

## Seguimiento 4

**Realizado por:** Tomás Ossa (A00372231) y Daniela Bonilla (A00372534)

### TAD Lista Enlazada Ordenada

Lista Enlazada Ordenada={Nodo=<Nodo>}

{inv: ListaEnlazada.Nodo.next>=ListaEnlazada.Nodo.prev  $\Leftrightarrow$  (si y solo si)  
ListaEnlazada.Nodo.prev != null }

Operaciones primitivas:

Lista Enlazada Ordenada= L.E.O

Métodos	Tipo Operación	Entradas	Salida
CrearNodo:	(Constructora)		→Nodo
InsertarNodo Ordenadamente:	(Modificadora)	Nodo x L.E.O	→ Lista Enlazada Ordenada
EstaVacía:	(Analizadora)	L.E.O	→ Booleana
ExisteNodo:	(Analizadora)	Nodo x L.E.O	→ Booleana
RemoverNodo:	(Modificadora)	Nodo x L.E.O	→ L.E.O
ObtenerNodo:	(Analizadora)	Nodo x L:E.O	→ Nodo

### createNode - CrearNodo :

\*Crear un nuevo nodo\*

{pre:TRUE}

{post: nodo creado. Si la lista leo está vacía entonces el primer nodo de la lista es n}

### insertNode - InsertarNodoOrdenadamente(n, leo )

\*Se inserta el Nodo n dentro de la lista leo de forma que su nodo siguiente (si existe) sea estrictamente mayor y su elemento anterior (si existe) sea menor o igual. El Nodo actual pasa a ser el Nodo insertado. \*

{pre: leo está inicializada}

{pre: n existe (es diferente de null, ya que debe ser comparada de alguna manera) }

{post: El tamaño de la lista leo aumenta en uno}
--

<b>isEmpty - EstaVacía (leo)</b>
----------------------------------

*Verificar si la lista enlazada ordenada contiene elementos (nodos) o no*
---

{pre:Lista enlazada ordenada exista}
--------------------------------------

{post: <code>True</code> si la L.E.O == null <code>False</code> de lo contrario }
--

<b>existingNode - ExisteNodo (n, leo)</b>
---

*Verifica la existencia de un Nodo n en la lista leo*
---

{pre:Lista enlazada ordenada exista}
--------------------------------------

{pre: n existe (es diferente de null, ya que no se puede buscar algo nulo) }
--

{post: <code>False</code> si n no existe en leo <code>True</code> si n existe en leo}
--

<b>deleteNode - RemoverNodo (n, leo)</b>
--

*Se elimina el elemento actual de la lista siempre y cuando exista, esta verificación se hace con el método ExisteNodo(). El nodo anterior al eliminado pasa a tener las relaciones de siguiente y anterior que tenía el nodo eliminado.*
---

{pre: lista enlazada ordenada está inicializada}
--

{pre: n existe (es diferente de null, ya que debe ser comparada de alguna manera) }
{pre: n existe en la lista leo }

{post: Nodo eliminado}
------------------------

<b>getNode - ObtenerNodo (n,leo)</b>
--------------------------------------

*Se obtiene un Nodo n de la lista siempre y cuando exista, esta verificación se hace con el método ExisteNodo().*
---

{pre:lista ordenada está inicializada}
--

{pre: n existe (es diferente de null, ya que debe ser comparada de alguna manera) }
{pre: n existe en la lista leo }

{post:Retorna el nodo a obtener}
----------------------------------