Algoritmos y Estructuras de Datos I

Ejemplo operaciones de listas – desde lo abstracto a la implementación

Universidad Católica del Uruguay

Operación, funcionalidad, comportamiento:

Insertar un nuevo elemento

- **Objetivo**: dado un nuevo elemento, agregarlo al final de la lista.
- **Lenguaje Natural**: si el nuevo elemento es un válido, recorrer la lista hasta el final y "enganchar" allí el nuevo elemento (apuntar el que era último a éste nuevo). Condición especial: si la lista está vacía, el nuevo elemento pasará a ser el "primero".

TDA Lista Insertar

- **PRECONDICIONES.**
  - El nuevo elemento debe existir, y ser del tipo correcto (el que corresponde a los elementos de la lista definida)
  - La etiqueta / clave del elemento debe ser valida
  - ¿hay estados específicos de la lista que deban ser considerados?
- **POSTCONDICIONES:**
  - Si la lista estaba vacía antes de insertar, ahora el nuevo elemento es el *primero* y la cantidad de elementos es 1
  - La lista contendrá un elemento más (la cantidad de elementos se ha incrementado en 1)
  - El elemento insertado se encuentra en la lista
  - El elemento se encuentra al final de la lista (apunta a nulo en su campo “siguiente”)

TDA Lista Insertar SEUDOCÓDIGO

Universidad Católica del Uruguay

TLista.Insertar (unElemento : tipoElemento)
COM
Si vacía entonces
primero ←unElemento
SiNo
Actual ← primero
Mientras Actual.siguiente <> nulo
Actual ← Actual. Siguiente
FinMientras
Actual.siguiente ← unElemento // insertamos al final….
FinSi
FIN

```java
/**
 * Inserta un INodo al final de la lista
 */
public void insertar(INodo unNodo) {
    if (esVacia()) {
        primero = unNodo;
    }else{
        INodo aux = primero;
        while (aux.getSiguiente() != null) {
            aux = aux.getSiguiente();
        }
        aux.setSiguiente(unNodo);
    }
}
```

TDA Lista Insertar
CASOS DE PRUEBA

Universidad Católica del Uruguay

- A partir de la lista vacía, insertar un elemento conocido y comprobar que
  - El elemento está en la lista
  - Cantidad de elementos es 1
- Luego de varias inserciones, agregar un nuevo elemento y comprobar que:
  - El elemento está en la lista
  - El elemento es el último de la lista
  - Cantidad de elementos se ha incrementado en 1
- OTROS????

Universidad Católica del Uruguay

```java
@Test
public void testInsertar1() {
    INodo nodo = new Nodo(1);
    assertTrue(lista.cantElementos()==0);
    lista.insertar(nodo);
    assertTrue(lista.cantElementos()==1);
}
```

Universidad Católica del Uruguay

```java
@Test
public void testInsertar2() {
    INodo nodo = new Nodo(1);
    lista.insertar(nodo);
    assertTrue(lista.buscar(1)!=null);
}
```

```java
@Test
public void testInsertar3() {
    INodo nodo = new Nodo(1);
    lista.insertar(nodo);
    INodo resultado = lista.buscar(1);

    //Al ser el único elemento de la lista, la
    //referencia al siguiente debe ser nula
    assertNull(resultado.getSiguiente());
}
```

Operación, funcionalidad, comportamiento: *buscar y devolver un elemento que tenga cierta etiqueta*

- **Objetivo**: dada una etiqueta, devolver el elemento de la lista que tenga esa etiqueta o indicar que no hay en la lista un elemento con esa etiqueta.
- **Lenguaje Natural**: Si la lista no está vacía, recorrerla comenzando por el principio y comparando la etiqueta buscada con la de cada elemento visitado. Si se encuentra una coincidencia, devolver el elemento correspondiente. Si, de lo contrario, se llega al final de la lista sin coincidencia, o si la lista está vacía, indicar que la lista no contiene ningún elemento con clave como la indicada.

TDA Lista Buscar

- **PRECONDICIONES.**
  - La etiqueta a buscar es válida (discutir)
  - ¿hay otros estados específicos de la lista que deban ser considerados?
  - ¿Lista no vacía? Discutir….
- **POSTCONDICIONES:**
  - La lista no se ve alterada de ninguna forma (la cantidad de elementos no varía, el “primero” no se altera, etc.)
  - En caso de existir, el elemento devuelto no se ve afectado de ninguna manera.

TDA Lista Buscar SEUDOCODIGO

Universidad Católica del Uruguay

```
TLista.buscar (unaEtiqueta : tipoEtiqueta): Telemento
// devuelve un TElemento
COM
  Si vacía entonces
     Devolver nulo                                  // salir
  SiNo
    Actual ← primero
     Mientras Actual  <>  nulo
       Si Actual.Etiqueta = unaEtiqueta
Devolver Actual                                     //salir
   Actual ← Actual. Siguiente
   FinMientras
   Devolver nulo                                    // salir
  FinSi
FIN
```

TDA Lista Buscar SEUDOCODIGO

TLista.buscar (unaEtiqueta : tipoEtiqueta): TElemento // devuelve un TElemento

COM

..tempElem ←nulo

Actual ← primero

Mientras Actual <> nulo

Si Actual.Etiqueta = unaEtiqueta

tempElem ← Actual

salir bucle //break

FinSi

Actual ← Actual.Siguiente

FinMientras

Devolver tempElem // **salir**

FIN

```java
/**
 * Busca un nodo en la lista a partir de la etiqueta.
 * La etiqueta puede ser cualquier tipo Comparable
 */
public INodo buscar(Comparable etiqueta) {
    if (esVacia()) {
        return null;
    } else {
        INodo aux = primero;
        while (aux != null) {
            if (aux.getEtiqueta().equals(etiqueta)) {
                return aux;
            }
            aux = aux.getSiguiente();
        }
    }
    return null;
}
```

TDA Lista Buscar
CASOS DE PRUEBA

Universidad Católica del Uruguay

- A partir de la lista **vacía**
  - Al invocar *buscar* sobre la lista vacía, con cualquier etiqueta, la operación devuelve *nulo*
  - Insertar un elemento conocido y comprobar que Al invocar *buscar* con la etiqueta del elemento que se ha insertado, se lo encuentra (es decir, devuelve un elemento con etiqueta igual a la buscada)
- Insertar varios elementos – con etiquetas conocidas- en la lista y
  - Comprobar que encuentra un elemento entre los recientemente insertados
  - Comprobar que devuelve *nulo* cuando se invoca con una etiqueta que sabemos que no está.
  - Cantidad de elementos se ha incrementado en 1
- OTROS????

```java
@Test
public void testBuscar_listaVacia() {
    INodo resultado = lista.buscar(12873);
    assertNull(resultado);
    assertTrue(lista.cantElementos()==0);
}
```

```java
@Test
public void testBuscarRecienInsertado() {
    INodo nuevo = new Nodo("ETI:12983");
    lista.insertar(nuevo);
    INodo resultadoBusqueda = lista.buscar("ETI:12983");
    assertNotNull(resultadoBusqueda);
}
```