Ejercicios Tema 4

Ejercicios Adaptados de Apuntes y Exámenes de EDA
Germán Moltó
gmolto@dsic.upv.es
Estructuras de Datos y Algoritmos
Escuela Técnica Superior de Ingeniería Informática
Universidad Politécnica de Valencia

Implementación de Métodos de LEG<E>

1. Implementa el método indiceDe con el siguiente perfil:

/** devuelve la posición de la primera aparición de e en una LEG (Secuencia), o devuelve -1 si e no está en el Grupo **/

public int **indiceDe**(E e);

2. Implementa el método:

/** SII 0<=i<talla(): inserta un Elemento x en la posición i-ésima de una LEG (Secuencia) **/

public void **insertar**(E x, int i);

3. Implementa el método:

public E **buscar**(E x) throws **ElementoNoEncontrado**;

LEGCircular

- Diseñar e Implementar el método de borrado de una **LEGCircular**.
  - Plantear análisis de situaciones y soluciones.
  - Escribir transcripción algorítmica.

public boolean **eliminar**(E x)

LDEG toString

- Escribir un método **toString**() de **LDEG** que obtenga una representación textual de los Nodos de la Lista en orden descendente, del último al primero

LDEG insertarEnFin

- Escribir el método **insertarEnFin**() de la clase **LDEG**.

public void **insertarEnFin**(E x);

eliminarMayor

1. Diseñar el método public void eliminarMayor(E x) que elimina de una LEG todos los elementos mayores que x
   - Nota: Si crees necesario utilizar alguno de los mecanismos vistos en teoría para restringir el tipo de E, puedes usarlos.
2. Diseñar el método public void eliminarMayor(E x) que elimina de una LEGOrdenada todos los elementos mayores que x.

Borra última aparición en LEG

- Se desea añadir a la clase LEG un nuevo método de borrado con la siguiente especificación:
  - Borra la última aparición del objeto x en la LEG retornándolo, salvo cuando no exista lo que advierte lanzando la excepción de usuario ElementoNoEncontrado;

public E borraUltimaAparicion(E x) throws ElementoNoEncontrado

Eliminar i-ésimo

- Se pide implementar el siguiente método en la clase LEG<E>:

public boolean **eliminar**(int i);

- Borra el i-ésimo Elemento de una LEG, 0<=i<talla().
- Si i no es una posición válida (i<0 ó i>=talla(),) lo advierte devolviendo false.

toStringOAMayoresQue

- Dada la clase LEGOrdenada vista en anteriores ejercicios, que mantiene los elementos ordenados y tiene la siguiente definición:
  - public class LEGOrdenada**<E extends Comparable<E>> extends LEG<E>**
- Se pide añadir un método que devuelva un String en el que aparezcan ordenados ascendentemente aquellos datos de una LEGOrdenada que sean mayores que uno dado.
  - Si no hay ninguno lanza la excepción ElementoNoEncontrado.

esMediana en LEGOrdenada

- Dada la clase LEGOrdenada vista en clase de teoría con la siguiente especificación:

  public class LEGOrdenada<E extends Comparable<E>> extends LEG<E>

- Se pide diseñar en la clase anterior un método esMediana tal que compruebe si un cierto x dado es el elemento mediana de una LEGOrdenada.
  - Debe indicar si el número de elementos mayores que x en una LEGOrdenada es igual al de menores que x;
  - Asúmase en el diseño que x puede o no ser un elemento de la LEGOrdenada y que no hay elementos duplicados en ella