ESTRUCTURA DE DATOS Y ALGORITMOS **8-Febrero-2007**
Titulación: Ingeniero Técnico en Informática de Gestión **Curso: 2º**

1. Ejercicio (1 puntos)

Dado el array A={8, 3, 7, 1, 4, 9, 5, 2, 6}

Describir cual es el estado del array A después de cada paso principal del algoritmo: InsertionSort.

2. Ejercicio. (3 puntos)

Dada la siguiente clase que define la información de un single musical:

```
class singleMusical {
      private String artista, titulo;
      private date fecha;
      ...
}
```

Hemos implementado una clase Discográfica para guardar la información de varios singles musicales. Para que las búsquedas sean eficientes, la información se mantiene ordenada de dos formas:

1) por título y
2) por artista (si hay varios singles del mismo artista, se ordena por titulo).

En la discográfica no existen singles con el mismo titulo. Hay que tener en cuenta que la discográfica **se actualiza frecuentemente**, bien porque llegan nuevos singles musicales o bien porque se han vendido los existentes. Dada la siguiente especificación parcial de dicha clase:

```
class Discografica {
      private … // parte de la especificación de la clase
      public void añadir(SingleMusical single);
      ...
}
```

Se pide:

a) Completar la especificación de la parte privada de la clase *Discografica*. Especificar cualquier clase auxiliar que se utilice. Argumenta de forma breve tu decisión

b) Teniendo en cuenta la especificación del apartado a), representa gráficamente la siguiente información *(artista, titulo, fecha):*

```
Ismael Serrano, Ultimamente, 1998
Ana Pozas, Seguir Viviendo, 2001
Javier Alvarez, Sunset Boulevard, 1996
Pedro Guerra, Pasa, 1997
```

c) **Implementar** la operación *añadir* para que dado un single, lo añada en la galería y actualice la estructura de modo adecuado. Especificar cualquier método auxiliar que se utilice.

3. Ejercicio (3 puntos)

Dada la siguiente especificación de las clases BTNode, BinTree y BinTreeItr:

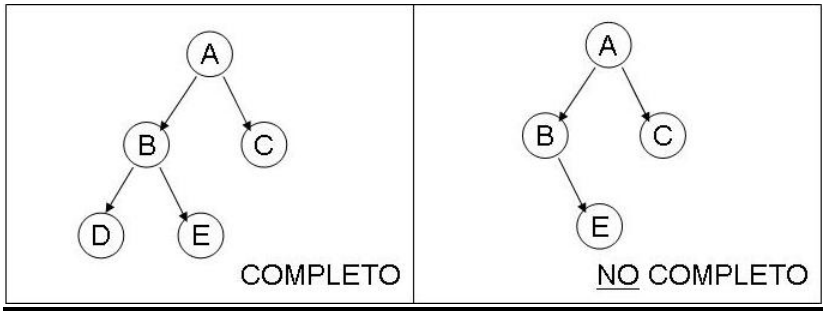
```
public class BinTreeItr{
     BTNode current;
     BinTree bTree;
}

public class BinTree{
     BTNode root;
}
```

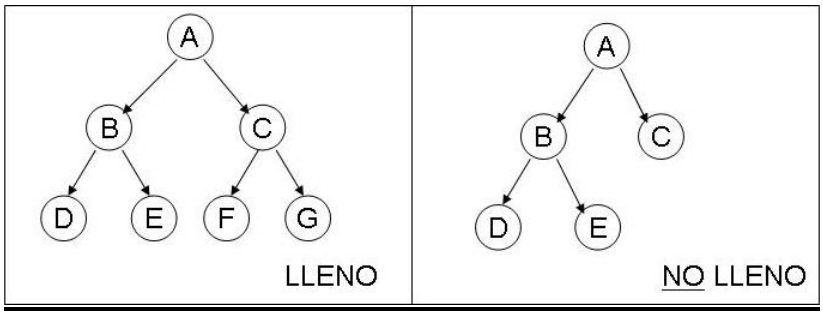
```
public class BTNode{
     Object content;
     BTNode left;
     BTNode right;
}
```

Nota: Todo método necesario tendrá que ser implementado.

Un árbol **está completo** si todos los nodos, excepto las hojas, tienen 2 hijos.



Un árbol **está lleno** si, además de ser completo, todas las hojas están en el mismo nivel.



Especificar, diseñar e implementar el siguiente método de un árbol binario de elementos:

boolean ***estaLleno()***

que calcula si el árbol **está lleno** (devuelve true) o no (devuelve false).

4. Ejercicio (3 puntos)

Tenemos como datos el fichero **grandesDesastres.txt**, donde cada línea contiene la siguiente información sobre grandes desastres:

- Fecha del desastre,
- Número de víctimas.

Los datos del fichero están ordenados cronológicamente ***del más antiguo al más reciente***.

Diseñar e implementar un subprograma que calcule el histórico de los desastres que hayan sobrepasado el número máximo de victimas ocurrido en un desastre anterior, ordenado cronológicamente ***del más reciente al más antiguo*** (escribir por pantalla la fecha y el nº de victimas de los desastres).
Utilizar **un T.A.D.** visto en clase como estructura auxiliar. El programa debe de ser lo más eficiente posible, que se intente aproximar a un O(n).

Ejemplo:

Para el siguiente fichero:

14/06/1992	12
22/10/1995	6
26/09/1998	11
30/12/1999	4
05/11/2001	13
17/06/2003	5
08/02/2004	16
01/03/2006	10

Obtendríamos por pantalla:

08/02/2004	16
05/11/2001	13
14/06/1992	12

Para la realización de este ejercicio se dispone de las siguientes clases:

```
class Token {
        //Inicializa el fichero
        Token(String file);

        //cierto si existen más líneas en el fichero, falso en caso contrario
        bolean hasMoreTokens();

        //devuelve los datos de una línea del fichero en un objeto de tipo Datos
        Datos getToken();
}
```

```
class Datos{
        //devuelve la fecha del desastre
        public int getFecha();
        //devuelve el nº de victimas del
        desastre
        public int getNumVictimas();
}
```

```
class Desastre{
        //Constructor
        Trabajo(int pFechaD, int pVictimasD);
        //devuelve la fecha del desastre
        public int getFecha();
        //devuelve el nº de victimas del desastre
        public int getNumVictimas();
}
```