

1. Módulo Matriz

Interfaz

parámetros formales

género *significado*
función $\text{COPIAR}(\text{in } a : \text{significado}) \rightarrow res : \text{significado}$
Pre $\equiv \{\text{true}\}$
Post $\equiv \{res =_{\text{obs}} a\}$
Complejidad: $\Theta(\text{copy}(a))$
Descripción: función de copia de *significado*

se explica con: DICCIONARIO(POS, SIGNIFICADO)

géneros: matriz

El modulo funciona como un diccionario, pero solo se utiliza con claves del tipo *pos*. Extiende el TAD para contemplar que la creacion de una nueva matriz requiere dos parametros, el *alto* y el *ancho*.

Operaciones básicas de matriz

NUEVAMATRIZ(**in** *al* : nat, **in** *an* : nat) $\rightarrow res : \text{matriz}$
Pre $\equiv \{\text{true}\}$
Post $\equiv \{res =_{\text{obs}} \text{vacío}\}$
Complejidad: $O(n * m)$
Descripción: Crea una nueva matriz de alto *al* x ancho *an*.

COLOCAR(**in** *p* : pos, **in** *s* : significado, **in/out** *m* : matriz)
Pre $\equiv \{m =_{\text{obs}} m_0\}$
Post $\equiv \{m =_{\text{obs}} \text{definir}(p, s, m_0)\}$
Complejidad: $O(1)$
Descripción: Coloca (*define*) el significado *s* en la posicion *p* de la matriz *m*.

OCUPADA?(**in** *p* : pos, **in** *m* : matriz) $\rightarrow res : \text{bool}$
Pre $\equiv \{\text{true}\}$
Post $\equiv \{res =_{\text{obs}} \text{def?}(p, m)\}$
Complejidad: $O(1)$
Descripción: Devuelve *true* si la posicion *p* esta ocupada.

OBTENER(**in** *p* : pos, **in** *m* : matriz) $\rightarrow res : \text{significado}$
Pre $\equiv \{\text{def?}(p, m)\}$
Post $\equiv \{res =_{\text{obs}} \text{obtener}(p, m)\}$
Complejidad: $O(1)$
Descripción: Retorna el *significado* almacenado en la posicion *p*.

ELIMINAR(**in** *p* : pos, **in/out** *m* : matriz)
Pre $\equiv \{\text{def?}(p, m) \wedge m =_{\text{obs}} m_0\}$
Post $\equiv \{m =_{\text{obs}} \text{borrar}(p, m_0)\}$
Complejidad: $O(1)$
Descripción: Elimina el contenido de la posicion *p* de la matriz *m*.

POSICIONESOCUPADAS(**in** *m* : matriz) $\rightarrow res : \text{conj}(\text{pos})$
Pre $\equiv \{\text{true}\}$
Post $\equiv \{res =_{\text{obs}} \text{claves}(m)\}$
Complejidad: $O(1)$
Descripción: Devuelve el conjunto de posiciones ocupadas en la matriz *m*

Representación

Representación de Matriz

matriz se representa con *estr*

donde *estr* es *tupla*(*alto*: nat , *ancho*: nat , *claves*: conjRapido(*pos*), *tablero*: vector(vector(*info*)))
donde *info* es *tupla*(*definido*: bool, *dato*: significado)
donde *pos* es *tupla*(*fila*: nat, *columna*: nat)

Invariante de representacion en castellano:

1. La longitud de tablero es *alto*
2. Para toda posicion de tablero, el vector que contiene posee longitud *ancho*
3. Para toda clave *p* en el rango de la matriz, *p* contenida en *claves* implica que las componentes de *c* (.fila, .columna) en *tablero* dan una tupla *info* donde .definido es *true*.
4. Analogo al anterior, pero para toda *p* que este en el rango de la matriz y no este contenida en *claves*, la tupla *info* posee .definido igual a *false*

Rep : *estr* \longrightarrow bool

Rep(*e*) \equiv true \iff

1. Longitud(tablero) =_{obs} alto \wedge
2. $(\forall i : \text{int}) (i < \text{Longitud}(\text{tablero})) \Rightarrow_{\text{L}} \text{Longitud}(\text{tablero}[i]) =_{\text{obs}} \text{ancho} \wedge_{\text{L}}$
3. $(\forall p : \text{pos}) (p.\text{fila} \leq \text{alto} \wedge p.\text{columna} \leq \text{ancho} \wedge p \in \text{e.claves}) \Rightarrow_{\text{L}}$
(*tablero*[*p.fila*][*p.columna*].definido =_{obs} *true*)
4. $(\forall p : \text{pos}) (p.\text{fila} \leq \text{alto} \wedge p.\text{columna} \leq \text{ancho} \wedge p \notin \text{e.claves}) \Rightarrow_{\text{L}}$
(*tablero*[*p.fila*][*p.columna*].definido =_{obs} *false*)

Abs : *estr e* \longrightarrow *dicc*(*pos*, *significado*)

{Rep(*e*)}

Abs(*e*) \equiv m : *dicc*(*pos*, *significado*)

$(\forall p : \text{pos}) \text{def?}(p, \text{e.claves}) =_{\text{obs}} \text{def?}(p, m) \wedge$

$(\forall p : \text{pos}) \text{def?}(p, \text{e.claves}) \Rightarrow_{\text{L}} \Pi_2(\text{obtener}(p, \text{e})) =_{\text{obs}} \text{obtener}(p, m))$

Las claves definidas y sus significados son iguales

Algoritmos

Lista de algoritmos

1.	NuevaMatriz	3
2.	Colocar	3
3.	Ocupada?	3
4.	Obtener	3
5.	Eliminar	4
6.	PosicionesOcupadas	4

*i*NuevaMatriz(**in** *al*: nat, **in** *an*: nat) → res: estr

```
begin
  res.alto ← al                                // O(1)
  res.ancho ← an                                // O(1)
  res.claves ← Vacio()                          // O(1)
  res.tablero ← CrearArreglo(al)                 // O(al)
  for i ← 0..(al - 1) do                         // O(al)
    | res.tablero[i] ← CrearArreglo(an)         // O(an)
  end
end
Complejidad:  $O(al * an)$ 
```

Algoritmo 1: NuevaMatriz

*i*Colocar(**in** *p*: pos, **in** *s*: significado, **in/out** *e*: estr)

```
begin
  if p.fila > e.alto ∨ p.columna > e.ancho then    // O(1)
    | return e
    | else
      | Agregar(p, e.claves)                       // O(1)
      | e.tablero[p.fila][p.columna] ← <true, s>    // O(1)
    | end
  end
end
Complejidad: O(1)
```

Algoritmo 2: Colocar

*i*Ocupada?(**in** *p*: pos, **in** *e*: estr) → res: bool

```
begin
  if p.fila > e.alto ∨ p.columna > e.ancho then    // O(1)
    | res ← false                                     // O(1)
    | else
      | res ←  $\Pi_1$ (e.tablero[p.fila][p.columna])    // O(1)
    | end
  end
  return res
end
Complejidad: O(1)
```

Algoritmo 3: Ocupada?

*i*Obtener(**in** *p*: pos, **in** *e*: estr) → res: significado

```
begin
  | res ←  $\Pi_2$ (e.tablero[p.fila][p.columna])         // O(1)
  | return res
end
Complejidad: O(1)
```

Algoritmo 4: Obtener

```

iEliminar(in  $p$ : pos, in/out  $e$ : estr)
begin
  | Eliminar( $p$ ,  $e.claves$ ) //O(lo que diga mati)
  |  $\Pi_1(e.tablero[p.alto][p.columna]) \leftarrow false$  //O(1)
end
Complejidad: O(1)

```

Algoritmo 5: Eliminar

```

iPosicionesOcupadas(in  $e$ : estr)  $\rightarrow$  res: conj(pos)
begin
  | res  $\leftarrow e.claves$  //O(1)
  | return  $res$ 
end
Complejidad: O(1)

```

Algoritmo 6: PosicionesOcupadas
