Análisis y Diseño de Software, Curso 2023-2024

Práctica 3

Introducción a la Programación Orientada a Objetos con Java

Inicio: A partir del 26 de Febrero.
Duración: 3 semanas.
Entrega: En Moodle, una hora antes del comienzo de la siguiente práctica según grupos (semana del 18 de Marzo)
Peso de la práctica: 20%

El objetivo de la práctica es introducir al alumno en la programación orientada a objetos con el lenguaje Java. Se pide desarrollar de forma incremental varias clases en Java, incluyendo sus pruebas y documentación. Las clases implementarán funcionalidades relacionadas con un planificador de menús según distintos criterios nutricionales.

En el desarrollo de la práctica se utilizarán principalmente los siguientes conceptos de Java:

- *tipos de datos primitivos, String, arrays, listas, mapas*, *clases y enumerados* definidos por el programador
- *clases* sencillas definidas por el programador para implementar objetos con *atributos (o variables) de instancia y de clase, métodos de instancia, métodos de clase,* y *constructores*
- *entrada/salida elemental* para la lectura de archivos en formato texto y la visualización de texto en la consola
- *herencia, sobrescritura de métodos*
- *comentarios para documentación automática mediante javadoc*

Apartado 1. Ingredientes e información nutricional (2 puntos)

Comenzaremos creando la parte de la aplicación para representar ingredientes. Un ingrediente tiene un nombre, un tipo (carne, pescado, verdura/fruta, legumbre, cereal, huevo, lácteo, u otros), y una información nutricional (calorías, hidratos de carbono, grasas totales, grasas saturadas, proteínas, azúcares, fibra y sodio). La información nutricional de un ingrediente puede darse por cada 100 gramos o por unidad, pero sólo de una forma. Por ejemplo, la información nutricional del ingrediente pasta se daría por peso, y la del ingrediente tomate por unidad. Además, un ingrediente puede tener cero o más de los siguientes alérgenos: gluten, lactosa, huevos o frutos secos.

A modo de ejemplo, el siguiente programa crea varios ingredientes y los muestra por pantalla. Los constructores de las clases `InfoNutricionalPeso` e `InfoNutricionalUnidad` reciben los parámetros en este orden: calorías, hidratos de carbono, grasa total, grasa saturada, proteínas, azúcares, fibra, y sodio.

```
public class IngredientesTester {
   public static void main (String args[]) {
        IngredientesTester tester = new IngredientesTester();
        for (Ingrediente ingrediente : tester.crearIngredientes().values())
           System.out.println("* " + ingrediente);
   }

   public Map<String, Ingrediente> crearIngredientes() {
        Map<String, Ingrediente> ingredientes = new LinkedHashMap<>();
        ingredientes.put("Pasta", new Ingrediente("Pasta", TipoIngrediente.CEREAL,
                                        new InfoNutricionalPeso(372, 74, 1.8, 0.277,12, 2.6, 2.9, 6))
                                    .tieneAlergenos(Alergeno.GLUTEN, Alergeno.HUEVO));
        ingredientes.put("Tomate", new Ingrediente("Tomate", TipoIngrediente.FRUTA_VERDURA,
                                        new InfoNutricionalUnidad(14, 2.2, 0.2, 0, 0.7, 2.04, 1, 4)));
        ingredientes.put("Aceite", new Ingrediente("Aceite", "Grasa" /*otro tipo*/,
                                        new InfoNutricionalPeso(885, 0, 100, 12.81, 0, 0, 0, 2)));
        ingredientes.put("Huevo", new Ingrediente("Huevo", TipoIngrediente.HUEVO,
                                        new InfoNutricionalUnidad(84.6, 0.35, 6.3, 0.2, 6.6, 0.6, 0, 0.1))
                                    .tieneAlergenos(Alergeno.HUEVO));
        ingredientes.put("Chorizo", new Ingrediente("Chorizo", TipoIngrediente.CARNE,
                                        new InfoNutricionalPeso(203, 5, 14.3, 4.6, 13.6, 0, 2, 800))
                                    .tieneAlergenos(Alergeno.LACTOSA));
        ingredientes.put("Patata", new Ingrediente("Patata", TipoIngrediente.FRUTA_VERDURA,
                                        new InfoNutricionalPeso(85, 17.6, 0.1, 0, 2, 0, 2.6, 2)));
        ingredientes.put("Caldo", new Ingrediente("Caldo", "Caldo" /*otro tipo*/,
                                        new InfoNutricionalPeso(267, 18, 14, 3.4, 17, 17, 0, 23.875)));
        return ingredientes;
   }
}
```

Se pide: Crear todo el código necesario para que el programa anterior produzca la siguiente salida:

```
* [Cereal] Pasta: INFORMACION NUTRICIONAL POR 100 g -> Valor energetico: 372.00 kcal, Hidratos de carbono: 74.00 g, Grasas: 1.80 g, Saturadas: 0.28 g, Proteinas: 12.00 g, Azucares: 2.60 g, Fibra: 2.90 g, Sodio: 6.00 mg. CONTIENE gluten, huevo
```

```
* [Frutas y verduras] Tomate: INFORMACION NUTRICIONAL POR UNIDAD -> Valor energetico: 14.00 kcal, Hidratos de carbono: 2.20 g, Grasas: 0.20 g, Saturadas: 0.00 g, Proteinas: 0.70 g, Azucares: 2.04 g, Fibra: 1.00 g, Sodio: 4.00 mg.
* [Grasa] Aceite: INFORMACION NUTRICIONAL POR 100 g -> Valor energetico: 885.00 kcal, Hidratos de carbono: 0.00 g, Grasas: 100.00 g, Saturadas: 12.81 g, Proteinas: 0.00 g, Azucares: 0.00 g, Fibra: 0.00 g, Sodio: 2.00 mg.
* [Huevo] Huevo: INFORMACION NUTRICIONAL POR UNIDAD -> Valor energetico: 84.60 kcal, Hidratos de carbono: 0.35 g, Grasas: 6.30 g, Saturadas: 0.20 g, Proteinas: 6.60 g, Azucares: 0.60 g, Fibra: 0.00 g, Sodio: 0.10 mg. CONTIENE huevo
* [Carne] Chorizo: INFORMACION NUTRICIONAL POR 100 g -> Valor energetico: 203.00 kcal, Hidratos de carbono: 5.00 g, Grasas: 14.30 g, Saturadas: 4.60 g, Proteinas: 13.60 g, Azucares: 0.00 g, Fibra: 2.00 g, Sodio: 800.00 mg. CONTIENE lactosa
* [Frutas y verduras] Patata: INFORMACION NUTRICIONAL POR 100 g -> Valor energetico: 85.00 kcal, Hidratos de carbono: 17.60 g, Grasas: 0.10 g, Saturadas: 0.00 g, Proteinas: 2.00 g, Azucares: 0.00 g, Fibra: 2.60 g, Sodio: 2.00 mg.
* [Caldo] Caldo: INFORMACION NUTRICIONAL POR 100 g -> Valor energetico: 267.00 kcal, Hidratos de carbono: 18.00 g, Grasas: 14.00 g, Saturadas: 3.40 g, Proteinas: 17.00 g, Azucares: 17.00 g, Fibra: 0.00 g, Sodio: 23.88 mg.
```

Apartado 2. Platos (2 puntos)

En este apartado se tratará la definición de platos, que tienen en su composición uno o más ingredientes y/o platos. Para cada ingrediente de un plato hay que indicar su peso en gramos o el número de unidades, en función de cómo esté definido el ingrediente. Para el caso de platos que forman parte de otros platos, no hay que indicar nada (se considera que se añade 1 unidad del plato). Un plato no puede tener dos elementos repetidos. La información nutricional y los alérgenos de un plato se calculan a partir de los ingredientes y platos que lo forman.

A modo de ejemplo, el siguiente programa crea varios platos y los muestra por pantalla.

```java
public class PlatosTester extends IngredientesTester {
   public static void main(String[] args) {
        PlatosTester tester = new PlatosTester();
        for (Plato plato : tester.crearPlatos().values())
          System.out.println("* " + plato);
   }

   public Map<String, Plato> crearPlatos() {
        Map<String, Ingrediente> ing = this.crearIngredientes();
        Plato p1, p2, p3;
        p1 = new Plato("Macarrones");
        if (p1.addIngrediente(ing.get("Pasta"), 90)) System.out.println("ingrediente repetido");
        if (p1.addIngrediente(ing.get("Pasta"), 90)) System.out.println("ingrediente repetido");
        if (p1.addIngrediente(ing.get("Tomate"), 4)) System.out.println("ingrediente repetido");
        if (p1.addIngrediente(ing.get("Tomate"), 4)) System.out.println("ingrediente repetido");
        p1.addIngrediente(ing.get("Aceite"), 10);
        p1.addIngrediente(ing.get("Chorizo"), 30);
        p2 = new Plato("Tortilla");
        p2.addIngrediente(ing.get("Huevo"), 2);
        p2.addIngrediente(ing.get("Patata"), 150);
        p2.addIngrediente(ing.get("Aceite"), 10);
        p3 = new Plato("Tortilla guisada");
        p3.addPlato(p2);
        p3.addIngrediente(ing.get("Caldo"), 80);
        return Map.of("Macarrones", p1, "Tortilla", p2, "Tortilla guisada", p3);
   }
}
```

Se pide: Crear todo el código necesario para que el programa anterior produzca la siguiente salida:

```
ingrediente repetido
ingrediente repetido
* [Plato] Tortilla: INFORMACION NUTRICIONAL DEL PLATO -> Valor energetico: 385.20 kcal, Hidratos de carbono: 27.10 g, Grasas: 22.75 g, Saturadas: 1.68 g, Proteinas: 16.20 g, Azucares: 1.20 g, Fibra: 3.90 g, Sodio: 3.40 mg. CONTIENE huevo
* [Plato] Tortilla guisada: INFORMACION NUTRICIONAL DEL PLATO -> Valor energetico: 598.80 kcal, Hidratos de carbono: 41.50 g, Grasas: 33.95 g, Saturadas: 4.40 g, Proteinas: 29.80 g, Azucares: 14.80 g, Fibra: 3.90 g, Sodio: 22.50 mg. CONTIENE huevo
* [Plato] Macarrones: INFORMACION NUTRICIONAL DEL PLATO -> Valor energetico: 540.20 kcal, Hidratos de carbono: 76.90 g, Grasas: 16.71 g, Saturadas: 2.91 g, Proteinas: 17.68 g, Azucares: 10.50 g, Fibra: 7.21 g, Sodio: 261.60 mg. CONTIENE gluten, huevo, lactosa
```

Nota: Cuando añadas estas clases, no dudes en reorganizar el código del apartado 1 si es necesario, para seguir buenas prácticas de orientación a objetos.

Apartado 3. Menús (2 puntos)

En este apartado se pide crear la clase `Menú`, que contiene uno o más platos. Al crear un menú, se le asignará automáticamente un identificador numérico único, que no puede modificarse una vez creado. A continuación tienes un programa de ejemplo que crea varios menús y muestra su composición, información nutricional y alérgenos por pantalla.

```
public class MenusTester extends PlatosTester {
	public static void main(String[] args) {
		MenusTester tester = new MenusTester();
		for (Menu menu : tester.crearMenus())
			System.out.println("* " + menu);
	}
	public List<Menu> crearMenus() {
		Map<String, Plato> platos = this.crearPlatos();
		Menu m1 = new Menu(platos.get("Macarrones"),platos.get("Tortilla"));
		Menu m2 = new Menu(platos.get("Macarrones"),platos.get("Tortilla guisada"));
		Menu m3 = new Menu(platos.get("Macarrones"));
		return List.of(m1, m2, m3);
	}
}
```

Se pide: Crear todo el código necesario para que el programa anterior produzca la siguiente salida:

```
ingrediente repetido
ingrediente repetido
* Menu 1 [Macarrones, Tortilla]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 925.40 kcal, Hidratos de carbono: 104.00 g, Grasas: 39.46 g, Saturadas: 4.59 g, Proteinas: 33.88 g, Azucares: 11.70 g, Fibra: 11.11 g, Sodio: 265.00 mg. CONTIENE gluten, huevo, lactosa
* Menu 2 [Macarrones, Tortilla guisada]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 1139.00 kcal, Hidratos de carbono: 118.40 g, Grasas: 50.66 g, Saturadas: 7.31 g, Proteinas: 47.48 g, Azucares: 25.30 g, Fibra: 11.11 g, Sodio: 284.10 mg. CONTIENE gluten, huevo, lactosa
* Menu 3 [Macarrones]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 540.20 kcal, Hidratos de carbono: 76.90 g, Grasas: 16.71 g, Saturadas: 2.91 g, Proteinas: 17.68 g, Azucares: 10.50 g, Fibra: 7.21 g, Sodio: 261.60 mg. CONTIENE gluten, huevo, lactosa
```

Apartado 4. Ficheros (2 puntos)

Extiende tu implementación para poder leer ingredientes, platos y menús de un fichero de texto; y para poder guardar una lista de menús a fichero. Cuando se guarde un menú a fichero, también deberán guardarse los platos e ingredientes que lo componen, directa o indirectamente. Además, el fichero no podrá contener ingredientes, platos ni menús repetidos. En los ficheros, cada ingrediente se definirá en 1 línea distinta con el siguiente formato:

`(INGREDIENTE_PESO|INGREDIENTE_UNIDAD);NOMBRE;TIPO;CALORIAS;HIDRATOS;GRASAS_TOTALES;GRASAS_SATURADAS;PROTEINAS;AZUCARES;FIBRA;SODIO;GLUTEN;LACTOSA;FRUTOS_SECOS;HUEVO`

Cada plato se definirá en 1 línea distinta con el siguiente formato:

`PLATO;nombre(;INGREDIENTE nombre:cantidad|PLATO nombre)+`

Cada menú se definirá en 1 línea distinta con el siguiente formato: `MENU(;plato)+`

El siguiente listado muestra un fichero con formato válido:

```
INGREDIENTE_PESO;Pasta;CEREAL;372.0;74.0;1.8;0.277;12.0;2.6;2.9;6.0;S;N;N;S
INGREDIENTE_UNIDAD;Tomate;FRUTA_VERDURA;14.0;2.2;0.2;0.0;0.7;2.04;1.0;4.0;N;N;N;N
INGREDIENTE_PESO;Aceite;Grasa;885.0;0.0;100.0;12.81;0.0;0.0;0.0;2.0;N;N;N;N
INGREDIENTE_PESO;Chorizo;CARNE;203.0;5.0;14.3;4.6;13.6;0.0;2.0;800.0;N;S;N;N
PLATO;Macarrones;INGREDIENTE Pasta:90;INGREDIENTE Tomate:4;INGREDIENTE Aceite:10;INGREDIENTE Chorizo:30
INGREDIENTE_UNIDAD;Huevo;HUEVO;84.6;0.35;6.3;0.2;6.6;0.6;0.0;0.1;N;N;N;S
INGREDIENTE_PESO;Patata;FRUTA_VERDURA;85.0;17.6;0.1;0.0;2.0;0.0;2.6;2.0;N;N;N;N
PLATO;Tortilla;INGREDIENTE Huevo:2;INGREDIENTE Patata:150;INGREDIENTE Aceite:10
MENU;Macarrones;Tortilla
INGREDIENTE_PESO;Caldo;Caldo;267.0;18.0;14.0;3.4;17.0;17.0;0.0;23.875;N;N;N;N
PLATO;Tortilla guisada;PLATO Tortilla;INGREDIENTE Caldo:80
MENU;Macarrones;Tortilla guisada
MENU;Macarrones
```

A continuación tienes un programa de prueba y la salida esperada, que generaría un fichero similar al anterior.

```
public class FicherosTester extends MenusTester {
	public static void main(String[] args) {
		FicherosTester tester = new FicherosTester();
		List<Menu> menus = tester.crearMenus();
		// guardar lista de menús a fichero
		ManejadorFicheros.guardarFichero("comida.txt", menus);
		// leer lista de menús de fichero, e imprimirlos por pantalla
		ManejadorFicheros.leerFichero("comida.txt");
		for (Menu menu : ManejadorFicheros.getMenus())
			System.out.println("* " + menu);
	}
}
```

Se pide: Crear todo el código necesario para que el programa anterior produzca la siguiente salida:

```
ingrediente repetido
ingrediente repetido
* Menu 4 [Macarrones, Tortilla]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 925.40 kcal, Hidratos de carbono: 104.00 g, Grasas: 39.46 g, Saturadas: 4.59 g, Proteinas: 33.88 g, Azucares: 11.70 g, Fibra: 11.11 g, Sodio: 265.00 mg. CONTIENE gluten, huevo, lactosa
* Menu 5 [Macarrones, Tortilla guisada]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 1139.00 kcal, Hidratos de carbono: 118.40 g, Grasas: 50.66 g, Saturadas: 7.31 g, Proteinas: 47.48 g, Azucares: 25.30 g, Fibra: 11.11 g, Sodio: 284.10 mg. CONTIENE gluten, huevo, lactosa
* Menu 6 [Macarrones]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 540.20 kcal, Hidratos de carbono: 76.90 g, Grasas: 16.71 g, Saturadas: 2.91 g, Proteinas: 17.68 g, Azucares: 10.50 g, Fibra: 7.21 g, Sodio: 261.60 mg. CONTIENE gluten, huevo, lactosa
```

Apartado 5. Planificador de menús (2 puntos)

Implementa un planificador de menús que sea capaz de crear automáticamente un menú que cumpla un conjunto de criterios nutricionales dados. El planificador recibirá la lista de platos disponibles para crear el menú; un rango de calorías (ej. 800-2500 kcal); si se deben excluir ciertos alérgenos (ej. sin gluten y sin huevo); y puede indicar un valor máximo para algunos valores nutricionales (ej. menos de 20 g. de grasa saturada, y menos de 10 g. de sodio). El algoritmo de creación del menú se deja a elección del alumno, teniendo en cuenta que no se va a evaluar su eficiencia, pero sí que el resultado sea correcto y se siga un buen diseño orientado a objetos. Si no es posible construir el menú (ej. porque se ha pedido un menú sin gluten, pero no hay platos sin gluten), el planificador devolverá un valor nulo como resultado.

El siguiente programa ilustra el uso del planificador pedido, y un posible menú de salida (puede haber otros).

```java
public class PlanificadorTester extends PlatosTester {
	public static void main (String[] args) {
		PlanificadorTester tester = new PlanificadorTester();
		List<Plato> platos = new ArrayList<>(tester.crearPlatos().values());
		PlanificadorMenu planificador =
				new PlanificadorMenu(platos)
					.conMaximo(ElementoNutricional.GRASA_SATURADA, 20.0)
					.conMaximo(ElementoNutricional.AZUCARES, 15.0)
					.sinAlergenos(Alergeno.GLUTEN, Alergeno.HUEVO);
		// busqueda 1
		Menu menu = planificador.planificar(800, 2500);
		System.out.println("* "+menu);
		// busqueda 2
		planificador = new PlanificadorMenu(platos)
					.conMaximo(ElementoNutricional.GRASA_SATURADA, 20.0)
					.conMaximo(ElementoNutricional.AZUCARES, 15.0)
					.sinAlergenos(Alergeno.FRUTOS_SECOS);
		menu = planificador.planificar(800, 2500);
		System.out.println("* "+menu);
	}
}
```

Se pide: Crear todo el código necesario para que el programa anterior produzca la siguiente salida (o equivalente):

```
ingrediente repetido
ingrediente repetido
* null
* Menu 15 [Tortilla, Macarrones]: INFORMACION NUTRICIONAL DEL MENU -> Valor energetico: 925.40 kcal, Hidratos de carbono: 104.00 g, Grasas: 39.46 g, Saturadas: 4.59 g, Proteinas: 33.88 g, Azucares: 11.70 g, Fibra: 11.11 g, Sodio: 265.00 mg. CONTIENE huevo, gluten, lactosa
```

Comentarios adicionales

- No olvides que, además del correcto funcionamiento de la práctica, un aspecto fundamental en la evaluación será la **calidad del diseño**. Tu diseño debe utilizar los principios de orientación a objetos, además de ser claro, fácil de entender, extensible y flexible. Presta atención a la calidad del código, evitando redundancias.
- Organiza el código en **paquetes**.
- Crea **programas de prueba** para comprobar el correcto funcionamiento del código de cada apartado, y entrégalos.

Normas de entrega

- Se debe entregar el **código Java** de los apartados, la **documentación** generada con *javadoc*, los **programas de prueba** creados, y un **diagrama de diseño** de toda la práctica (en PDF) junto con una breve explicación.
- El nombre de los alumnos debe ir en la cabecera *javadoc* de todas las clases entregadas.
- La entrega la realizará uno de los alumnos de la pareja a través de Moodle.
- Se debe entregar un único fichero ZIP / RAR con todo lo solicitado, que debe llamarse de la siguiente manera: `GR<numero_grupo>_<nombre_estudiantes>.zip`. Por ejemplo, Marisa y Pedro, del grupo 2261, entregarían el fichero: `GR2261_MarisaPedro.zip`.