

Universidad de Buenos Aires
Facultad de Ingeniería

75.41 – Algoritmos y Programación II

Cátedra Ing. Patricia Calvo

2º Cuatrimestre 2007

Trabajo Práctico

Hormigas

Índice

<u>Índice.....</u>	<u>3</u>
<u>.....</u>	<u>3</u>
<u>Enunciado.....</u>	<u>4</u>
<u>Objetivo.....</u>	<u>4</u>
<u>Contexto.....</u>	<u>4</u>
<u>Recolección de alimento.....</u>	<u>4</u>
<u>Reglas.....</u>	<u>6</u>
<u>Simulación.....</u>	<u>6</u>
<u>Terreno.....</u>	<u>6</u>
<u>Hormiguero.....</u>	<u>7</u>
<u>Hormiga.....</u>	<u>7</u>
<u>Semilla.....</u>	<u>8</u>
<u>Planta.....</u>	<u>8</u>
<u>Piedra.....</u>	<u>9</u>
<u>Mecánica.....</u>	<u>9</u>
<u>Diseño.....</u>	<u>10</u>
<u>TDAs.....</u>	<u>10</u>
<u>Herencia.....</u>	<u>15</u>
<u>.....</u>	<u>15</u>
<u>Descripciones.....</u>	<u>15</u>
<u>.....</u>	<u>18</u>
<u>Código Fuente.....</u>	<u>19</u>
<u>Entorno de desarrollo.....</u>	<u>19</u>
<u>Índice de archivos.....</u>	<u>19</u>
<u>Fuentes.....</u>	<u>20</u>

Enunciado

Objetivo

Construir una aplicación que simule el comportamiento de una colonia de hormigas durante la recolección de alimento.

Contexto

Muchas especies de hormigas utilizan sustancias químicas que son capaces de segregar para establecer la comunicación entre los miembros de la colonia. Estas sustancias químicas reciben el nombre de *Feromonas*.

Las hormigas aplican este mecanismo de comunicación para conducirse coordinadamente a través de su medio ambiente.

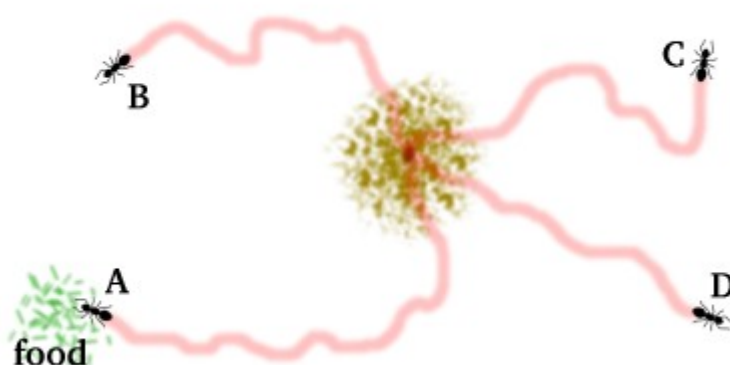
Recolección de alimento

Consideremos el caso de una colonia de hormigas buscando comida. Asumamos que inicialmente ningún miembro de la colonia tiene información sobre la localización del alimento y se encuentran las proximidades del hormiguero.

Las hormigas realizan la búsqueda de comida caminando de manera aleatoria por todo el terreno.

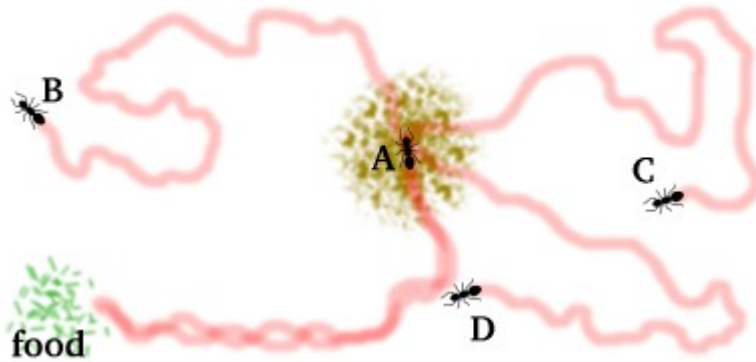
Existen dos problemas que deberán superar para cumplir con su cometido. En primer lugar cómo regresar al hormiguero una vez que encuentren el alimento, llevando consigo una parte del mismo. Luego, cómo pueden informales a otras hormigas sobre la localización del alimento, para que toda la colonia pueda participar del proceso de recolección.

Para resolver el problema de encontrar el camino de vuelta, cada hormiga deja un rastro de feromona cuando está buscando comida. Como se muestra en el siguiente diagrama cada hormiga que participa de la recolección sale del hormiguero caminando de manera aleatoria dejando un rastro de feromona.

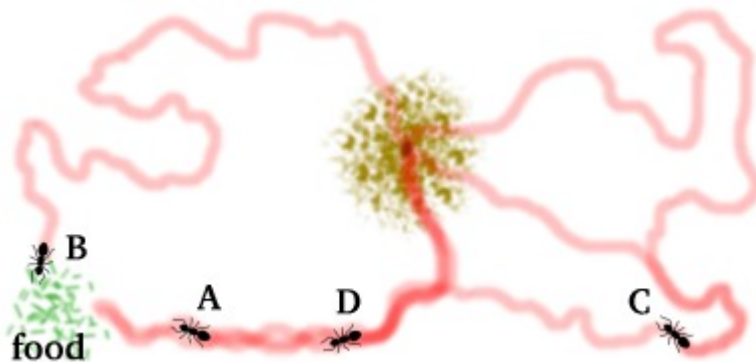


Cuando una hormiga encuentra comida puede seguir su propio rastro hasta el hormiguero.

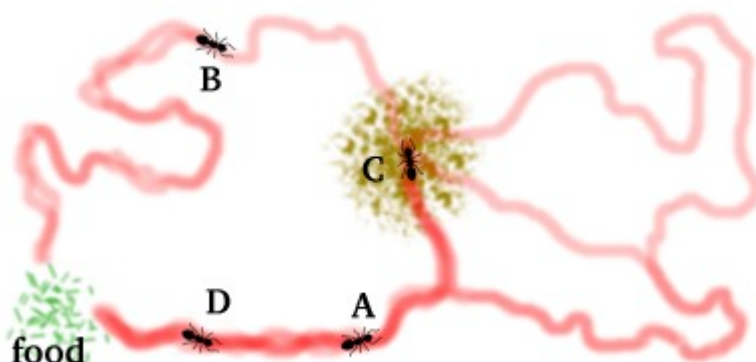
En el camino de regreso hasta el hormiguero se ocupa de resolver el problema de indicarle al resto de la colonia sobre su hallazgo intensificando su propio rastro, creando una ruta mejor delimitada que conducirá desde el hormiguero hasta el alimento.



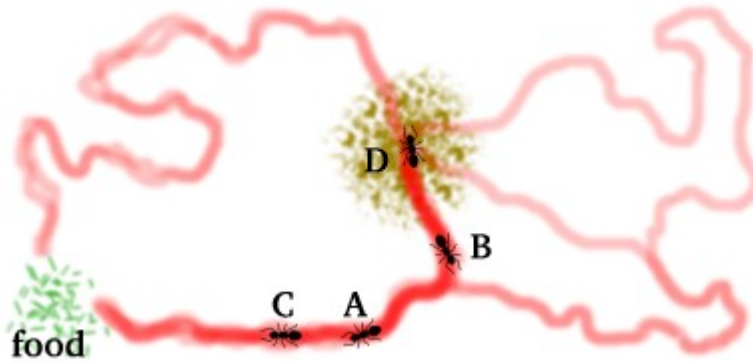
En el dibujo anterior la hormiga A encontró el alimento y luego regresó al hormiguero dejando un rastro más intenso de feromonas, mientras que el resto (B, C y D) continúan la búsqueda, sin éxito hasta el momento. Cuando otra hormiga se encuentra con un camino de feromonas abandona su propio rastro y comienza a seguirlo, intensificándolo aún más.



Como se puede ver en el esquema anterior las hormigas A, D y C se conducirán desde el alimento hasta el hormiguero siguiendo el camino originalmente descubierto por A. También se observa que eventualmente la hormiga B estableció su propio recorrido.



Luego de regresar al hormiguero y antes de emprender un nuevo recorrido cada hormiga selecciona el camino a seguir comparando la intensidad del rastro de todos los caminos salientes del mismo. En el esquema, cuando la hormiga B regresa al hormiguero se encuentra con un camino mejor delimitado (el que definió originalmente A) y por lo tanto sigue ese camino.



Reglas

Todo el proceso está gobernado por un pequeño conjunto de reglas simples que sigue cada uno de los miembros de la colonia

Condición	Acción
No está acarreando comida. No está siguiendo un rastro de feromonas.	Camina en dirección aleatoria. Deja a su paso un rastro de feromonas.
No está acarreando comida. Está siguiendo un rastro de feromonas.	Sigue el rastro de feromonas. Deja a su paso feromonas, intensificando el rastro.
Está siguiendo un rastro de feromonas. Llegó al hormiguero sin comida.	Da la vuelta. Sigue el rastro en sentido contrario.
Encontró comida.	Toma la comida. Da la vuelta. Sigue el rastro en sentido contrario.
Está acarreando comida.	Sigue el rastro. Deja a su paso feromonas, intensificando el rastro.
Llegó al hormiguero con comida.	Deposita la comida. Da la vuelta. Sigue el rastro en sentido contrario.

En todos los casos en que debe seguir un rastro, ante la necesidad de decidir entre múltiples rastros elige aquel cuyo nivel de feromonas sea más intenso.

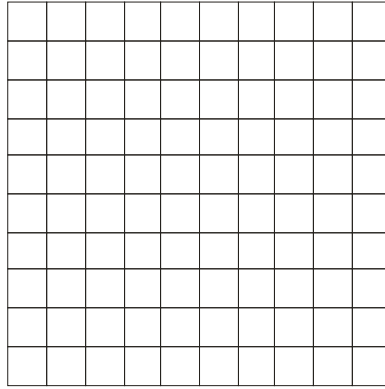
Aclaración: Los diagramas y dibujos que se muestran son descriptivos, tienden a mostrar la mecánica del juego de manera clara y concreta, pero de ninguna manera establecen algún tipo de requerimiento respecto a una interfaz gráfica del programa a desarrollar. Toda la comunicación del programa desde y hacia el usuario se realizará a través de archivos.

Simulación

La simulación deberá sustentarse en las definiciones realizadas en la presente sección.

Terreno

El terreno constituye el medio en el que toda la simulación será realizada. Es una grilla rectangular que define las posiciones que pueden ocupar el resto de los elementos de la simulación.



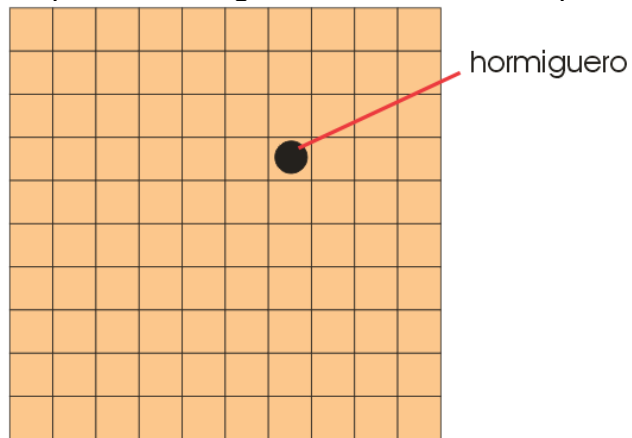
Hormiguero

Lugar al que las hormigas deben llevar el alimento recolectado.

Ocupa una posición en el terreno.

No puede moverse.

Almacena los alimentos que las hormigas acarrearán hasta su posición.

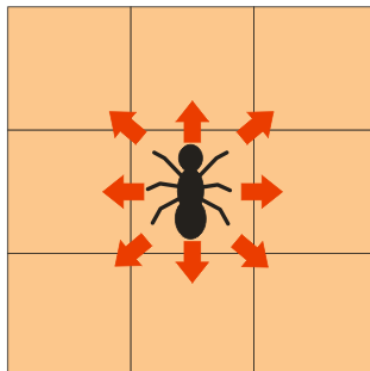


Hormiga

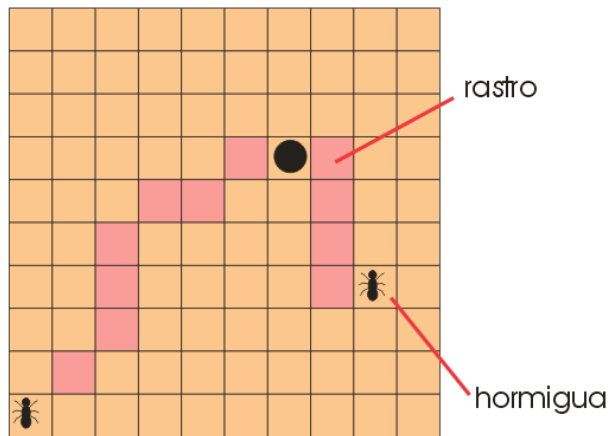
Miembro de la colonia que buscan alimentos y los acarrearán hasta el hormiguero.

Ocupa una posición en el terreno.

Se desplazan a una de las ocho (8) posiciones contiguas cada vez que caminan.



Pueden impregnar con feromona la posición del terreno en la que se encuentran.

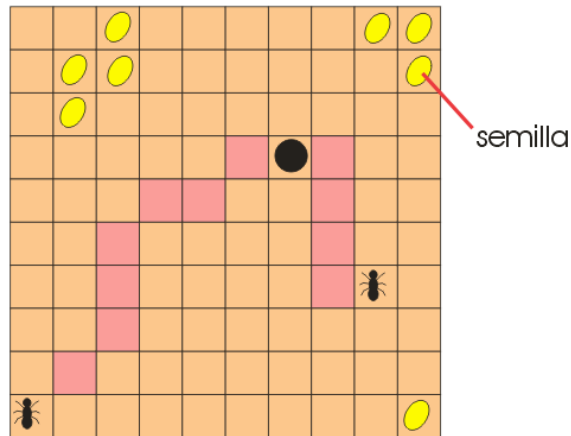


El nivel de feromona en una posición del terreno es un valor entre 0 y 999.

Semilla

Alimento que puede ser acarreado por una hormiga.

Ocupa una posición en el terreno.



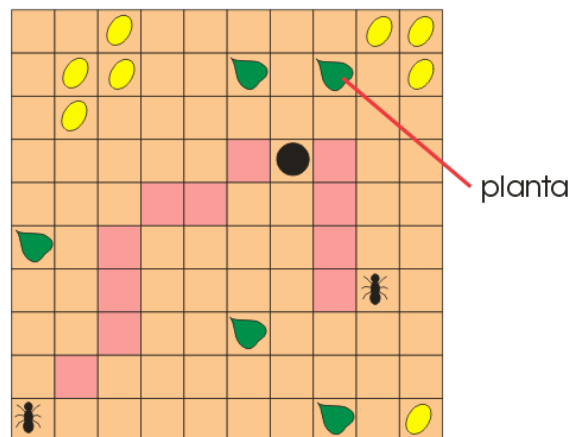
Planta

Alimento que las hormigas pueden cortar segmentos y acarrear.

No puede moverse.

Ocupa una posición en el terreno.

El tamaño de la planta determina la cantidad de segmentos que pueden ser cortados de la misma.

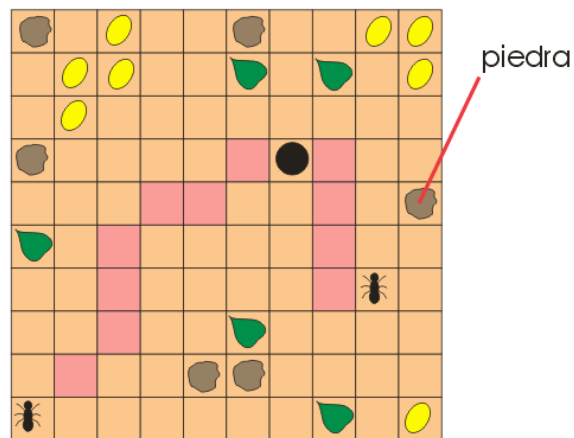


Piedra

Obstáculo inamovible presente en el terreno.

Ocupa una posición en el terreno.

No puede moverse.



Mecánica

La simulación comienza a partir de la definición del terreno y de todos los elementos presentes en el mismo. Esta información estará disponible en un archivo con la estructura que se describe en la sección: Archivos de Entrada, Terreno.

La simulación está constituida por una secuencia de rondas. En cada una de las rondas cada miembro de la colonia tendrá un turno para ejecutar una única acción de acuerdo a la condición en que se encuentre.

Dentro de cada ronda, el orden de los turnos para las hormigas es aleatorio y diferente ronda tras ronda. Por ejemplo si tenemos las hormigas A, B, C y D, el orden de los turnos por ronda podría ser:

Ronda 1 - B C A D

Ronda 2 - D A B C

Ronda 3 - A B C D

Ronda 4 - B A C D

Las acciones que una hormiga puede realizar en un turno

Acción	Descripción
Moverse	Moverse a uno de los 8 (ocho) casillero contiguos que se encuentre desocupado, dejando un rastro nuevo de feromona o intensificando un rastro ya existente.
Tomar semilla	Tomar una semilla que se encuentre en uno de los casilleros contiguos.
Cortar planta	Cortar y tomar un segmento de una planta que se encuentre en uno de los casilleros contiguos.
Dejar carga	Dejar la semilla o segmento de planta que tenga cargado en uno de los casilleros contiguos.
Ninguna	No hacer nada.

Diseño

TDAs

Definición de todos los TDA's desarrollados por el grupo.

TDA HORMIGUERO

Descripción: Lugar de "almacenamiento" de alimento, se lleva la cuenta de la cantidad almacenada. puede diferenciar el tipo de alimento guardado. ocupa un lugar "físico" en el terreno.

Primitivas:

CREAR

pre: *que exista el terreno*

post: *se crea una instancia hormiguero vacío*

CANTIDAD de SEMILLAS

Devuelve la cantidad de Semillas que tiene.

pre: -

post: -

CANTIDAD de PLANTAS

Devuelve la cantidad de porciones de plantas recolectadas.

pre: -

post: -

CARGAR COMIDA

Permite insertar comida en el hormiguero. Se identifica si es una semilla o parte de una planta.

pre: -

post: *se incrementa la cantidad de semillas o plantas en una unidad*

DESTRUIR

Elimina la instancia hormiguero.

pre: -

post: *se elimina el hormiguero y sus datos asociados*

TDA HORMIGA

Descripción: Transporta alimentos de a una unidad por vez. Ocupa un solo casillero en el tablero. Puede moverse. Deja rastros de feromonas.

Primitivas:

CREAR

pre: -

post: *se crea una instancia de hormiga*

RECOGER

pre: -

post: *se modifica el estado, se asigna una carga*

DEPOSITAR

pre: *debe haber una carga*
post: *se vacia la carga de hormiga*

MOVERSE

pre: -
post: *se le asigna una nueva posición*

DESTRUIR

pre: -
post: *se elimina la instancia*

TDA SEMILLA

Descripción: Ocupa una posición en el terreno, es transportable hasta el hormiguero. No tiene movimiento propio.

Primitivas:

CREAR

pre: *que exista el terreno*
post: *se crea una instancia semilla*

DESTRUIR

pre: -
post: *se elimina la instancia semilla*

TDA PLANTA

Descripción: Estructura de HOJAS transportables.
La planta ocupa una sola posición en el tablero y no es transportable.
debe tener al menos 1 (una) HOJA.

Primitivas:

CREAR

pre: *que exista el terreno, recibir la cantidad de hojas inicial*
post: *se crea una instancia planta con una cantidad de hojas*

CANTIDAD

Devuelve la cantidad de HOJAS disponibles.

pre: -
post: -

DAME HOJA

pre: *debe tener al menos una hoja*
post: *la planta tiene una hoja menos*

ESTA VACIA

Devuelve un booleano (true= vacia, false= tiene 1 o mas hojas)

pre: -
post: -

DESTRUIR

pre: -

post: *se elimina la instancia planta y hojas*

TDA TERRENO

Descripción: Lugar donde están las hormigas, y todos los elementos de la simulación.

Primitivas:

CREAR

pre: -

post: *se crea la instancia terreno*

ADMINISTRAR FEROMONAS

Aumenta la intensidad de feromona en una posición dada.

pre: -

post: -

ELIMINAR PLANTA

Elimina la instancia de planta en una posición dada.

pre: -

post: -

DESTRUIR

pre: -

post: *se elimina la instancia terreno*

TDA PIEDRA

Descripción: Objeto inamovible en el terreno.
Ocupa una posición en el terreno.

Primitivas:

CREAR

pre: *que exista el terreno*

post: *se crea una instancia piedra*

DESTRUIR

pre: -

post: *se elimina la instancia piedra*

TDA CELDA

Descripción: Unidad básica del Terreno, solo puede contener un elemento por vez.

Axioma: La Posición de una celda no puede ser modificada

Primitivas:

CREAR

pre: -

post: *instancia del objeto celda creada*

INCREMENTA FEROMONA

pre: -

post: *se incrementa el valor de feromona en una unidad*

BORRAR CONTENIDO

pre: -

post: *se elimina el contenido*

ES HORMIGUERO

Responde si es una instancia de hormiguero.

pre: -

post: -

ES COMIDA

pre: -

post: -

POSICION

Devuelve una posicion.

pre: -

post: -

FEROMONA

Nos dice la cantidad de feromonas.

pre: -

post: -

DESTRUIR

pre: -

post: *la instancia del objeto es destruida*

TDA SIMULADOR

Descripción: Controla la temática de la simulación. Vincula los elementos con sus interacciones. Pasa turnos aleatorios para la cantidad hormigas disponibles.

Primitivas:

CREAR

pre: -

post: *se crea la instancia del objeto*

SIMULAR

Inicia el proceso de turnos y rondas

pre: -

post: -

DESTRUIR

pre: -

post: *se elimina la instancia simulador*

TDA ELEMENTO

Descripción: Es la unidad elemental de todos los objetos. (es algo así como un átomo, todo se deriva de este)

Primitivas:

CREAR

pre: -

post: *se crea una instancia de elemento*

NOMBRE

Nos dice el nombre del elemento.

pre: -

post: -

POSICION

Nos dice en que posición se encuentra.

pre: -

post: -

ASIGNAR POSICION

pre: -

post: -

DESTRUIR

pre: -

post: *se elimina la instancia elemento*

TDA POSICION

Descripción: Ubicación dentro del terreno de simulación.

Primitivas:

CREAR

pre: -

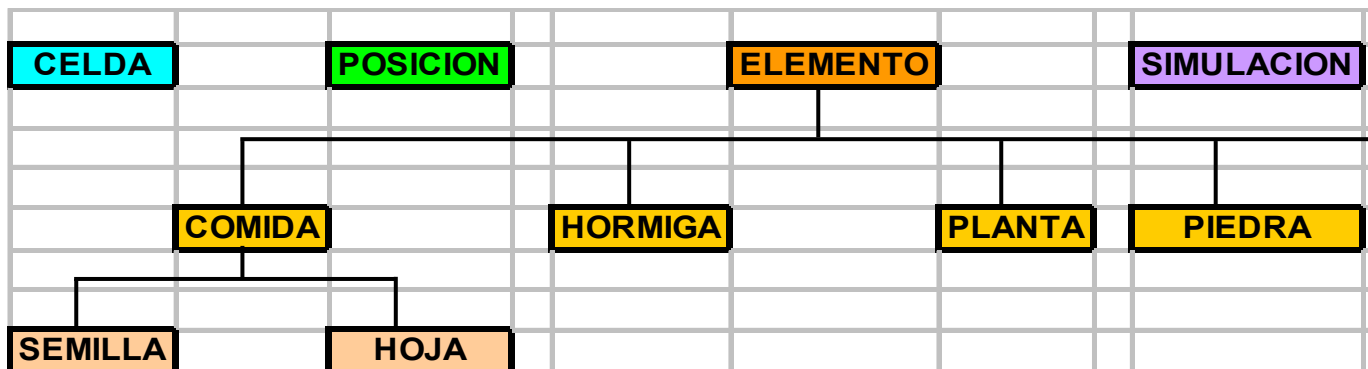
post: *se crea una instancia de posición*

DESTRUIR

pre: -

post: *se elimina la instancia de posición*

Herencia



Descripciones

TDA CELDA

Esta clase se encarga de administrar cada celda del campo. Es la encargada de crearlas, destruirlas y maneja sus contenidos, a saber su creación, destrucción y la obtención de datos acerca de ellos. Además es el encargado de administrar el nivel de feromona en cada celda del terreno. Tiene un objeto del tipo POSICION en su interior y un puntero a un ELEMENTO.

TDA COMIDA

Esta clase es la que administra los objetos transportables por los TDA HORMIGAS. Se encarga de crearlas de dos maneras distintas (con o sin argumentos). De ella se obtienen 2 hijas, TDA SEMILLA Y TDA HOJA, que heredan sus métodos.

TDA SEMILLA/ HOJA

Estas clases de diseño similar, derivan de su padre TDA COMIDA, y poseen sus propios métodos de creación y destrucción. Cada uno especificado para cada clase.

TDA ELEMENTO

Es una de las mas importantes, ya que varios TDA de este programa son hijas de esta clase, si bien sus métodos son bastante simples. Creación, destrucción y la obtención de datos acerca del objeto (su nombre) son sus métodos particulares.

TDA HORMIGA

Esta clase define varios de los métodos que la hormiga utilizara durante el desarrollo del programa. Se encarga de crearlas y destruirlas. Además cada hormiga puede dar información sobre su carga (lo que este llevando en ese momento), manejar la posición en la que se encontraba un turno antes, e

interactuar con el medio. Las acciones que utiliza para esto ultimo son recoger y depositar.

TDA HORMIGUERO

Es el encargado de administrar la comida que las hormigas ingresan en el hormiguero, devuelve información sobre sus cantidades y se encarga de incrementarlas en el momento de que alguna hormiga deposite comida. Contiene dos listas de punteros, cada una sobre un tipo de comida distinto.

TDA PIEDRA

Clase que define este objeto, sus métodos son creación y destrucción ya que no posee ninguna utilidad para las hormigas.

TDA PLANTA

Esta clase posee una lista de punteros a HOJAS. Se encarga de dar información acerca de la cantidad de hojas que posee, y también interactúa con la hormiga al “darle” una hoja cuando esta la recoge. Posee su constructor y destructor.

TDA POSICION

Esta clase administra el manejo de los objetos POSICION. Permite crearlos de dos maneras distintas (con o sin argumentos), modificar los datos que contienen y dar información acerca de ellos. Es la base para poder moverse por el terreno de simulación.

TDA SIMULACION

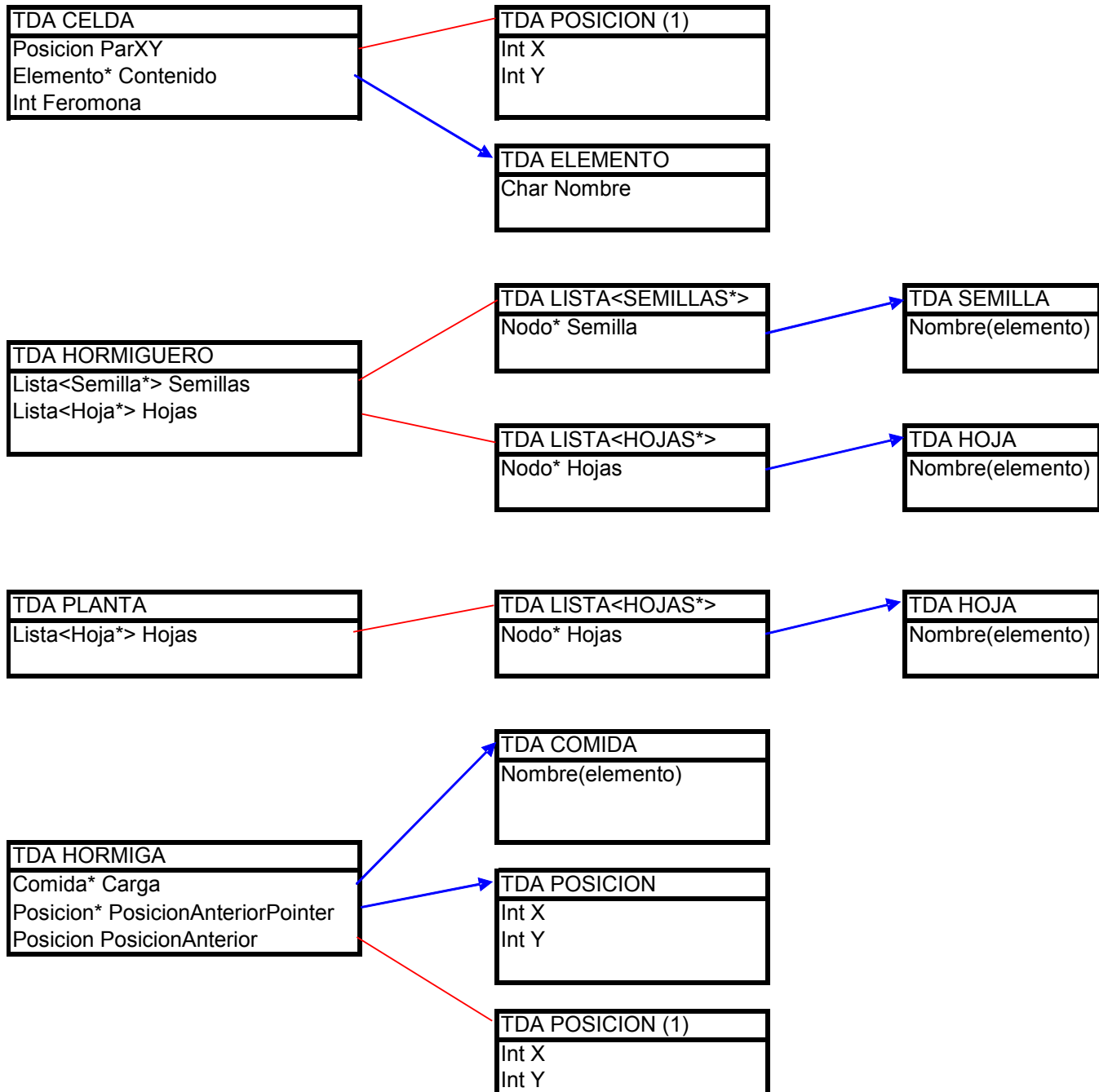
El mas importante de todos, es el encargado de administrar todo el proceso. Posee un puntero a TERRENO y un valor denominado Rondas (que serán la cantidad de veces que las hormigas realizaran sus turnos). Se encarga de ejecutar la construcción del terreno, ejecutar las rondas y en cada una de ellas administra las acciones de cada hormiga durante su turno. Al final de cada ronda, muestra los resultados de esa ronda.

TDA TERRENO

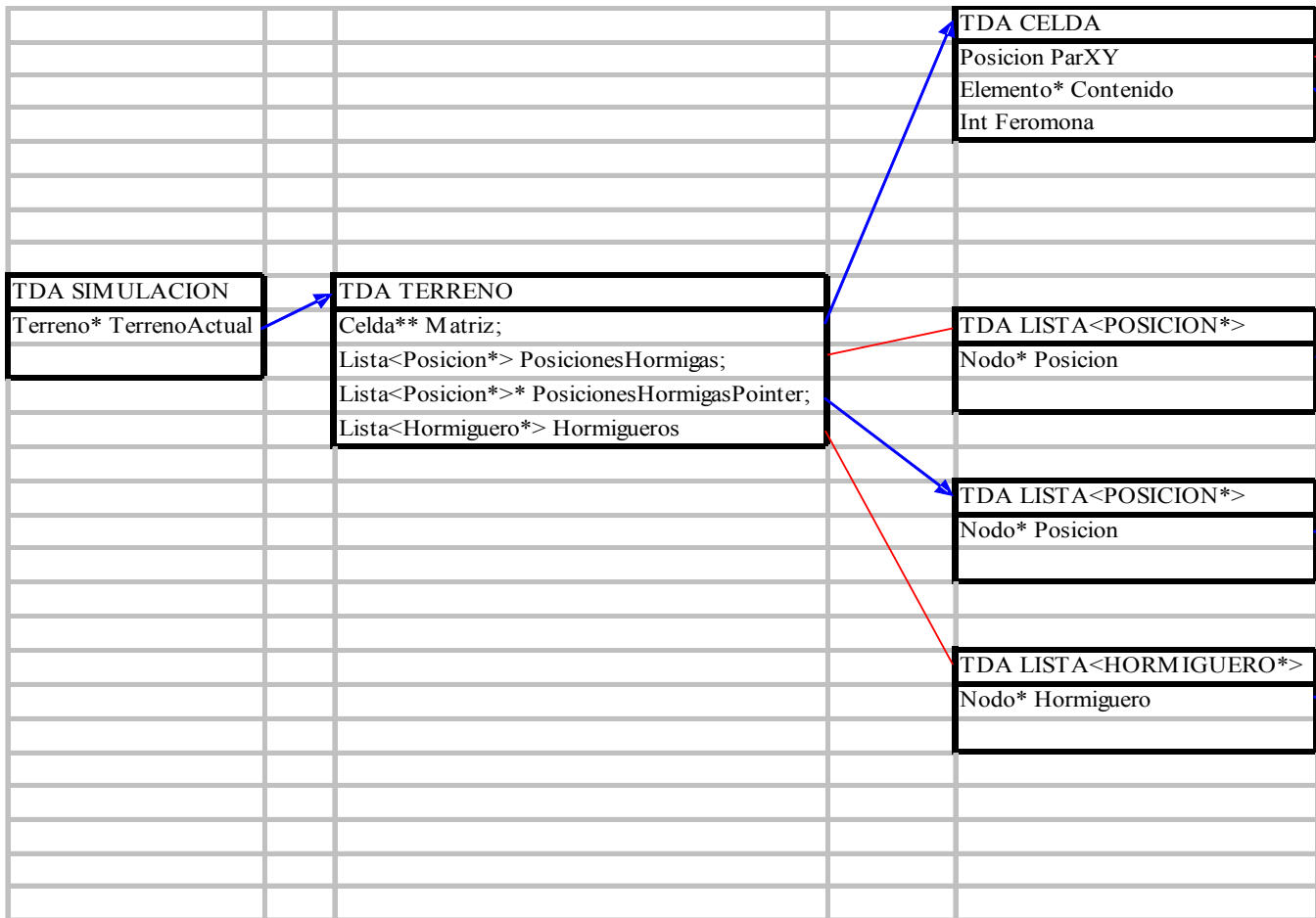
Posee un puntero a un puntero a CELDA y 2 listas: una de punteros a POSICIONES, un puntero a esa lista y una lista de punteros a HORMIGUEROS. Se encarga de la creación del terreno y la ubicación de los objetos dentro de el a medida que se lee el archivo de texto, modificando las dimensiones del terreno si es necesario. Además posee información sobre las posiciones de cada hormiga para su posterior utilización.

TDA VACIO

Este TDA simplemente sirve para colocar en los lugares del terreno en donde no se encuentra ningún objeto, para ser mas simple su identificación durante el transcurso de la aplicación.



Interacciones



Código Fuente

Entorno de desarrollo

Utilizamos el compilador MinGW y Eclipse Platform como IDE

Índice de archivos

#ifndef TDAHORMIGUERO_H_.....	20
#ifndef TDAHORMIGA_H_.....	20
#ifndef TDASEMILLA_H_.....	21
#ifndef TDAPLANTA_H_.....	21
#ifndef TDATERRENO_H_.....	22
#ifndef TDAPIEDRA_H_.....	22
#ifndef TDACELDA_H_.....	22
#ifndef TDAELEMENTO_H_.....	24
#ifndef TDAPOSICION_H_.....	24
#include "TdaElemento.h".....	24
#include "TdaHormiga.h".....	25
#include "TdaSemilla.h".....	25
#include "TdaPlanta.h".....	25
#include "TdaTerreno.h".....	26
#include "TdaPiedra.h".....	27
#include "TdaCelda.h".....	27
#include "TdaElemento.h".....	28
#include "TdaPosicion.h".....	28

Fuentes

```
#ifndef TDAHORMIGUERO_H_
#define TDAHORMIGUERO_H_

class Hormiguero:public Elemento {

private:
    int semillas;
    int plantas;
public:
    /*pre:-
    post:instancia del objeto Hormiguero creada*/
        Hormiguero(Posicion, char);

    /*pre:-
    post:*/
        int cantSemillas();

    /*pre:-
    post:*/
        int cantPlantas();

    /*pre:-
    post: se modifica el estado de la instancia (se aumenta la cantidad de carga)*/
        void cargarComida(char);

    /*pre:-
    post:la instancia del objeto es destruida */
        ~Hormiguero();
};

#endif /*TDAHORMIGUERO_H_*/

#ifndef TDAHORMIGA_H_
#define TDAHORMIGA_H_

class Hormiga:public Elemento
{
private:
    char Carga;
public:
    /*pre:-
    post:instancia del objeto Hormiga creada*/
        Hormiga(Posicion, char);

    /*pre:-
    post: se modifica el estado de la instancia (se asigna una carga)*/
        void Recoger(char);

    /*pre: debe tener una carga anteriormente asignada
    post: se modifica el estado de la instancia (se vacia la carga)*/
        void Depositar();
};
```

```

        /*pre:-
        post: se modifica el estado de la instancia (se asigna la nueva posicion)*/
        void Moverse(Posicion);

        /*pre:-
        post:la instancia del objeto es destruida */
        ~Hormiga();
};
#endif /*TDAHORMIGA_H_*/

#ifndef TDASEMILLA_H_
#define TDASEMILLA_H_

class Semilla:public Elemento
{
public:
    /*pre:-
    post:instancia del objeto Piedra creada*/
    Semilla(Posicion, char);

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Semilla();
};

#endif /*TDASEMILLA_H_*/

#ifndef TDAPLANTA_H_
#define TDAPLANTA_H_

class Planta:public Elemento
{
private:
    int hojas;
public:
    /*pre:-
    post:instancia del objeto Plamta creada*/
    Planta(int,Posicion,char);

    /*pre:-
    post:-*/
    bool estaVacia();

    /*pre:-
    post:-*/
    int getCantidad();

    /*pre:-
    post:se modifica el estado de la instancia(cantidad de hojas disminuida)*/
    void darHoja();

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Planta();
};

```

```

#endif /*TDAPLANTA_H_*/

#ifndef TDATERRENO_H_
#define TDATERRENO_H_

class Terreno{
/*Axioma:El Terreno tiene un tamaño variable hasta la finalizacion del proceso de
construccion;
luego tiene un tamaño fijo hasta su destruccion*/

private:
    Celda* Matriz;

public:
    /*pre:-
    post:instancia del objeto Celda creada*/
    Terreno(int filas, int columnas);

    /*pre:llamada de un objeto Hormiga
    post:se envia un mensaje a un objeto Celda;el terreno sera modificado*/
    void AdministraFeromona(Posicion par);

    /*pre:estaVacía
    post:se elimina un objeto planta;el terreno sera modificado*/
    void eliminaPlanta(Posicion par);

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Terreno();
};

#endif /*TDATERRENO_H_*/

#ifndef TDAPIEDRA_H_
#define TDAPIEDRA_H_

class Piedra:public Elemento
{
private:
public:
    /*pre:-
    post:instancia del objeto Piedra creada*/
    Piedra(Posicion, char);

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Piedra();
};

#endif /*TDAPIEDRA_H_*/

#ifndef TDACELDA_H_
#define TDACELDA_H_

```

```

#include "TdaPosicion.h"
#include "TdaElemento.h"

class Celda{
//Axioma:La Posicion de una Celda no puede ser modificada;

private:
    Posicion ParXY;

    Elemento* Contenido;

    int Feromona;

public:
    /*pre:-
    post:instancia del objeto Celda creada*/
    Celda(Posicion par,Elemento* c);

    /*pre:-
    post:-*/
    bool EsHormiguero();

    /*pre:-
    post:-*/
    bool EsComida();

    /*pre:-
    post:estado de instancia modificado(se aumenta el valor de Feromona en
una unidad)*/
    void IncrementaFeromona();

    /*pre:-
    post:estado de instancia modificado(Contenido apunta a NULL)*/
    void BajaContenido(void* c);

    /*pre:-
    post:-*/
    Posicion getPosicion();

    /*pre:-
    post:-*/
    int getFeromona();

    /*pre:-
    post:se realiza una asignacion sobre la instancia del objeto*/
//    Celda operator=(Celda &c);

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Celda();

};

#endif /*TDACELDA_H_*/

```

```

#ifndef TDAELEMENTO_H_
#define TDAELEMENTO_H_
#include "TdaPosicion.h"

class Elemento
{
protected:
    char Nombre;
    Posicion _Posicion;
public:
    /*pre:-
    post:instancia del objeto Elemento creada*/
    Elemento(Posicion, char);

    /*pre:-
    post:-*/
    char getNombre();

    /*pre:-
    post:-*/
    Posicion getPosicion();

    /*pre:-
    post:la instancia del objeto es destruida */
    ~Elemento();
};

#endif /*TDAELEMENTO_H_*/

#ifndef TDAPOSICION_H_
#define TDAPOSICION_H_

class Posicion{

    private:
        int X;
        int Y;
    public:
        /*pre:-
        post:instancia del objeto Posicion creada*/
        Posicion(int x,int y);
};

#endif /*TDAPOSICION_H_*/

#include "TdaElemento.h"
#include "TdaHormiguero.h"

Hormiguero::Hormiguero(Posicion par, char o='O'): Elemento(par, o), plantas(0),
semillas(0){}

int Hormiguero::cantSemillas()
{
    return semillas;
}

```

```

}

int Hormiguero::cantPlantas()
{
    return plantas;
}

void Hormiguero::cargarComida(char c)
{
    if (c=='S') semillas++;
    else {
        plantas++;
    }
}

Hormiguero::~Hormiguero(){}

#include "TdaElemento.h"
#include "TdaHormiga.h"

Hormiga::Hormiga(Posicion par, char h='H'):Elemento (par, h), Carga('-'){}

void Hormiga::Recoger(char comida)
{
    Carga=comida;
}

void Hormiga::Depositar()
{
    Carga='-';
}

void Hormiga::Moverse(Posicion par)
{
    _Posicion= par;
}

#include "TdaElemento.h"
#include "TdaSemilla.h"

Semilla::Semilla(Posicion par, char s='S'): Elemento(par,s){}

Semilla::~Semilla(){}

#include "TdaElemento.h"
#include "TdaPlanta.h"

Planta::Planta(int h,Posicion par, char p='P'):Elemento(par, p){

    if(h>9)hojas=9;
    hojas=h;
}

```

```
bool Planta::estaVacia() {
    return(getCantidad()==0);
}
```

```
int Planta::getCantidad()
{
    return hojas;
}
```

```
void Planta::darHoja()
{
    hojas--;
}
```

```
Planta::~~Planta() {}
```

```
#include "TdaCelda.h"
/*#include "TdaHormiga.h"
#include "TdaPiedra.h"
#include "TdaPlanta.h"
#include "TdaPosicion.h"
#include "TdaSemilla.h"*/
#include "TdaElemento.h"
#include "TdaTerreno.h"
```

```
Terreno::Terreno(int filas, int columnas){
```

```
    /* creo un vector de punteros a punteros a Celda */
    // Celda*** Matriz = new Celda**[filas];
```

```
    /* para cada una de las posiciones creo un vector de punteros a Celda */
    /* for (int indice = 0; indice < filas; indice++) {
        Matriz[indice] = new Celda*[columnas];
    } */
```

```
    /* creo las Celdas */
    /* for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            Matriz[i][j] = new Celda();
        }
    } */
```

```
}
```

```
void Terreno::AdministraFeromona(Posicion par){//IncrementarFeromona();
```

```
    /* int i,j;
    i=par.X;
    j=par.Y;
    Matriz[i,j].IncrementaFeromona(); */
```

```
}
```

```
void Terreno::eliminaPlanta(Posicion par){//ESTE SE ENCARGA DE LA EUTANASIA
XD!
```

```
/* int i,j;
i=par.X;
j=par.Y;
Matriz[i,j]->~Planta(); */
```

```
}
```

```
Terreno::~Terreno(){
```

```
    /* libero la memoria */
/*    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            delete Matriz[i][j];
        }
    }

    for (int indice = 0; indice < 20; indice++)
        delete [] Matriz[indice];

    delete [] Matriz; */
}
```

```
#include "TdaElemento.h"
```

```
#include "TdaPiedra.h"
```

```
Piedra::Piedra(Posicion par, char p='P'): Elemento (par,p){}
```

```
Piedra::~Piedra() {}
```

```
#include <string.h>
```

```
#include "TdaElemento.h"
```

```
#include "TdaCelda.h"
```

```
Celda::Celda(Posicion par,Elemento* c):ParXY(par),Contenido(c), Feromona(0)
{}//Constructor
```

```
bool Celda::EsHormiguero(){
    if ((*Contenido).getNombre()=='O')
        return true;
    else return false;
}
```

```
bool Celda::EsComida(){
    if (((*Contenido).getNombre()=='S')||
    ((*Contenido).getNombre()=='P'))/"S":semilla;"P":planta;
    return true;
    else return false;
}
```

```
void Celda::IncrementaFeromona(){
    Feromona++;
}
```

```
void Celda::BajaContenido(void* c){
    c=NULL;
}
```

```
Posicion Celda::getPosicion(){
```

```

        return ParXY;
    }
int Celda::getFeromona(){
    return Feromona;
}

/*Celda Celda::operator=(Celda &c){
    ParXY=c.ParXY;//para hacer las asignaciones durante la construccion del
terreno
    Feromona=c.Feromona;
    if (c.Contenido!= NULL) Contenido=c.Contenido;
    else Contenido=NULL;
    return this;
}*/

Celda::~Celda(){
    delete Contenido;}//Destructor

#include "TdaElemento.h"
#include "TdaPosicion.h"

Elemento::Elemento(Posicion par, char p):Nombre(p), _Posicion(par){}

char Elemento::getNombre()
{
    return Nombre;
}

Posicion Elemento::getPosicion()
{
    return _Posicion;
}

Elemento::~Elemento(){}

#include "TdaPosicion.h"

Posicion::Posicion(int x,int y):X(x),Y(y){}

```