Paradigmas de Programación

**Trabajo Práctico N° 2**

El presente trabajo práctico tiene como objetivo aplicar los conceptos, principios y características del paradigma de la programación orientada a objetos (Clases, objetos, métodos, interfaz, herencia, polimorfismo, colecciones, etc).

**Entrega intermedia: 27 de Mayo de 2025 (diagrama de clases tentativo)**

**Fecha de entrega final:  1 y 8 de Julio de 2025**

Objetivo

Implementar los conceptos abordados del paradigma de la Programación Orientada a Objetos, como solución a un problema real, partiendo del análisis de la problemática, el diseño de la solución (Diagrama de Clases) y posteriormente la construcción del producto final (Software).

Objetivos específicos

* Aplicar los conceptos fundamentales del paradigma orientado a objetos en un contexto lúdico y concreto.
* Analizar una problemática real y modelarla a través de clases, relaciones y responsabilidades.
* Diseñar un sistema extensible, con bajo acoplamiento y alto nivel de cohesión.
* Trabajar con colecciones, estructuras de datos dinámicas y archivos externos.
* Realizar pruebas automáticas que verifiquen el correcto comportamiento del sistema ante diversos escenarios.

Consigna

Temática

Modelar un sistema de crafteo de objetos inspirado en los videojuegos, aplicando principios de programación orientada a objetos. Se busca representar recetas de crafteo, ingredientes básicos, objetos intermedios y un inventario del jugador, sin utilizar estructuras de grafos. Además, se espera que la solución pueda adaptarse fácilmente a la incorporación de nuevos objetos, ingredientes o reglas.

**Craftear** es el proceso de combinar distintos elementos o materiales para crear nuevos objetos. Es una mecánica común en videojuegos, donde los jugadores deben reunir recursos básicos y usarlos según recetas para fabricar herramientas, armas, estructuras u otros ítems útiles.

En este sistema, algunos objetos pueden *craftearse* combinando otros objetos, y otros son **elementos básicos**, que no se pueden fabricar y deben recolectarse. Cada receta indica qué cantidad de cada ingrediente se necesita, y que cantidad de objetos devuelve. Por ejemplo: Se puede necesitar 2 maderas y 1 carbón para producir 3 antorchas en 6 minutos

El jugador tiene un **inventario** con cantidades de distintos objetos. A partir de esto, se deben responder consultas relacionadas con el sistema de crafteo.

Glosario

* **Craftear:** combinar elementos para fabricar nuevos objetos.
* **Receta:** conjunto de ingredientes y cantidades necesarias para fabricar algo.
* **Ingrediente básico:** recurso recolectable, no fabricable.
* **Objeto intermedio:** objeto que se fabrica y luego se puede usar como parte de otra receta.
* **Inventario:** listado de objetos y cantidades que posee el jugador.

Descripción General

Se espera que el sistema permita responder a las siguientes funcionalidades:

**1. ¿Qué necesito para craftear un objeto?**

Dado un objeto crafteable, mostrar la lista de ingredientes y cantidades necesarias (sólo el primer nivel de la receta, sin descomponer los ingredientes).

**2. ¿Qué necesito para craftear un objeto desde cero?**

Dado un objeto crafteable, mostrar todos los elementos básicos necesarios, con sus cantidades totales, considerando la descomposición completa de sus ingredientes en elementos básicos.

**3. ¿Qué me falta para craftear un objeto?**

Dado un objeto y un inventario, indicar qué ingredientes y en qué cantidad faltan para poder craftearlo (primer nivel solamente).

**4. ¿Qué me falta para craftear un objeto desde cero?**

Lo mismo que el punto 3, pero considerando los elementos básicos necesarios para toda la cadena de crafteo.

**5. ¿Cuántos puedo craftear?**

Dado un objeto y un inventario, indicar cuántas unidades del objeto se pueden fabricar utilizando los elementos del inventario (y fabricando los ingredientes intermedios si es necesario).

**6. Realizar el crafteo indicado**

El inventario dispone de ciertos elementos y objetos en principio, y debe poderse ejecutar un crafteo. Esto es una acción que modifica el contenido del inventario, el cual debe actualizarse en consecuencia.

**7. En todos los casos, indicar los tiempos necesarios**

Craftear, por su naturaleza, requiere de un tiempo. No es instantáneo, y esos tiempos deben considerarse e informarse en cada una de las preguntas anteriores. Al realizar crafteos en cadena, los tiempos deben sumarse, y multiplicarse apropiadamente de acuerdo a la cantidad de unidades involucradas.

**8. Historial de crafteos**

Registrar cada objeto que se ha crafteado, con sus ingredientes usados y la fecha o turno de creación.

Requisitos técnicos

* No utilizar grafos.
* Utilizar clases, objetos y relaciones entre ellos.
* Aplicar principios de encapsulamiento, composición, y responsabilidad única.
* Utilizar pruebas automatizadas y métodos de verificación simples que permitan testear los casos anteriores.
* Se debe poder agregar nuevas recetas y objetos sin modificar el código existente (principio de abierto/cerrado).
* No utilizar grafos.

Integración con Prolog

Realizar la integración con Prolog, para poder responder la siguiente pregunta:

¿Cuáles son todos los productos que podría generar con el inventario actual?  
El sistema debe traducir el inventario actual a hechos en Prolog (tengo/2) y definir las reglas y recetas (ingrediente/3). Prolog responderá con la lista de objetos posibles, deduciendo automáticamente a partir de las recetas y cantidades disponibles.

**Ejemplo (NO resuelve el problema planteado):**

% Hechos

ingrediente(bastón, madera, 2).

ingrediente(espada, hierro, 3).

ingrediente(espada, bastón, 1).

elemento\_basico(madera).

elemento\_basico(hierro).

% Inventario

tengo(madera, 4).

tengo(hierro, 6).

% Reglas

puedo\_craftear(Objeto) :-

    ingrediente(Objeto, Ing, Cant),

    tengo(Ing, CantDisponible),

    CantDisponible >= Cant.

Datos de origen

Deben utilizarse, al menos, dos archivos para ingresar la información necesaria al sistema. La misma **no debe estar codificada, sino que debe usarse dependiendo del archivo ingresado**.

1. Un archivo recetas.json (o XML), que describa los elementos y los ítems, indicando también los ingredientes que lo forman (si no fueran elementos básicos). Es decir, todo lo necesario para tener la información para operar.
2. Un archivo inventario.json (o XML), que especifique los elementos presentes en el inventario inicial del jugador.
3. Bonus (\*)(1 punto): al cerrar el programa, crear un archivo inventario-out.json (o XML), que especifique los elementos presentes en el inventario final del jugador.

Pruebas

Crear una batería de pruebas para cada pregunta funcional, así como también para la variación del inventario conforme se realicen los crafteos.

Utilizar JUnit para esto, y validar que los crafteos se ejecuten correctamente. Se espera una cobertura de pruebas adecuada para todas las funcionalidades descritas, incluyendo tanto casos exitosos como escenarios con errores o restricciones.

Bonus

Los bonus consisten en objetivos opcionales que potenciarán la nota del grupo. La nota del TP con todo lo anterior cumplido es de 8 (si todo está perfecto). A ello pueden sumar uno o varios de estos bonus para aumentar la nota (6 como máximo). Los pueden utilizar para compensar alguna otra funcionalidad que no haya quedado terminada completamente, o para llevar su nota a un 10.

**1. Mostrar el árbol de crafteos (1 punto)**

Mediante una interfaz de texto, mostrar la forma de realizar un crafteo determinado, informando adecuadamente cada paso necesario.

**2. Recetas alternativas o variantes (1 punto)**

Permitir que un mismo objeto pueda craftearse con diferentes combinaciones de ingredientes (por ejemplo, una antorcha puede hacerse con carbón mineral o carbón vegetal).

**3. Inventario final, previamente mencionado (1 punto)**

**4. Catalizadores (3 puntos)**

Los catalizadores son elementos adicionales que permiten optimizar el proceso de fabricación: pueden reducir la cantidad de materiales base necesarios o aumentar la cantidad de materiales obtenidos como resultado. Cada catalizador está vinculado a un tipo específico de receta (por ejemplo, los catalizadores de fuego solo pueden utilizarse con recetas de tipo fuego). Existe un tipo de catalizador por cada tipo de receta, y el inventario puede contener desde ninguno hasta múltiples ejemplares de cada uno. Importante: los catalizadores no pueden ser utilizados como ingredientes en la creación de recetas.

**5. Mesas de trabajo (3 puntos)**

Las mesas de trabajo son herramientas opcionales que amplían el repertorio de recetas disponibles para fabricar. Cada mesa desbloquea entre 1 y N recetas adicionales. Deben existir múltiples tipos de mesas, y su presencia o ausencia en el inventario (0 o 1 por tipo) afecta directamente qué recetas están disponibles, modificando así el comportamiento general del sistema. Este ítem puede combinarse con el punto bonus 2, pero no es compatible con la mecánica de catalizadores (punto 4).

(+1 pto) Si tuviera los ingredientes, puedo craftear mesas de trabajo.

Interfaz

Se evaluará el correcto funcionamiento del sistema utilizando dos enfoques simultáneos:

1. Se deberá probar cada una de las funcionalidades desarrolladas, utilizando JUnit correctamente
2. Se podrá ejecutar un main, en el cual se crean los elementos y se realizan los crafteos de forma automática.

Sugerimos tener varios escenarios preparados, para mostrar todas las características desarrolladas.

Entrega

El trabajo se realizará en forma grupal, con equipos de 4 a 6 personas.

Se espera un informe que contenga al menos: carátula, índice, introducción, desarrollo, conclusiones y referencias (APA).

Habrá una defensa y presentación oral en la fecha designada, y una serie de rúbricas se aplicarán oportunamente.

Más información

Grow, A. et al. 2017. Crafting in Games  
[https://www.digitalhumanities.org/dhq/vol/11/4/000339/000339.html](https://www.google.com/url?q=https://www.digitalhumanities.org/dhq/vol/11/4/000339/000339.html&sa=D&source=editors&ust=1751142068561573&usg=AOvVaw2igD6FioDOb2SRey_tPqd8)