ESTRUCTURA DE DATOS Y ALGORITMOS **4-Febrero-2010**
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 (3 puntos)

a) ¿Qué estructura de datos utilizarías para transformar la siguiente expresión a su equivalente expresión postfija? Argumenta tu decisión.

Ej. entrada: `(x-y)/(z+w)-(z+y)^x` Salida: `xy-zw+/zy+x^-`

b) La solución de un problema concreto requiere la ejecución de las dos siguientes operaciones una y otra vez sobre un conjunto de datos: 1) buscar un dato por contenido y 2) insertar más datos después del dato encontrado en 1). ¿Qué estructura de datos sería la más adecuada? Argumenta tu decisión.

c) Cual es la complejidad del algoritmo de búsqueda de un elemento por posición en una lista? Es siempre igual? Argumenta tu respuesta.

d) En las TablasHash abiertas o hashing enlazado, la operación de insertar un elemento se ha realizado como primer elemento de la lista. Si la inserción en la lista enlazada se hace de tal forma que esté ordenada respecto a la clave. ¿Qué ventajas y desventajas tiene respecto a la eficiencia de las operaciones *insertar*, *buscar* y *eliminar*?

e) Muestra que es posible soportar simultáneamente las operaciones siguientes en tiempo constante: `apilar`, `desapilar`, `cima` y `buscarMin`. Observa que `eliminarMin` no forma parte del repertorio. ***Pista***: mantener dos pilas.

1. Explica que es lo que almacenarías en cada una de las pilas y la estrategia que sigues o el uso que haces de las dos pilas.

2. representa el estado de las dos pilas, después de ejecutar aquellas instrucciones, que cambian el estado de las dos pilas (p.ej. `apilar`, `desapilar` y `eliminarMin`), del siguiente programa:

programa	1) PilaCP p = new PilaCP();	2) p. apilar(new Integer(6));
estado		
programa	3) p. apilar(new Integer(4));	4) p. apilar(new Integer(7));
estado		
programa	5) p. apilar(new Integer(5));	6) p. apilar(new Integer(3));
estado		
programa	7) p. cima();	8) p.buscarMin();
devuelve		
programa	9) p.desapilar();	**devuelve**
estado		
programa	10) p. cima();	11) p.buscarMin();
devuelve		
programa	12) p.eliminarMin();	**devuelve**
estado		
programa	13) p. cima();	14) p.buscarMin();
devuelve		
programa	15) p.eliminarMin();	**devuelve**
estado		
programa	16) p. cima();	17) p.buscarMin();
devuelve		
programa	18) p.eliminarMin();	**devuelve**
estado		

ESTRUCTURA DE DATOS Y ALGORITMOS **4-Febrero-2010**
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)*

Se quiere desarrollar un programa para la consulta de teléfonos. Principalmente y con mucha frecuencia se ejecutan las consultas por nombre y por número de teléfono. Los datos se encuentran en un fichero de texto dónde en cada línea se describen el nombre y su número de teléfono. Se supone que no hay datos repetidos.

Se dispone de las siguientes clases:

```
public class Agenda{
        /**
          * Crea una agenda vacia
          */
        public Agenda();
        /**
          *  Carga los datos de los contactos desde el fichero de texto
          */
        public void cargarDatos(String fichero);
        /**
          *  Añade un nuevo contacto en la agenda
          */
        public void anadirContacto(Contacto contacto);
        /**
          * Devuelve el teléfono del contacto con el mismo nombre que el parámetro nombre. Si no
          * existe, entonces devuelve null.
          */
        public String buscarTelefono(String nombre);
        /**
          * Devuelve el nombre del contacto que tiene mismo teléfono que el parámetro telefono. Si no
          * existe, entonces devuelve null.
          */
        public String buscarNombre(String telefono);
}
```

```
public class Contacto {
  private String nombre;
  private String telefono;
  public String getNombre()
  public String getTelefono()
}
```

agenda.txt

Miguel	943010003
Antton	945010001
Valeriana	943010001
Isabel	946010001
Gepeto	945010002

```
public class Token{
  /**
    * Abre el fichero de texto indicado por '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 los datos de un contacto, correspondiendo a una línea del fichero
    */
  public Contacto getToken();
}
```

Se pide:

a) (0,75 punto) **Diseñar** e **implementar** la estructura (las variables miembro) más adecuada(s) para la clase `Agenda`, cuya misión es gestionar la información de los contactos, teniendo en cuenta las condiciones arriba descritas y las estructuras de datos vistas en clase (menos la Hashtable). Representa gráficamente cómo se quedan almacenados los datos después de haber ejecutado `cargarDatos(`*"agenda.txt"*`)`. **Nota**: En caso de necesitar algo más, implementa las clases y los métodos necesarios. Argumenta tus decisiones.

b) (1 punto) **Implementar** los métodos `buscarTelefono` y `buscarNombre` de la clase `Agenda`. Ten en cuenta que se ejecutan con mucha frecuencia y tu respuesta en el apartado anterior.

c) (0,25 puntos) Calcula cual es la complejidad, en notación *O*, de los métodos implementados en b).

Nota: Se valorarán los aspectos de reutilización..

3. *Ejercicio (1 punto)*

Si el orden en que se almacenan los elementos en una lista no es importante, entonces podemos acelerar las búsquedas usando la siguiente heurística, conocida como *mover-al-frente*: siempre que accedas a un elemento, muévelo al principio de la lista. La razón por la que esto constituye una mejora, es que los elementos a los que se accede frecuentemente tienden a migrar hacia el principio de la lista, mientras que los menos frecuentemente accedidos migran hacia el final de la lista. En consecuencia, los elementos más frecuentemente accedidos requieren una búsqueda breve.

Se pide:

a) **Implementar** las clases necesarias (sin los métodos) para representar una lista enlazada con doble terminación.

b) **Implementar** la heurística *mover-al-frente* para las listas enlazadas con doble terminación.

c) **Describir** gráficamente los pasos principales de la heurística *mover-al-frente*, mediante un ejemplo.

5. ***Ejercicio (3 puntos)***

Se desea realizar un sistema informático para escribir libros de cuentos dinámicos. Un *libro de cuentos dinámicos* tiene un titulo y está compuesto por varios capítulos, los cuales no se leen en el mismo orden de impresión. La característica principal del libro es que el lector es el creador de distintos cuentos (normales), los cuales tendrán varios capítulos del libro en común. La secuencia de los capítulos que forman un cuento (normal) la va estableciendo el lector al final de cada capitulo.

Un capítulo está compuesto por un titulo, seguido de un texto y al final, puede tener una pregunta o no. Todas las preguntas de cualquier capitulo siempre tendrán dos respuestas (sí o no). Además, cada respuesta indicará cual es el siguiente capítulo dónde continúa el cuento (los capítulos deben ser distintos), además de la página. De esta forma, dependiendo de las respuestas que vaya seleccionando el lector se creará un cuento u otro. Aquellos capítulos que no tengan una pregunta denotan el final de los cuentos.

La escritura de estos cuentos no es nada fácil. Como es de pensar, puede que algunas combinaciones de capítulos formen cuentos que no tengan ni pie ni cabeza y también, puede ocurrir que se formen ciclos, lo cual implica que se puede llegar a no terminar nunca un cuento. La prevención de crear este tipo de cuentos es una de las principales tareas del escritor.

La metodología que sigue el escritor para crear un libro de cuentos dinámicos es el siguiente:

1. Debido a la dificultad de escribir este tipo de cuento y para que no haya ciclos en el desarrollo de la secuencia del cuento, el escritor decide que todos los capítulos menos el de inicio solamente pueden tener un único capítulo como su predecesor en la secuencia de un posible cuento.

   Para ello, mientras va escribiendo los capítulos, el escritor prefija una posible secuenciación de capítulos mediante un esquema. Este esquema, llamado *esquema de cuentos dinámicos*, es una estructura arborescente (ver Figura 1) y sus objetivos principales son: *i)* organizar los capítulos que van a componer un libro de cuentos dinámicos en un orden determinado; *ii)* le sirva de ayuda al escritor para que escribir cuentos coherentes; y *iii)* cumpla las condiciones arriba descritas. Este esquema será su guía y lo mantendrá hasta que finalice el libro.

2. La edición de los cuentos dinámicos habrá finalizado, cuando no queden capítulos sin terminar. Un capitulo *está sin terminar*, si tiene pregunta y alguna de sus respuestas no tiene asignado el siguiente capitulo.

Para la representación del esquema de cuentos dinámicos se provee de las siguientes clases:

```
public class EsquemaCuentosDinamicos {
  private String titulo;
  private BinTree capitulos;
  private CapitulosBinTreeItr itrCapitulos;
  ...
}
```

```
public class Capitulo {
  private String titulo;
  private String texto;
  private String pregunta;
  public boolean tienePregunta();
  // y los correspondientes get y set
}
```

```
public class CapitulosBinTreeItr extends
BinTreeItr{
 ...
}
```

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

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

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

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

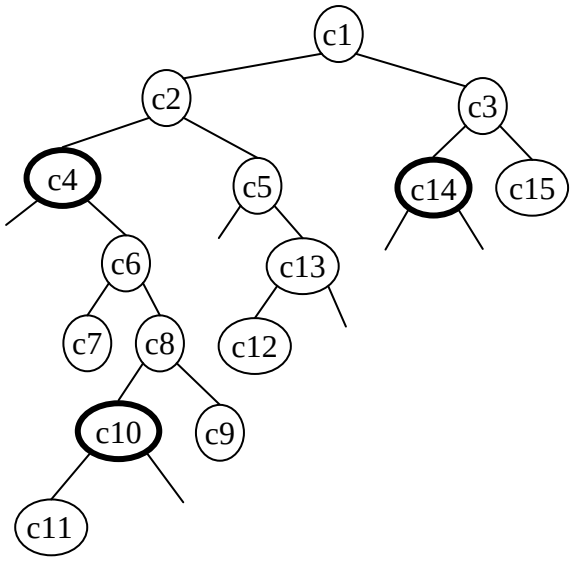
ESTRUCTURA DE DATOS Y ALGORITMOS **4-Febrero-2010**
Titulación: Ingeniero Técnico en Informática de Gestión **Curso: 2º**

Se pide:

Especificar, **diseñar** e **implementar** un método que dado un esquema de cuentos dinámicos (`EsquemaCuentosDinamicos`), representado mediante un árbol binario de capítulos, y un número entero *(i)*, calcule cuantos capítulos están sin terminar, que serían *i*.esimos de un cuento normal y siguientes de los capítulos (*i-1*)esimos, en caso de que su pregunta se responda afirmativamente.

```
public int numeroDeCapitulosEnIafirmativosYsinTerminar(int i)
```

La siguiente figura muestra un posible esquema de cuentos dinámicos:



Por ejemplo, tomando el esquema de al lado y:

- Si **i = 2,** entonces devuelve **0**
- Si **i = 3,** entonces devuelve **2**
- Si **i = 4**, entonces devuelve **0**
- Si **i = 6,** entonces devuelve **1**

Figura 1. Árbol binario que representa un esquema de cuentos dinámicos. El nodo que no tiene ramas por debajo, significa que el capitulo no tiene pregunta, por lo tanto está terminado. Las ramas izquierdas representan la respuesta afirmativa a la pregunta del capitulo y la rama derecha, la negativa.