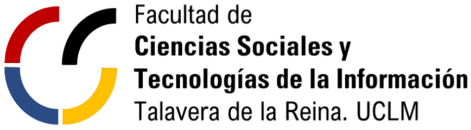

Grado en Ingeniería Informática

Fundamentos de Programación II 25/26

Sesión 4:
Proyecto práctico: Hito 1

Resumen

El objetivo de esta práctica es afianzar los conocimientos adquiridos sobre creación y manejo de clases y objetos, aplicando un flujo completo de análisis, diseño e implementación orientado a objetos. El trabajo se plantea por parejas y se centra en la construcción del núcleo de un videojuego RPG (Role-Playing Game).

1. Objetivos

- Aplicar los fundamentos de programación orientada a objetos en un caso práctico realista.
- Diseñar e implementar un sistema modular con clases, relaciones y responsabilidades bien definidas.
- Practicar el uso de arrays de tamaño fijo y lógica de control de flujo en métodos.
- Consolidar el proceso de trabajo por fases: análisis, diseño e implementación.

2. Normas de trabajo

El ejercicio se realizará por parejas y se entregará en la tarea habilitada en Campus Virtual. Se deberán seguir estas normas:

- **Entrega**
  - Todos los archivos deberán comprimirse en un único archivo ZIP.
  - El nombre del ZIP debe seguir el formato `LX_XX.zip` según el grupo de Campus Virtual.
  - El ZIP deberá incluir:
    - Imagen del diagrama de clases y relaciones UML (`.png` o `.jpg`).
    - Únicamente la carpeta `src` con los archivos `.java`.
- **Restricciones de implementación**
  - No utilizar múltiples `return` en una misma función.
  - No utilizar `continue`.
  - No utilizar `break`, salvo en sentencias `switch`.
- Queda totalmente **prohibido el uso de Inteligencia Artificial** para realizar estas prácticas. Cualquier indicio de su utilización será penalizado.

3. Fases del desarrollo orientado a objetos

Se pide seguir las fases del desarrollo de software orientado a objetos:

- **Análisis:** En esta fase se identifican los requisitos del sistema. Intentamos encontrar los objetos y relaciones adecuados para el problema propuesto. Investigamos el problema centrándonos en los conceptos del dominio del problema, sin preocuparnos por la implementación. Es importante entender el problema y definir claramente los requisitos antes de pasar a la siguiente fase.
- **Diseño:** En esta fase se define la arquitectura del sistema. Partiendo de los objetos y relaciones identificados en el análisis, se diseña la estructura del sistema. Se definen las clases, sus atributos y métodos (aún sin pensar en la implementación, solo en aquella funcionalidad que cada clase debe tener), así como las relaciones entre ellas (herencia, composición, etc.). Es importante diseñar un sistema modular y flexible que permita cambios futuros sin afectar a toda la aplicación. En esta fase es fundamental el uso de diagramas UML (especialmente el diagrama de clases).
- **Implementación:** Únicamente cuando tenemos claro el diseño, pasamos a la implementación. En esta fase se escribe el código fuente de la aplicación siguiendo el diseño establecido. Es importante seguir buenas prácticas de programación y mantener el código limpio y organizado.

4. Desarrollo de la práctica

En esta práctica aplicaréis los conceptos básicos de la programación orientada a objetos mediante el diseño, la implementación y el análisis de un videojuego RPG. Con este ejercicio, reforzaréis la creación y uso de clases, el manejo de arrays de tamaño fijo y la implementación de la lógica de control de flujo en los métodos. La principal interacción será a través de un menú donde podremos ver la información del sistema y realizar las acciones requeridas.

Imaginad que estáis empezando a desarrollar el núcleo interno de un videojuego RPG. En esta primera etapa vamos a disponer de un personaje que tendrá un nombre y podrá almacenar diferentes objetos en su inventario. En esta primera prueba de concepto del videojuego se implementarán los objetos metal y arma, que comparten algunas propiedades como: nombre, precio y peso. En el caso del metal, se caracteriza por su dureza, pureza y composición: cobre, acero, carbono, etc. (normalmente expresado por su símbolo químico). Cada arma se caracteriza por tener diferente ataque, longitud, antigüedad y, en un futuro este arma se podrá degradar, afectando su integridad. Cada personaje podrá tener equipada, un solo arma, pero podrá disponer de múltiples armas y metales en su inventario.

Desde el menú tendremos que poder ver la información básica del personaje, explorar su inventario, equipar un arma del inventario o desequipar su arma y meterla en el inventario para que, si en un futuro hay que diseñar zonas libres de conflicto, se tenga la posibilidad de que el personaje no tenga un arma equipada.

4.1. Formato del fichero `personaje.txt`

Para facilitar futuras pruebas y ampliaciones se proporcionará un archivo de entrada con formato fijo:

- Primera línea: datos del personaje (si empieza por `PERSONAJE`).
- Segunda línea: datos del arma equipada (si empieza por `ARMA`).
- Líneas siguientes: objetos del inventario (si empiezan por `INV`).

Ejemplo:

```
PERSONAJE;Alba;8
ARMA;EspadaCorta;50;3;12;95.5;3;88
INV;ARMA;HachaGrande;80;6;18;120.0;12;74
INV;METAL;LingoteHierro;25;4;7;75;Fe
INV;METAL;LingoteCobre;20;3;4;80;Cu
```