Examen de prácticas de Programación 1

Escuela de Ingeniería y Arquitectura
Departamento de Informática e Ingeniería de Sistemas

3 de septiembre de 2014

- Tiempo para realizar el trabajo de programación propuesto: **90 minutos**.
- Entrega del trabajo a través de la plataforma Moodle, a la que deberá subirse el fichero correspondiente al código fuente Java de la clase que se pide desarrollar (fichero de extensión «`.java`», no «`.class`»).

Especificación del trabajo que debe realizarse

En esta prueba se va a trabajar con los mismos tipos de ficheros que se utilizaron en el trabajo de la asignatura: un fichero de clientes y un fichero de ventas.

El fichero de clientes es de texto, se denomina «`clientes.txt`» y se ubica en una carpeta cuya ruta de acceso desde el programa es «`datos/trabajo`». Su estructura es idéntica a la de los ficheros de ciudadanos con los que se ha trabajado en el capítulo 6 de la colección de problemas y en el trabajo de la asignatura. Para leer la información del fichero de clientes se deberá aprovechar código ya desarrollado y trabajar exclusivamente con los recursos que facilitan las clases `problemas.cap6.Ciudadano` y `problemas.cap6.FicherosDeCiudadanos` desarrolladas en las prácticas de la asignatura.

El fichero de ventas es un fichero binario denominado «`ventas.dat`» y también está ubicado en la carpeta «`datos/trabajo`». Su estructura es la misma que la descrita en el capítulo 7 de la colección de problemas y en el trabajo de la asignatura, y se muestra a continuación utilizando la notación BNF:

```
<fichero_ventas> ::= { <venta> }
<venta> ::= <dni_cliente> <código_producto> <cantidad> <fecha>
<dni_cliente> ::= int
<código_producto> ::= int
<cantidad> ::= double
<fecha> ::= int
```

Se pide desarrollar una única clase ubicada en el paquete `examen.septiembre` cuyo nombre coincida con sus apellidos (ej.: **`class`** `LopezGutierrez`). Esta clase debe incluir un método `main` que, al ser ejecutado, presente en la pantalla un listado con los códigos de los productos que aparecen en el fichero de ventas y el nombre de un cliente que los haya comprado. Dicho nombre se obtendrá del fichero de clientes, ya que el DNI de cliente que aparecen en cada venta del fichero de ventas coinciden con el DNI de exactamente un cliente del fichero de clientes.

En el listado, se mostrará cada producto vendido como máximo una vez. Si el producto se vendió a más de un cliente, es indiferente el nombre que se muestre, siempre y cuando sea el de un cliente que haya comprado dicho producto. Es indiferente el orden en el que se muestren los productos. Se sabe que los códigos de los productos están comprendidos entre 0 y 200.

El programa también informará acerca del código del producto más comprado por hombres y el más comprado por mujeres.

Se muestra a continuación un ejemplo de ejecución de dicho programa.

```
Producto 101 comprado al menos por ESTHER MULA ZAMORA
Producto 102 comprado al menos por DIEGO TOMAS TEJEDOR
Producto 103 comprado al menos por LAURA BERMUDEZ ESCORIAL
...
Producto 123 comprado al menos por TIBURCIO ZURDO ACIN
Producto 124 comprado al menos por ARANZAZU PANADERO ESCOLANO
Producto 125 comprado al menos por MARIA ISABEL BADENES SIPAN

El producto más comprado por hombres es el Producto 124
El producto más comprado por mujeres es el Producto 103
```

Para la realización de pruebas se facilitan un fichero de clientes y otro de ventas a los que se puede acceder a través de la web de la asignatura, en la página «Materiales docentes comunes», accediendo al enlace situado en el apartado «Código Java descargable» y, posteriormente, a la carpeta «`datos/trabajo`».

En esta prueba se valorarán los siguientes aspectos y con este orden de importancia:

1. Comportamiento del programa según las especificaciones de este enunciado.
2. Legibilidad del código.
3. Diseño algorítmico del código.

Un programa que tenga errores de compilación o que, al ser ejecutado, no proporcione ningún resultado, será calificado con un cero.

Solución

```
package examen.septiembre;

import java.io.DataInputStream;
import java.io.EOFException;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import problemas.cap6.Ciudadano;
import problemas.cap6.FicherosDeCiudadanos;

/**
 * Esta clase tiene un método main que, al ser ejecutado, presenta en la
 * pantalla un listado con los códigos de los productos que aparecen en el
 * fichero de ventas «FICHERO_VENTAS» y el nombre de un cliente que los haya
 * comprado, según el contenido del fichero «FICHERO_CLIENTES». También
 * informa acerca del código del producto más comprado por hombres y el más
 * comprado por mujeres.
 *
 * Ejemplo de ejecución de dicho programa:
 *
 * Producto 101 comprado al menos por ESTHER MULA ZAMORA
 * Producto 102 comprado al menos por DIEGO TOMAS TEJEDOR
 * Producto 103 comprado al menos por LAURA BERMUDEZ ESCORIAL
 * ...
 * Producto 123 comprado al menos por TIBURCIO ZURDO ACIN
 * Producto 124 comprado al menos por ARANZAZU PANADERO ESCOLANO
 * Producto 125 comprado al menos por MARIA ISABEL BADENES SIPAN
 *
 * El producto más comprado por hombres es el Producto 124
 * El producto más comprado por mujeres es el Producto 103
 */
public class Laboratorio {

    /**
     * Número máximo de productos distintos.
     */
    private static final int MAX_PRODUCTOS_DISTINTOS = 201;
```

```
/**
 * Ruta y nombre del fichero de ventas.
 */
private static final File FICHERO_VENTAS = new File(
        "datos/trabajo/ventas.dat");

/**
 * Ruta y nombre del fichero de clientes.
 */
private static final File FICHERO_CLIENTES = new File(
        "datos/trabajo/clientes.txt");

/**
 * Pre:  «FICHERO_VENTAS» y «FICHERO_CLIENTES» existen, pueden ser
 *       abiertos para su lectura y tienen las estructuras planteadas en
 *       el enunciado.
 * Post: Ha presentado en la pantalla un listado con los códigos de los
 *       productos que aparecen en el fichero de ventas «FICHERO_VENTAS»
 *       y el nombre de un cliente que los haya comprado, según el
 *       contenido del fichero «FICHERO_CLIENTES». También ha informado
 *       acerca del código del producto más comprado por hombres y el
 *       más comprado por mujeres. El formato del listado se corresponde
 *       con el del ejemplo de la cabecera de esta clase.
 */
public static void main(String[] args) {
    // Lectura del fichero de clientes a través de la clase
    // «FicherosDeCiudadanos»
    Ciudadano[] clientes =
            FicherosDeCiudadanos.leerFichero(FICHERO_CLIENTES);

    if (clientes != null) {
        try {
            // Apertura del fichero de ventas
            DataInputStream f = new DataInputStream(
                    new FileInputStream(FICHERO_VENTAS));

            // Definición de estructuras de datos para almacenar la
            // información necesaria para resolver el problema: nombres
            // de los compradores indexados por código de productos y
            // número de veces que se compran por hombres y mujeres,
            // también indexados por código de producto.
            String[] compradores = new String[MAX_PRODUCTOS_DISTINTOS];
            int[] vecesHombre = new int[MAX_PRODUCTOS_DISTINTOS];
```

```
            int[] vecesMujer = new int[MAX_PRODUCTOS_DISTINTOS];

            // Lectura y procesamiento del fichero de ventas
            try {
                while (true) {
                    // Lectura de una venta
                    int dniCliente = f.readInt();
                    int codProducto = f.readInt();
                    f.readDouble(); // cantidad
                    f.readInt(); // fecha

                    // Actualización de las estructuras de datos
                    procesarVenta(dniCliente, codProducto, clientes,
                            compradores, vecesHombre, vecesMujer);
                }
            }
            catch (EOFException ex) {
            }

            f.close();

            // Escritura en la pantalla de los resultados
            escribirCompradoresProductos(compradores);
            System.out.println();
            escribirMasVendidoPorSexo(vecesHombre, vecesMujer);
        }
        catch (IOException ex) {
            System.out.println("Error de E/S: " + ex.getMessage());
        }
    }
}
```

```java
/**
 * Pre:  0 <= «codProducto» < MAX_PRODUCTOS_DISTINTOS,
 *       «compradores».length = «vecesHombre».length =
 *       = «vecesMujer».length = MAX_PRODUCTOS_DISTINTOS,
 *       existe en «clientes» un ciudadano cuyo DNI es «dniCliente».
 * Post: Si no había ningún nombre de un comprador en la tabla
 *       «compradores» asociado a «codProducto», se le asocia el nombre
 *       del ciudadano con DNI «dniCliente» según el contenido de la
 *       tabla «clientes». También según el contenido de la tabla
 *       «clientes», si el ciudadano con DNI «dniCliente» es un hombre,
 *       se ha incrementado en una unidad el valor de la componente
 *       correspondiente a «codProducto» de la tabla «vecesHombre».
 *       Si es mujer, se ha hecho lo propio con la tabla «vecesMujer».
 */
private static void procesarVenta(int dniCliente, int codProducto,
        Ciudadano[] clientes, String[] compradores, int[] vecesHombre,
        int[] vecesMujer) {

    // Búsqueda del cliente
    Ciudadano cliente = buscar(dniCliente, clientes);

    // Actualización de «compradores», «vecesHombre» y «vecesMujer»
    if (compradores[codProducto] == null) {
        compradores[codProducto] =
                cliente.nombre() + "␣" + cliente.apellidos();
    }

    if (cliente.esHombre()) {
        vecesHombre[codProducto]++;
    }
    else {
        vecesMujer[codProducto]++;
    }
}
```

```java
/**
 * Pre:  Existe en «clientes» un ciudadano cuyo DNI es igual a
 *       «dniCliente».
 * Post: Ha devuelto el objeto de la clase Ciudadano de la tabla
 *       «clientes» cuyo DNI es igual a «dniCliente».
 */
private static Ciudadano buscar(int dniCliente, Ciudadano[] clientes) {
    // Esquema de búsqueda con garantía de éxito
    int i = 0;
    while (clientes[i].dni() != dniCliente) {
        i++;
    }
    // clientes[i].dni() == dniCliente
    return clientes[i];
}

/**
 * Pre:  «compradores» != null.
 * Post: Ha presentado en la pantalla un listado con los códigos de los
 *       productos que según la tabla «compradores» han sido comprados,
 *       junto con el nombre de un cliente que los ha comprado. El
 *       formato del listado se corresponde con el del ejemplo de la
 *       cabecera de esta clase.
 */
private static void escribirCompradoresProductos(String[] compradores) {
    for (int i = 0; i < compradores.length; i++) {
        if (compradores[i] != null) {
            System.out.printf("Producto␣%d␣comprado␣al␣menos␣por␣%s%n",
                              i, compradores[i]);
        }
    }
}
```

```
    /**
     * Pre:  «vecesHombre».length = «vecesMujer».length.
     * Post: Ha escrito en la pantalla el código del producto más comprado
     *       por hombres y el código del producto más comprado por mujeres,
     *       según la información de las tablas «vecesHombre» y
     *       «vecesMujer». El formato se corresponde con el del ejemplo de
     *       la cabecera de esta clase.
     */
    private static void escribirMasVendidoPorSexo(int[] vecesHombre,
            int[] vecesMujer) {
        // Índice y código del producto más comprado por hombres
        int iMaxHombre = 0;
        // Índice y código del producto más comprado por mujeres
        int iMaxMujer = 0;

        for (int i = 1; i < vecesMujer.length; i++) {
            if (vecesHombre[i] > vecesHombre[iMaxHombre]) {
                iMaxHombre = i;
            }
            if (vecesMujer[i] > vecesMujer[iMaxMujer]) {
                iMaxMujer = i;
            }
        }

        System.out.println("El␣producto␣más␣comprado␣por␣hombres␣es␣el␣"
                           + "Producto␣" + iMaxHombre);
        System.out.println("El␣producto␣más␣comprado␣por␣mujeres␣es␣el␣"
                           + "Producto␣" + iMaxMujer);
    }
}
```