

Alumno/a:

La tienda de frutas tropicales “FrutasRibera” prepara los pedidos que les llegan cada día a través de su web y los envían a los clientes en 24h.

Los pedidos los deben preparar los empleados de la tienda a través de una aplicación de escritorio que usa unos ficheros generados por la página web llamados **Pedidos.csv** y **ProductosPedidos.csv** y también se necesita información que se almacena en una BD Oracle.

El objetivo es repartir los pedidos entre los empleados según el turno en el que lleguen y generar los ficheros con la factura detallada de la compra que luego se imprimirá y se adjuntará con el pedido en el envío al cliente.

Para crear la aplicación debes seguir los siguientes pasos:

1. Crea la jerarquía de clases necesaria para implementar la herencia y poder reutilizar código con las siguientes clases:
  - Persona es una clase abstracta que tiene como atributos dni, nombre, apellidos, dirección.
  - Empleado: dni, nombre, apellidos, dirección, turno, **pedidosAsignados** es un Array list de pedidos
  - Cliente: dni, nombre, apellidos, dirección, métodoPago.

Otras clases que se necesitan son:

- ProductoPedido: codigoPedido(entero), codigoProducto(entero), nombreProducto, descripción, temporada, precioKg; kg
- Pedido: codigoPedido, dniCliente, descuento, turno(entero)
- Factura: codigoFactura(cadena), codigoPedido(entero), dniCliente, productosPedidos(Array list) y total(double)

La clase Factura.java implementa una interfaz llamada impuestos

```
public interface impuestos {  
    public double ivaReducido();  
    public double totalSinIVA();  
}
```

No olvides incluir en cada clase los constructores, getters, setters y toString.

Alumno/a:

## 2. En la clase Factura.java:

A) Crea el método cargarProductos() que lee los productos correspondientes a un pedido del fichero ProductosPedidos.csv y los devuelva en el array list.

```
public ArrayList<ProductoPedido> cargarProductos()
```

B) Programa los métodos que se implementan de la interfaz:

```
@Override  
public double totalSinIVA()
```

Devuelve la **suma total** de los productos pedidos sin sumar el IVA es decir, la suma del **precioKG\*KG**. Por ejemplo, si un pedido tiene dos productos que son aguacates y papaya, como los aguacates cuestan 5,98€/kg y se compran 3 kg el método debe devolver 17,94€ y a esto hay que sumar lo que vale 5,22€/kg por 4 kg de papaya que son 20,88€.

Es decir totalSinIVA() devuelve 17,94+20,88=38,82€

Aguacate	5.98€/kg	3.0kg
Papaya	5.22€/kg	4.0kg
		<b>TOTAL 38.82€</b>

Este valor total será el valor del atributo total del objeto de la clase Factura. No olvides actualizar el valor.

C) El siguiente método devuelve el 4% del valor del total de la factura que se ha calculado en el apartado A

```
@Override  
public double ivaReducido()
```

Para el caso anterior

Aguacate	5.98€/kg	3.0kg
Papaya	5.22€/kg	4.0kg
		<b>TOTAL 38.82€</b>
		<b>IVA 1.5528€</b>


Para que se vea parecido, el toString de la clase ProductoPedido puede ser:

```
return ("\t"+this.nombre+"\t"+this.precioKg+"€/kg\t"+this.kg+"kg");
```

Alumno/a:

### 3. Clase BD.java:

A) Modifica el método cargarParámetrosConexion() para que lea los parámetros del siguiente fichero:

 configTienda.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
jdbc:oracle:thin:@localhost:1521:XE-tiendaFruta-tiendaFruta
```

La cadena de conexión, el usuario y la contraseña están en una línea separados por el carácter “-”

B) Añade a la clase BD.java el siguiente método para cargar de la BD los empleados de la tienda que tienen que preparar los pedidos. El valor turno será 1 ó 2 o bien, distinto de 0

```
public ArrayList<Empleado>cargaEmpleados()
```

### 4. Clase Empleado:

A) Los empleados tienen un atributo llamado pedidosAsignados que es un ArrayList de pedidos. Crea el método

```
public void asignarPedidosTurno()
```

que a cada empleado le debe asignar los pedidos que se realizan en su turno siendo el turno 1 ó 2. Lee del fichero pedidos.csv y asigna a cada empleado los pedidos que tiene que preparar según el turno.

B) Cada empleado debe procesar los pedidos que se le han asignado por lo que debe invocar a generarFactura para cada pedido y realizar una inserción en la tabla factura de la BD que está definida como: CódigoFactura(es el string formado por codigoPedido\_dniCliente), codigoPedido, dniCliente, total(totalsinIVA+IVAReducido(4%)-descuento del pedido)

```
public Factura generarFactura(Pedido p)
```

La salida del método es un objeto de la clase Factura que se vuelca en un fichero de texto con el nombre FacturaDNI.txt y que incluye la siguiente información:

Alumno/a:

 Factura11111111K.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Factura: 1000\_11111111K

Cliente: 11111111K

\*\*\*\*\*Productos\*\*\*\*\*

1	Aguacate	5.98€/kg	3.0kg
2	Papaya	5.22€/kg	4.0kg

TOTAL 38.82€

IVA 1.5528€

Descuento 4.0€

IMPORTE TOTAL: 36.3728

que posteriormente se imprimirá y se enviará al cliente junto con su pedido.

La sentencia para insertar la factura en la BD es, para este ejemplo:

```
Insert into Factura values('1000_11111111K',1000,'11111111K',36.3728)
```

C) Añade a la clase BD.java el siguiente método para cargar de la BD los empleados de la tienda que tienen que preparar los pedidos

```
public ArrayList<Empleado>cargaEmpleados()
```

## 5. Clase App.java:

Al comenzar el programa carga los empleados en una lista con el método de cargaEmpleados() de la clase BD y asigna los pedidos que tiene que preparar según su turno con el método de la clase Empleado asignarPedidosTurno().

Según el parámetro que reciba el método main App hará lo siguiente:

A) Si el parámetro de main es

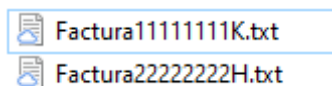
```
--empleado=DNI
```

Crea las facturas correspondientes a los pedidos que prepara ese empleado. Un ejemplo sería:

```
El empleado Carmen con DNI 61111111H prepara 2 pedidos
```

Alumno/a:

y se han generado los ficheros

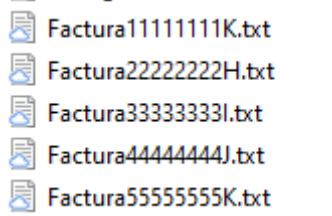


B) Si no hay ningún parámetro se generarán todas las facturas de todos los pedidos.

**El empleado Carmen con DNI 61111111H prepara 2 pedidos**

**El empleado Jose con DNI 15151515P prepara 3 pedidos**

Y se generan todos los ficheros



#### INSTRUCCIONES Y ACLARACIONES PARA LA REALIZACIÓN DE LA PRUEBA:

- Está permitido el uso de Internet y código fuente del alumno
- La entrega de la prueba será la carpeta del proyecto Java comprimida (toda la carpeta) y enviada al apartado correspondiente de fpribera.es

Según la NCOF del IES Ribera del Tajo la copia o utilización de medios ilícitos para la realización de las pruebas de evaluación, ya sean escritas o en cualquier otro formato, tanto de forma individual como colectiva se **considera una conducta gravemente perjudiciales para la convivencia en el Centro**. Por tanto, y de acuerdo con la NCOF, si se sospecha que se ha producido una copia de otro compañero o de alguna otra manera, se citará al alumno para realizar una defensa oral de su prueba delante de varios profesores del departamento de Informática que valorarán la autoría de la prueba, quedando el alumno obligado a su realización. En el supuesto de no acudir a la cita, la calificación de la prueba será de 0 puntos. Del mismo modo, si se prueba que el alumno ha realizado una copia total o parcial del código de forma ilícita la calificación de la prueba también será de 0.

Resultado de aprendizaje	Calificación
1. Reconoce la estructura de un programa informático,	-App funciona sin errores semánticos ni sintácticos y

Alumno/a:

<p>identificando y relacionando los elementos propios del lenguaje de programación utilizado</p> <p>2. Escribe y prueba programas sencillos, reconociendo y aplicando los fundamentos de la programación orientada a objetos</p> <p>3. Escribe y depura código, analizando y utilizando las estructuras de control del lenguaje</p> <p><b>20 puntos</b></p>	<p>hace lo que se pide 20 puntos</p> <p>-App funciona parcialmente o con errores sintácticos y/o semánticos o bien, no hace lo que se pide en el enunciado 0 puntos</p>
<p>4. Desarrolla programas organizados en clases analizando y aplicando los principios de la programación orientada a objetos</p> <p><b>5 puntos</b></p>	<p>- Implementación de la herencia correctamente: 1,5 puntos</p> <p>-Creación de clases con todos los atributos y métodos:</p> <ol style="list-style-type: none"> <li>1. Persona abstracta 0,5 puntos</li> <li>2. Empleado 0,5 puntos</li> <li>3. Cliente 0,5 puntos</li> <li>4. ProductoPedido 0,5 puntos</li> <li>5. Pedido 0,5 puntos</li> <li>6. Factura 0,5 puntos</li> </ol> <p>-Implementación de la interfaz por Factura 0,5 puntos</p>
<p>5. Realiza operaciones de entrada y salida de información, utilizando procedimientos específicos del lenguaje y librerías de clases.</p> <p><b>5 puntos</b></p>	<p>- FacturaDNI.txt se ha generado correctamente y contiene toda la información 5 puntos</p> <p>-FacturaDNI.txt no se ha generado correctamente o contiene toda la información de forma parcial 0 puntos</p>
<p>6. Escribe programas que manipulen información seleccionando y utilizando tipos avanzados de datos</p> <p><b>10 puntos</b></p>	<p>- Si se cargan y se muestran los productos pedidos correctamente en el AL de productos de la factura 5 puntos.</p> <p>-Si se cargan los pedidos de cada Empleado (asignarPedidosTurno())5 puntos</p> <p>- Si la carga no es correcta 0 puntos</p>
<p>7. Desarrolla programas aplicando características avanzadas de los lenguajes orientados a objetos y del entorno de programación.</p> <p><b>20 puntos</b></p>	<p>Se han sobrescrito y programado correctamente los métodos de la clase Factura</p> <ul style="list-style-type: none"> <li>• totalsinIVA 15 puntos</li> <li>• ivaReducido 5 puntos</li> </ul> <p>Si no se han sobrescrito y programado los métodos o se ha realizado de forma parcial 0 puntos</p>
<p>9. Gestiona información almacenada en bases de datos relacionales manteniendo la integridad y consistencia de los datos.</p> <p><b>20 puntos</b></p>	<p>- Modifica BD cargarParametros de forma correcta 2 puntos. Si no se han cargado los parámetros a través del método 0 puntos.</p> <p>- Inserta facturas correctamente en la BD 10 puntos. Si no se insertan 0 puntos</p> <p>- Programa cargarEmpleados correctamente 8 puntos. Si no se realiza la carga 0 puntos</p>