Examen de Estructuras de Datos y Algoritmos
(Ingeniería Informática)

Febrero 2007

Primera parte (50% nota del examen)

1) Escribir un método en Java que permita fusionar dos listas creando una tercera. La lista resultante debe contener alternativamente un elemento de la primera lista, uno de la segunda, uno de la primera, uno de la segunda, ..., hasta que se acabe una de las dos; luego aparecerán los elementos restantes de la lista más larga. Las listas originales serán objetos de clases que implementen la interfaz `List` de las colecciones Java. La lista resultante será un objeto de la clase `LinkedList`. La cabecera de este método será:

```
public static <E> List<E> fusionar
        (List<E> lista1, List<E> lista2)
```

2) Escribir una clase Java que permita determinar el género masculino o femenino de algunas de las palabras de un texto usando un mapa que verifica la interfaz `Map` de las colecciones Java. El constructor de la clase recibe como parámetro el texto como un objeto de la clase `Texto`, que se pasa como parámetro. La clase `Texto` contiene un método `obtenerPalabra`(), que retorna un `String`, y que permite obtener la siguiente palabra del texto, retornando `null` si el texto se ha terminado. El constructor debe crear el mapa a partir del texto, anotando las palabras como clave y su género como valor del tipo enumerado que se muestra abajo. Para determinar el género, si una palabra está precedida de un artículo masculino ("el", "un", "los", "unos"), se considera masculina; al revés si está precedida de un artículo femenino ("la", "una", "las", "unas"). Para que la comprobación sea eficiente, los artículos masculinos y femeninos se guardarán en sendos conjuntos de la clase `HashSet` que se crearán al efecto. Para mantener el método sencillo no añadir ninguna otra regla que pudiese mejorar la corrección o calidad del resultado. No es necesario tampoco tratar el caso en que una palabra sea unas veces masculina y otras femenina.

```
public enum Genero {masculino, femenino }
```

Nota: Otros métodos de la clase, que no se piden, permitirían posteriormente extraer la información a partir del mapa.

3) Escribir el pseudocódigo de un método que permite obtener para un grafo dirigido la conectividad de un conjunto de vértices, medida como el número de aristas que hay entre los elementos de ese conjunto (es decir, que tanto el origen como el destino de la arista sean elementos de ese conjunto). El grafo y el conjunto se le pasan como parámetros. Si algún vértice del conjunto no pertenece al grafo se retornará el valor -1.

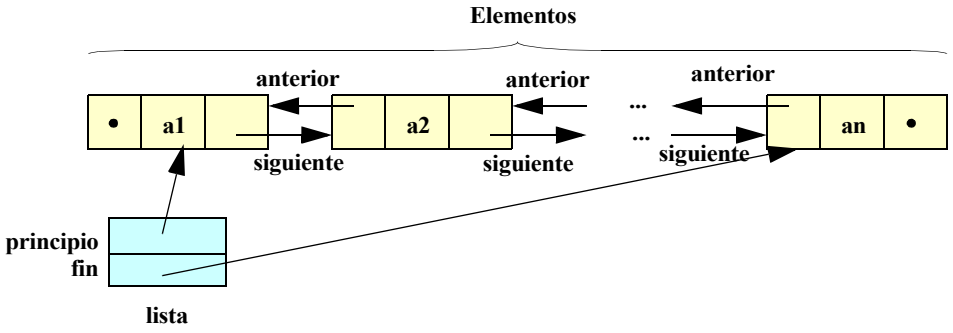
Examen de Estructuras de Datos y Algoritmos
(Ingeniería Informática)

Febrero 2007

Segunda parte (50% nota del examen)

4) Diseñar el pseudocódigo de la operación `containsValue()` de un mapa, que retorna `true` si el mapa contiene el valor indicado como parámetro, al menos una vez, y `false` en caso contrario. Esta operación es interna a la clase del mapa, y por tanto puede acceder a su estructura interna. Hacerlo para un mapa realizado mediante una tabla *hash* de troceado abierto, en la que cada casilla de la tabla es una lista de valores. Para la lista contamos con las operaciones e iterador habituales. Indicar la eficiencia de la operación resultante.

5) Se dispone una clase que define la estructura de listas doblemente enlazadas que se muestra en la figura. Escribir el pseudocódigo de un método interno a esta clase (y con acceso por tanto a las referencias que se muestran en la figura) al que se le pase otra segunda lista como parámetro y que, cambiando sólo un máximo de 3 referencias y sin duplicar las celdas de las listas, encadene esta segunda lista al final de la lista representada por el objeto actual. Tener en cuenta que la representación de una lista nula es con las referencias `principio` y `fin` nulas.



6) Escribir un método en Java que retorne la altura (máxima longitud de los caminos de la raíz hasta cada hoja) de un árbol que implementa la interfaz `Arbol` vista en clase. Indicar la eficiencia de esta operación.