Algoritmos y Estructuras de Datos

Guía de laboratorio 3
Lista doblemente enlazada

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Los siguientes ejercicios constituyen el Taller 3. Para aprobar el taller, todos los tests provistos deben pasar. Recuerden subir únicamente el archivo `ListaEnlazada.java` de la carpeta `main/java/aed` al campus.

Última fecha de entrega: domingo 20/10.

Consigna

En este taller deben implementar una *lista doblemente enlazada* con su correspondiente iterador.

En una *lista enlazada*, cada nodo apunta únicamente al nodo siguiente de la lista, mientras que en una *lista doblemente enlazada* cada nodo apunta, además, al nodo anterior. Por otro lado, una *lista doblemente enlazada* tiene un puntero al primer elemento y un puntero al último elemento. En la Figura 1 se puede ver un diagrama de la lista a implementar.

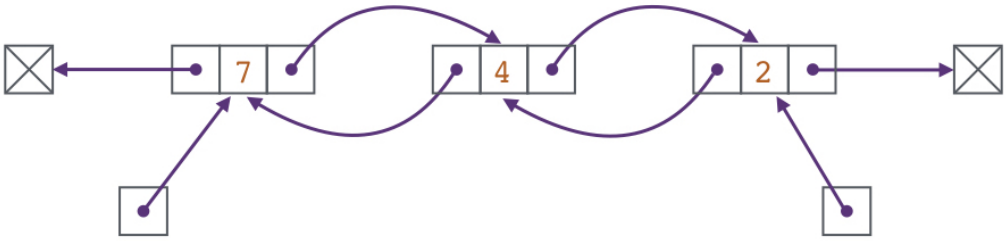

Figura 1: Lista doblemente enlazada que representa la secuencia $[7, 4, 2]$

Para resolver el taller cuentan con tres archivos: `Secuencia.java`, `Iterador.java`, y `ListaEnlazada.java`. El primero define la interfaz de una `Secuencia<T>`, mientras que el segundo define la interfaz de un iterador `Iterador<T>`. Estos dos archivos definen los métodos a implementar. El tercero corresponde a su implementación, bajo la clase `ListaEnlazada<T>`, la cual debe respetar la estructura de representación de una *lista doblemente enlazada* y constituye el archivo que deben completar.

La interfaz `Secuencia` declara los siguientes métodos:

- `int longitud();`
  Devuelve la cantidad de elementos que contiene la Secuencia.
- `void agregarAdelante(T elem);`
  Agrega un elemento al principio de la Secuencia.
- `void agregarAtras(T elem);`
  Agrega un elemento al final de la Secuencia.
- `void eliminar(Nat i);`
  Elimina el i-ésimo elemento de la Secuencia.
- `T obtener(Nat i);`
  Devuelve una referencia al elemento en la i-ésima posición de la Secuencia.
- `void modificarPosicion(Nat i, T elem);`
  Reemplaza al elemento en la i-ésima posición de la Secuencia por el elemento pasado por parámetro.
- `ListaEnlazada(ListaEnlazada<T> lista)`
  Constructor por copia de la lista pasada por parámetro.
- `String toString();`
  Devuelve un String con los elementos de la Secuencia, escritos entre corchetes y separados por coma (e.g., $[7, 4, 2]$).

- `Iterador<T> iterador;`
  Devuelve un iterador de la Secuencia.

Por su lado, la interfaz `Iterador` declara los siguientes métodos:

- `bool haySiguiente();`
  Devuelve **`true`** si hay un elemento siguiente al iterador, **`false`** en caso contrario.
- `bool hayAnterior();`
  Devuelve **`true`** si hay un elemento anterior al iterador, **`false`** en caso contrario.
- `T siguiente();`
  Devuelve el elemento siguiente al iterador y avanza al iterador a la siguiente posición.
- `T anterior();`
  Devuelve el elemento anterior al iterador y retrocede al iterador al iterador a la anterior posición.