Escuela Superior de Ingeniería y Tecnología
Universidad de La Laguna

Grado en Ingeniería Informática
Algoritmos y Estructuras de Datos Avanzadas

Curso 2025-2026

Práctica 2: Hormiga de Langton Generalizada

---

1. Objetivo

El objetivo de esta práctica es comprender el funcionamiento de los mecanismos de herencia y polimorfismo dinámico en programación orientada a objetos, y utilizarlos adecuadamente en un programa en lenguaje C++.

2. Entrega

La práctica consta de una sesión de entrega del 23 al 26 de febrero.

3. Enunciado

Esta práctica es una continuación del desarrollo realizado en la práctica 1, la Hormiga de Langton [1]. Para trabajar los mecanismos de herencia y polimorfismo en el lenguaje C++ se generaliza el concepto de hormiga, que pasa a ser un concepto abstracto, a partir del cual se generan distintos tipos de hormigas cada uno de los cuales tiene un comportamiento particular. Se pide implementar una aplicación en la que coexistan sobre una misma cinta varias hormigas de distintos tipos.

Para definir los distintos comportamientos vamos a utilizar la propuesta de Gale et al. [2], donde se generalizan las reglas clásicas de la Hormiga de Langton para permitir que las celdas de la cinta tomen más de dos colores. A cada color se le asocia a un sentido de giro (90º a la derecha 'D' o 90º a la izquierda 'I'). De esta forma, para una cinta con celdas de n colores, `0 <= c < n`, se pueden definir un tipo de hormiga que se identifica mediante una cadena de D's e I's. La letra en la i-ésima posición de la cadena indica el sentido de giro que debe realizar la hormiga cuando se encuentra sobre una celda del i-ésimo color.

Así, por ejemplo, el tipo de hormiga nombrado como "DDII" se mueve sobre una cinta bidimensional cuyas celdas se alternan entre 4 colores. Con los dos primeros colores la hormiga gira a la derecha; y en los dos últimos colores gira a la izquierda. Siguiendo esta nomenclatura el tipo de hormiga de la versión clásica con dos colores se identifica por "DI".

En esta nomenclatura se asume que los cambios de color que provoca una hormiga en la celda por la que pasa siguen en comportamiento ciclico. Esto es, incrementa el número del color y cuando llega al color `n` vuelve al color `0`. Aunque la implementación debe permitir que cada tipo de hormiga defina su regla para cambiar el color.

4. Notas de implementación

La implementación de esta práctica requiere realizar algunos cambios en el código desarrollado para la práctica 1:

- En la clase `Tape` se actualiza el concepto de celda para permitir que su color cambie en un rango de `n` colores. Se sugiere utilizar un tipo enumerado con ámbito [4] para definir los posibles colores.
- Clase `Ant` se convierte en la clase base abstracta que define un método nulo para realizar su movimiento. Cada tipo de hormiga derivada, definida según la nomenclatura vista en este enunciado, implementa sus propias reglas de movimiento.

- Cada clase `Ant_X`, que deriva de la clase base `Ant`, implementa el tipo particular de hormiga `X` la cadena que, siguiendo la nomenclatura propuesta en [2], indica el sentido de giro para cada color de celda. Utilizando las sentencias de escape ANSI [3], a cada tipo hormiga se le asigna un color para su visualización.

- El objeto `Simulator` sigue siendo el responsable de inicializar y evolucionar la simulación. En el fichero con los datos requeridos para la inicialización es necesario realizar los siguientes cambios:

  Línea 1: Tamaño de la cinta y número de colores de las celdas
  Línea 2: Tipo, posición inicial y orientación de cada hormiga (separadas por `';'`)
  Línea 3..: Posición y color de las celdas no blancas

  Por ejemplo:

```
Línea 1: 7 10 4
Línea 2: DDII 3 3 > ; IDID 5 7 v
Línea 3: 3 4 1
Línea 4: 3 5 2
Línea 5: 5 5 3
```

```
  0123456789
  ----------
0|          |
1|          |
2|          |
3|   >12    |
4|          |
5|     3 v  |
6|          |
```

- El programa principal recibe por línea de comandos el nombre de un fichero con los datos de inicialización, que pasa al constructor de simulador para que construya e inicialice la cinta y las hormigas. Antes de terminar el programa se pregunta al usuario si desea salvar el estado de la simulación en un fichero de texto con el formato dado.

Durante las sesiones de laboratorio se podrán proponer modificaciones y mejoras en el enunciado de la práctica.

5. Referencias

[1] Práctica 1. AyEDA curso 2025-2026:
https://campusvirtual.ull.es/2526/ingenieriaytecnologia/mod/resource/view.php?id=28424

[2] Gale, D.; Propp, J.; Sutherland, S.; Troubetzkoy, S. (1995). «Further Travels with my Ant». Mathematical Intelligencer 17: 48-56.

[3] Secuencias de escape ANSI. Librería de colores [GitHub]:
https://gist.github.com/Alfonzzoj/db207b89d56f24d9d0b17ff93e091be8

[4] Enum class [cppreference.com]: https://en.cppreference.com/w/cpp/language/enum.html