el alto (filas) del campus c .

COLUMNAS(**in** $c : \text{campus}$) $\rightarrow res : \text{nat}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{columnas}(c)\}$

Descripción: Devuelve el ancho (columnas) del campus c .

OCUPADA?(**in** $p : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{bool}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{posValida?}(p, c)\}$

Descripción: Devuelve *true* si la posición p es válida en el campus c , sino retorna *false*.

ESINGRESO?(**in** $p : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{bool}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{esIngreso?}(p, c)\}$

Descripción: Verifica si la posición p es una entrada del campus c .

INGRESOSUPERIOR?(**in** $p : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{bool}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{ingresoSuperior?}(p, c)\}$

Descripción: Verifica si la posición p es una entrada superior del campus c .

INGRESOINFERIOR?(**in** $p : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{bool}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{ingresoInferior?}(p, c)\}$

Descripción: Verifica si la posición p es una entrada inferior del campus c .

VECINOS(**in** $p : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{conj}(pos)$

Pre $\equiv \{\text{posValida}(p, c)\}$

Post $\equiv \{res =_{\text{obs}} \text{vecinos}(p, c)\}$

Descripción: Devuelve un conjunto de las posiciones que rodean a p en el campus c

DISTANCIA(**in** $p_0 : \text{pos}$, **in** $p_1 : \text{pos}$, **in** $c : \text{campus}$) $\rightarrow res : \text{nat}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{distancia}(p_0, p_1, c)\}$

Descripción: Devuelve la distancia, en casilleros, desde la posición p_0 a la posición p_1 .

PROXPOSICION(in p : pos, in d : dir, in c : campus) $\rightarrow res$: pos

Pre $\equiv \{posValida(p, c)\}$

Post $\equiv \{res =_{obs} proxPosicion(p, d, c)\}$

Descripción: Indica la posición que se encuentra al lado de p , en la dirección d .

OBSTACULOS(in c : campus) $\rightarrow res$: conj(pos)

Pre $\equiv \{true\}$

Post $\equiv \{res =_{obs} obstaculos(c)\}$

Descripción: Devuelve un conjunto que contiene todas las posiciones ocupadas por obstáculos en el campus c .
EXTENDER TAD.

Representación

Representación de la lista

campus se representa con **estr**

donde **estr** es `tupla(alto: nat, ancho: nat, obstaculos: matriz(bool))`

donde **pos** es `tupla(fila: nat, columna: nat)`

Invariante de representación en castellano:

1. Para toda p de tipo pos , si p está definida en `obstaculos`, entonces tanto la fila como la columna de p son menores o iguales a `alto` y `ancho` respectivamente.

$Rep : estr \rightarrow bool$

$Rep(e) \equiv true \iff$

$$1. (\forall p : pos) p \in claves(e.obstaculos) \Rightarrow (p.fila \leq e.alto \wedge p.columna \leq e.ancho)$$

$Abs : estr \rightarrow campus$

$\{Rep(e)\}$

$Abs(e) \equiv c : campus /$

$(\forall p : pos) def?(p, e.obstaculos) =_{obs} ocupada?(p, c) \wedge alto(c) =_{obs} e.alto \wedge ancho(c) =_{obs} e.ancho$

Algoritmos

Algoritmos de Campus

Lista de algoritmos

- | | | |
|----|----------------------------|---|
| 1. | NuevoCampus | 2 |
| 2. | AgregarObstaculo | 2 |

$iNuevoCampus(in al : nat, in an : nat) \rightarrow res: estr$

begin

$res.computadoras \leftarrow vacio()$

$//O(1)$

$res.mapa \leftarrow vacia()$

$//O(1)$

$res.indexToString \leftarrow vacia()$

$//O(1)$

end

Complejidad: $O(1)$

Algoritmo 1: NuevoCampus

$iAgregarObstaculo(in p : pos, in/out e : estr)$

begin

$Colocar(p, true, e.obstaculos)$

$//O(1)$

end

Complejidad: $O(1)$

Algoritmo 2: AgregarObstaculo