UNIDAD TEMÁTICA 3: Listas, Pilas y Colas

PRACTICOS DOMICILIARIOS INDIVIDUALES – codigo básico JAVA para el TDA LISTA

En los próximos trabajos de aplicación sobre listas, aplicaremos esta estructura de datos y sus operaciones para la solución de problemas comunes cotidianos. Muchas son las situaciones que pueden representarse y resolverse fácilmente con listas (lista de cursos y sus alumnos, listas de canciones en un media-player, mi lista de amigos en facebook, etc etc.)

Como preparación previa entonces es conveniente implementar en JAVA y probar las operaciones más elementales de listas:

- Crear los elementos y la lista
- Insertar un elemento en una lista (al final, la lista no está ordenada)
- Buscar un elemento en una lista
- Listar (imprimir) todos los elementos de una lista
- Eliminar un elemento de una lista, dada su clave.

Ejercicio

1. Escribe lenguaje natural, precondiciones y postcondiciones y seudocódigo para cada operación
2. Dadas las interfaces siguientes, implementa las clases correspondientes y realiza un programa JAVA que te permita probar los métodos implementados.
3. Trae todo lo desarrollado a la próxima clase, en la cual se utilizarán las mejores versiones para construir en Equipo un programa que resuelva un problema concreto.

```
public interface INodo {
	/**
	 * Retorna la clave contenida en el nodo.
	 *
	 * @return Clave contenida en el nodo.
	 */
	public String getClave();

	/**
	 * Asigna una clave al nodo.
	 * @param unaClave Clave a asignar.
	 */
	public void setClave(String unaClave);

	/**
	 * Asigna el siguiente nodo al nodo actual.
	 * @param nodo Nodo a asignar como siguiente.
	 */
	public void setSiguiente(INodo nodo);

	/**
	 * Retorna el siguiente nodo al nodo actual.
	 * @return Siguiente nodo del actual
	 */
	public INodo getSiguiente();
}
```

```
public interface ILista {
	/**
	 * Método encargado de agregar un nodo al final de la lista.
	 *
	 * @param nodo - Nodo a agregar
	 */
	public void agregar(INodo nodo);

	/**
	 * Método encargado de buscar un nodo cuya clave es la indicada.
	 *
	 * @param clave - Clave del nodo a buscar.
	 * @return El nodo encontrado. En caso de no encontrarlo, retornar null.
	 */
	public INodo buscar(String clave);

	/**
	 * Método encargado de eliminar un nodo cuya clave es la indicada.
	 *
	 * @param clave Clave del nodo a eliminar.
	 * @return True en caso de que la eliminación haya sido efectuada con éxito.
	 */
	public boolean eliminar(String clave);

	/**
	 * Método encargado de imprimir en consola las claves de los nodos
	 * contenidos en la lista.
	 */
	public void imprimir();

	/**
	 *  Retorna un String con las claves separadas por el separador pasado por
parámetro.
	 * @param separador Separa las claves
	 * @return
	 */
	public String imprimir(String separador);
	/**
	 * Retorna la cantidad de elementos de la lista. En caso de
	 * que la lista este vacía, retornar 0.
	 * @return Cantidad de elementos de la lista.
	 */
	public int cantElementos();

	/**
	 * Indica si la lista contiene o no elementos.
	 * @return Si tiene elementos o no.
	 */
	public boolean esVacia();

	/**
	 * Retorna el primer nodo de la lista.
	 * @return Primer nodo de la lista.
	 */
	public INodo getPrimero();

}
```