Examen de Programación 1. Miércoles 01/FEB/2012

- Disponer un documento de identificación con fotografía sobre la mesa.
- Comenzar a resolver cada parte del examen (cuestiones y problemas) en una hoja de papel diferente. Escribir en cada hoja de papel nombre y apellidos.
- Tiempo para realizar el examen: 3 horas

Fichero de texto de productos y ficheros binarios de pedidos

Vamos a trabajar en el diseño del sistema de información y gestión de un almacén de productos hortofrutícolas. El catálogo de productos que la empresa comercializa se encuentra almacenado en un **fichero de texto de productos** similar al mostrado a continuación. Cada línea describe un producto diferente. La **descripción** de un producto comienza con su denominación, una o más palabras separadas, en su caso, por uno o más espacios en blanco. Le sigue un **código numérico entero** que identifica cada producto. Finaliza con el **precio unitario** del producto, una cantidad numérica expresada en euros y, en su caso, céntimos. Para separar euros y centieuros se utiliza un punto (".").

```
bananas americanas 987 3.2
judías verdes 1102 4.67
guisantes 1055 3
patatas 1027 1.65
plátanos de canarias 1031 2.06
pimientos verdes 1045 1.47
pimientos rojos 1046 1.5
   ...
zanahorias 1075 2.00
```

La sintaxis de la disposición de los datos en el fichero queda formalizada mediante las siguiente reglas sintácticas.

```
<fichero_de_productos> ::=  <producto> { <producto> }
<producto> ::= <denominación>  <código>  <precio_unitario>
<denominación> ::= literal_string  { literal_string }
<código> ::= literal_entero
<precio_unitario> ::= literal_real
```

Los datos del fichero de productos no están necesariamente ordenados respecto de ningún criterio.

Cada uno de los pedidos que recibe el almacén se gestiona mediante un **fichero binario de pedidos** cuya estructura se muestra a continuación:

```
<fichero_de_pedidos> ::=  <pedido>  { <pedido> }
<pedido> ::=  <código>  <cantidad>
<código> ::=  int
<cantidad> ::=  double
```

Un fichero binario de pedidos almacena una secuencia de pares de datos que definen un pedido: el código de un producto (dato binario de tipo ***int***) y la cantidad de producto disponible (dato binario de tipo ***double***).

Los datos del fichero binario de pedidos no están necesariamente ordenados respecto de ningún criterio.

Cuestiones (1.5 puntos)

El fragmento de código que se muestra a continuación contiene los símbolos **public**, **static**, **void**, **main**, **String**, **args**, **int**, **cuenta**, **Scanner** y **entradaDatos**. Explicar la naturaleza de cada uno de dichos símbolos (es decir, qué son o qué representan) y su significado o función en el programa. Se valorará la claridad y precisión de las explicaciones, así como la calidad y legibilidad de la redacción.

```
public static void main (String [] args) {
    int cuenta;
    Scanner entradaDatos;
      ...
}
```

Problema 1º (1.5 puntos)

Diseñar una clase de nombre *Producto*, ubicada en el *package examenFebrero*, que gestione la siguiente información asociada a un producto hortofrutícola: **denominación** del producto, **código numérico** del producto y **precio unitario** del producto.

Los objetos de la clase *Producto* no han de tener atributos públicos. Para trabajar con ellos la clase ofrecerá los métodos públicos que se detallan en el siguiente diagrama UML:

examenFebrero.Producto
+ Producto (denominacion: String; codigo: int; precio: double) + producto(): String + codigo(): int + precio(): double

El constructor *Producto* permite definir los valores de los tres datos asociados a un objeto *Producto* (denominación, código numerico y precio unitario, respectivamente). Los métodos *producto()*, *codigo()*, y *precio()* devuelven el valor de cada uno de los datos asociados al objeto *Producto* (denominación, código numerico y precio unitario, respectivamente).

Se valorará principalmente la especificación de los métodos de la clase y, en segundo lugar, el código (su diseño y legibilidad).

Problema 2º (3.5 puntos)

Diseñar el método ***leerAlmacen*** de la clase ***examenFebrero.Almacen*** que se especifica a continuación.

```
/**
 * Pre: [nombre] es una referencia a un String que define el nombre de un fichero de texto
 *      de productos hortofrutícolas
 * Post: Crea una tabla de referencias a objetos [Producto]. Cada objeto [Producto] gestiona
 *       la información de uno de los productos descritos en el fichero denominado [nombre].
 *       Devuelve la referencia a la tabla creada, salvo en el caso de que se produzca
 *       alguna excepciónen el proceso de lectura del fichero, en cuyo caso devuelve [null]
 */
public static Producto[] leerAlmacen (String nombre)
```

Problema 3º (1.5 puntos)

Diseñar el método ***precio*** de la clase ***examenFebrero.Almacen*** que se especifica a continuación.

```
/**
 * Pre: ---
 * Post: Si hay un producto en la tabla [almacen] cuyo código sea igual al valor del parámetro
 *       código, entonces devuelve el precio unitario de dicho producto, en caso contrario
 *       devuelve un precio negativo
 */
public static double precio (Producto[] almacen, int codigo)
```

Observación importante: no se admitirá una solución no estructurada. Ello supone que cualquier bucle debe concluir exclusivamente al dejar de satisfacerse su condición de iteración.

Problema 4º (2.0 puntos)

Diseñar el método ***valorar*** de la clase ***examenFebrero.Almacen*** que se especifica a continuación.

```
/**
 * Pre: [nombre] define el nombre de un fichero binario de pedidos. Para cada uno de los
 *      productos cuyo código consta en el fichero anterior hay un único elemento en la
 *      tabla [almacen] que gestiona la información de dicho producto. Ningún elemento
 *      de la tabla [almacen] tiene valor [null]
 * Post: Devuelve el valor total de todas las mercancías descritas en el fichero binario
 *       de pedidos denominado [nombre], excepción hecha del caso de fallo en la lectura
 *       del fichero, en cuyo caso devuelve un valor negativo
 */
public static double valorar (Producto[] almacen, String nombre)
```