ESTRUCTURA DE DATOS Y ALGORITMOS **22-Junio-2011**
Titulación: Ingeniero Técnico en Informática de Gestión **Curso: 2º**
Nombre y apellidos: _____ Nota:

- Escribe tu **nombre y apellidos** en esta hoja e inmediatamente en todas las suplementarias, incluso las de sucio. El no hacerlo puede suponer tu expulsión
- Puedes utilizar el **lápiz** para tus respuestas. No está permitido el uso de apuntes, notas o libros. No puedes tener un **móvil** encendido, ni utilizar cualquier otro **aparato electrónico**.
- **Todos los alumnos implicados en una copia** de un ejercicio **tendrán una nota final de 0**. El alumno es responsable de velar por su examen. Es decir **tanto el que copia como el que se deja copiar (ya sea de manera activa o pasiva) recibirán el mismo castigo sin que exista atenuante alguno**

1. Ejercicio (2 puntos)

a) ¿Cuál es la estructura de datos que utiliza un intérprete de un lenguaje de programación para llevar a cabo la recursión? ¿Por qué?

b) Suponiendo que un árbol binario representa una expresión aritmética, **describe** que datos se almacenan en los nodos internos y externos. ¿Qué recorrido utilizarías para evaluar la expresión? ¿Por qué?

c) ¿Cuál es la ventaja principal de la estructura de datos TablaHash respecto a un array, Vector o Lista enlazada? Y la desventaja principal?

d) Una lista enlazada ¿Es una estructura de datos lineal o no-lineal?

e) Problema práctico: La gerencia de un taller de autos quiere establecer un orden para el arreglo de los coches según la categoría de fidelización del cliente y el orden de llegada del coche al taller. Para ello, categorizan a sus clientes de 1 a 5 según el gasto realizado (siendo el 1 el de mayor gasto, el más fiel). ¿Qué estructura de datos utilizarías para organizar los datos y poder elegir de forma eficiente el próximo coche para ser arreglado? ¿Por qué?

f) Vamos a suponer que en un mismo programa se han definido dos instancias de la clase iteradora sobre una misma lista enlazada simple. ¿Se pueden producir problemas? En caso afirmativo, explica como o cuando puede ocurrir. En caso negativo, explica cual es la característica principal para que no ocurra ningún problema.

ESTRUCTURA DE DATOS Y ALGORITMOS **22-Junio-2011**
Titulación: Ingeniero Técnico en Informática de Gestión **Curso: 2º**
Nombre y apellidos: _____

- Escribe tu **nombre y apellidos** en esta hoja e inmediatamente en todas las suplementarias, incluso las de sucio. El no hacerlo puede suponer tu expulsión
- Puedes utilizar el **lápiz** para tus respuestas. No está permitido el uso de apuntes, notas o libros. No puedes tener un **móvil** encendido, ni utilizar cualquier otro **aparato electrónico**.
- **Todos los alumnos implicados en una copia** de un ejercicio **tendrán una nota final de 0**. El alumno es responsable de velar por su examen. Es decir **tanto el que copia como el que se deja copiar (ya sea de manera activa o pasiva) recibirán el mismo castigo sin que exista atenuante alguno**

2. *Ejercicio (2 puntos)*

Tenemos un documento html conteniendo un gran número de etiquetas de apertura y de cierre que pueden ser diferentes. Cada elemento etiquetado siempre tiene la estructura <ETIQUETA> *contenido* </ETIQUETA>, donde *contenido* puede ser a su vez una serie de elementos etiquetados o simplemente texto. Cada etiqueta se separa del texto anterior y siguiente por uno o más blancos o por un cambio de línea. Consideraremos sólo aquellos documentos html donde toda etiqueta de apertura (<ETIQUETA>) tiene su correspondiente etiqueta de cierre (</ETIQUETA>). En un documento html bien formado siempre se cierra primero la última etiqueta abierta antes, por ejemplo:

```
<HTML>
  <HEAD>
     <TITLE> Documento de prueba </TITLE>
  </HEAD>
  <BODY>
     <H1> Esto es una "demo" de documento HTML </H1>
        <i> Esto es el <b> m´as sencillo </b> de los documentos <b> HTML </b> </i> .
  </BODY>
</HTML>
```

Implementar una función que procese el fichero para comprobar si las distintas etiquetas están abiertas y cerradas en el orden adecuado y que no hay ninguna etiqueta de apertura sin su correspondiente de cierre y viceversa. La cabecera de la función será la siguiente:

```
public boolean etiquetasEquilibradas(String fichero);
```

Para la lectura del fichero puedes utilizar la clase Token:

```
public class Token{
        /*
         * Abre el fichero de texto indicado por el parámetro 'file'
         */
        public Token(String file);
        /*
         * Devuelve True si no se ha llegado al final de fichero y False, en caso contrario
         */
        public boolean hasMoreTokens();
         /*
          * Devuelve una palabra o etiqueta, filtrando todos los espacios y saltos de línea
          */
        public String getToken();
}
```

3. *Ejercicio (3 puntos)*

Supongamos que queremos simular el funcionamiento de una aduana de entrada en un país con 12 puestos de atención a los pasajeros. Por cada pasajero guardaremos su dni, nombre y nacionalidad. Delante de cada puesto de atención hay una fila de pasajeros que esperan a ser atendidos por orden de llegada.

La simulación consiste en una serie de iteraciones que simulan la evolución del estado de las distintas filas. En cada iteración la siguiente función, aplicada a cada fila, nos dirá si se atiende o no, un nuevo pasajero de esa fila:

```
public boolean atiendePasajero(int it, int f);
```

siendo `it` la iteración y `f` la fila. En caso de que se atienda, habrá que eliminarlo de la fila correspondiente.

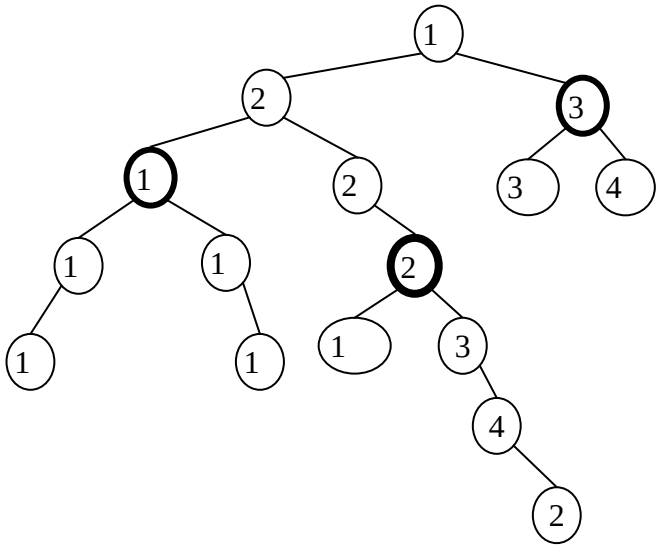
Al final de cada iteración trataremos de equilibrar la longitud de las distintas filas. Para ello, por cada fila i, se van pasando los últimos pasajeros al final de las filas que tengan menos pasajeros que ella, que pueden ser varias. Sin embargo, sólo será posible pasar un pasajero como mucho a otra fila.

Se pide:

a) (0.5 ptos.) Definir los tipos de datos más adecuados para guardar toda la información de la aduana.
b) (1 pto.) Implementar un algoritmo que realice el proceso de equilibrado de las distintas filas tal y como se ha descrito.
c) (1.5 ptos.) Implementar un algoritmo que realice la simulación hasta que se vacíen todas las filas y devuelva el número de pasajeros atendidos en cada puesto.

4. *Ejercicio (3 puntos)*

Se pide **especificar**, **diseñar** e **implementar** un método que dado un árbol binario de elementos de tipo entero, sume el contenido de los nodos cuya altura coincide con su profundidad.



Dado el árbol de al lado el resultado será **6**, que es la suma de los nodos que coinciden su altura con la profundidad (denotados en negrita).

Para la representación del árbol se provee de las siguientes clases:

```
public class BinTree{
  protected BTNode root;
  public BTNode getRoot();
}
```

```
public class BinTreeItr{
    protected BinTree bTree;
}
```

```
public class BTNode{
  protected Object content;
  protected BTNode left;
  protected BTNode right;
  // y los correspondientes get
}
```

Nota: Cualquier otro método que actúe sobre estas clases tendrá que ser implementado.