Estructuras para Listas

Algoritmos y Estructuras de Datos

Unidad 3

Listas

Listas

- Una lista es una estructura de datos lineal
- Esto significa que uno de sus elementos está unido solamente con una anterior o uno posterior

Representación Formal

- Formalmente se entiende una lista como una secuencia que contiene cero o más elementos del mismo tipo

$$a_1, a_2, \ldots, a_n \qquad n \geq 0$$

- Si n = 0, se dice que la lista está vacía
- Si n es distinta de cero, se dice que el primer elemento es la "cabeza" de la lista y el último es la "cola" de la lista.

Representación en Memoria

- Una lista puede representarse de dos maneras:
  - Secuencial - Estática
  - Ligada - Dinámica

Representación Ligada

- La representación ligada ofrece varias ventajas respecto a una representación secuencial
- Solo se utiliza el espacio que se necesita
- No está limitada a un tamaño inicial
- No es necesario re organizar los elementos de la estructura al insertar o borrar

TDA Lista

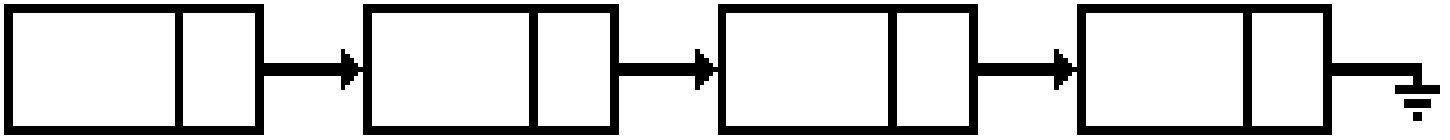
- Como cualquier TDA, la Lista tiene definidas un conjunto de operaciones que se pueden realizar con ella

Operaciones de una Lista

Operación	Descripción
lista()	Crear una lista vacía
bool vacia()	Indica si una lista está vacía o no
void inserta (elemento e)	Inserta un elemento en la lista
void borra (elemento e)	Borra un elemento de la lista
int busca (elemento e)	Regresa la posición de un elemento buscado en la lista

Listas Ligadas

- Se trata de la estructura de tipo Lista más sencilla que existe
- De manera conceptual, una representación secuencial se puede representar de la siguiente manera:

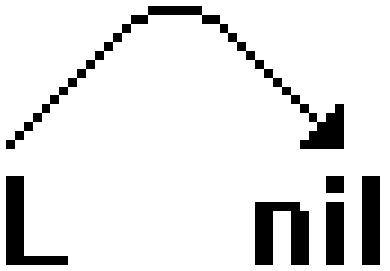


Listas Ligadas

- Se trata de un conjunto de nodos en dónde cada uno de ellos contiene al menos dos elementos:
  - La información que almacena
  - Un elemento que le indica quién es el que está después de él

Crear una Lista

- Cuando se crea una lista, ésta debe estar vacía



- De manera similar, al preguntar si una lista está vacía, su primer elemento deberá ser nulo (***null***)

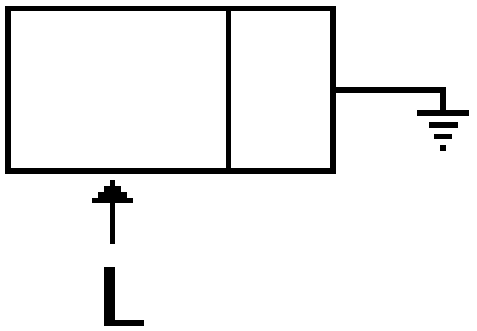
Insertar un Elemento

- Para insertar un elemento, se debe considerar si:
  - La lista está vacía
  - La lista ya contiene elementos

- Si ya se tienen elementos, se considera:
  - Si se inserta al final de la lista
  - Si se inserta al inicio de la lista
  - Si se inserta en una posición en particular

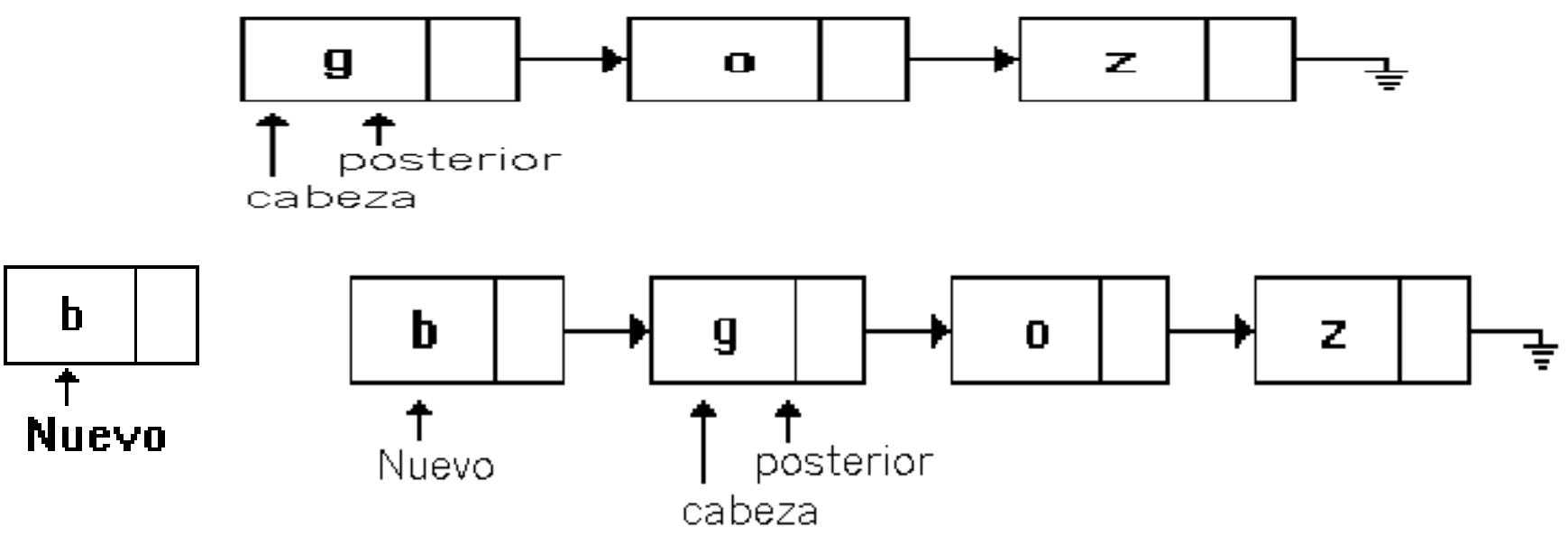
Insertar en una Lista Vacía

- Se ha determinado si la lista está vacía
- Se crea un nodo con la información a almacenar
- La liga de este nodo apuntará a nulo (***null***)



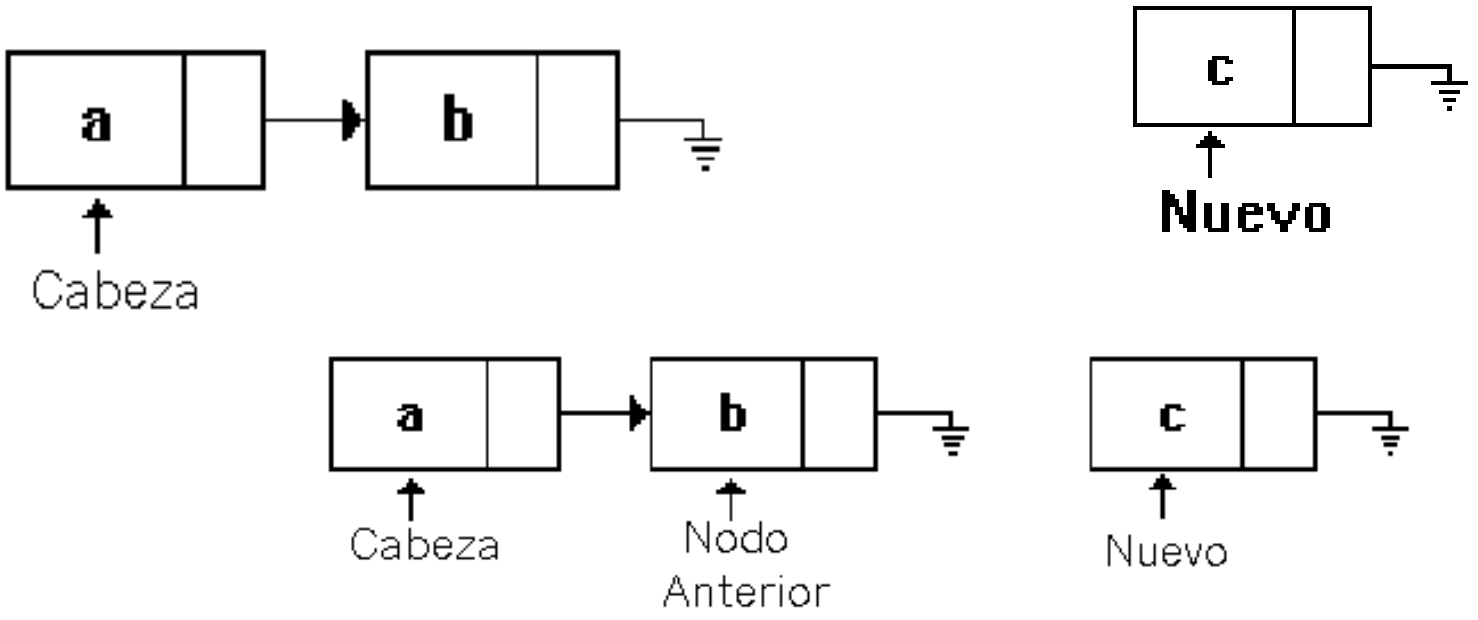
Insertar al Inicio de la Lista

- Se debe crear un nuevo nodo con la información a almacenar
- La liga de este nuevo nodo apuntará a la cabeza de la lista



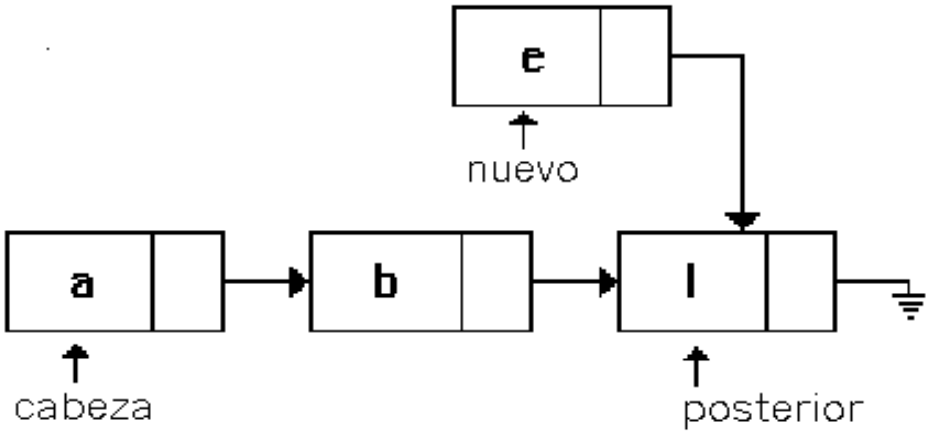
Insertar al Final de la Lista

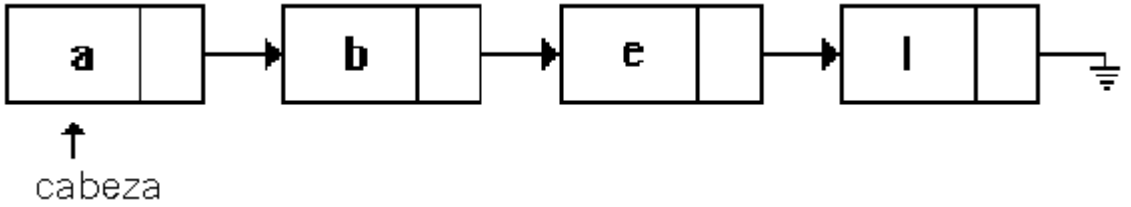
- Crear un nodo con el elemento a almacenar
- Llegar al final de la lista existente
- La liga del último nodo de la lista, apuntará al nuevo nodo creado



Insertar en una Posición en Particular

- Se debe crear un nuevo nodo con el elemento a insertar
- Buscar en la lista la posición a insertar





Eliminar un Elemento

- Al momento de eliminar un elemento, se tienen distintos casos:
  - El elemento es el único
  - El elemento es la cabeza de la lista
  - El elemento es la cola de la lista
  - El elemento está en una posición determinada

Eliminar un Elemento Único

- La lista quedará vacía por lo que será igual a nulo (***null***)

Eliminar la Cabeza de la lista

- Se debe tener cuidado de no eliminar la lista por completo
- La nueva cabeza ahora será el nodo al que apunta la cabeza actual

Eliminar la Cola de la Lista

- Se debe recorrer la lista hasta llegar al penúltimo elemento
- La liga de este elemento ahora apuntará a nulo

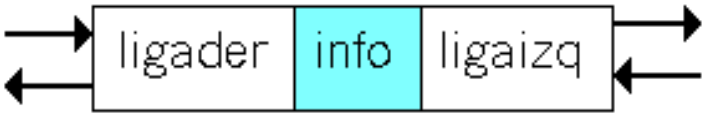
Eliminar un Elemento Intermedio

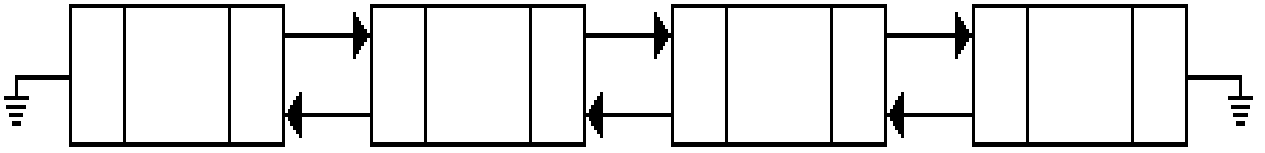
- Se debe buscar el elemento anterior al de la posición a eliminar
- La liga de ese elemento apuntará al elemento que apunta la liga del nodo a eliminar

Listas Doblemente Ligadas

Definición

- Se trata de una mejora a las listas ligadas, ya que pueden ser recorridas en "cualquier sentido"
- Una Lista Doblemente Ligada tiene acceso al elemento anterior y al siguiente



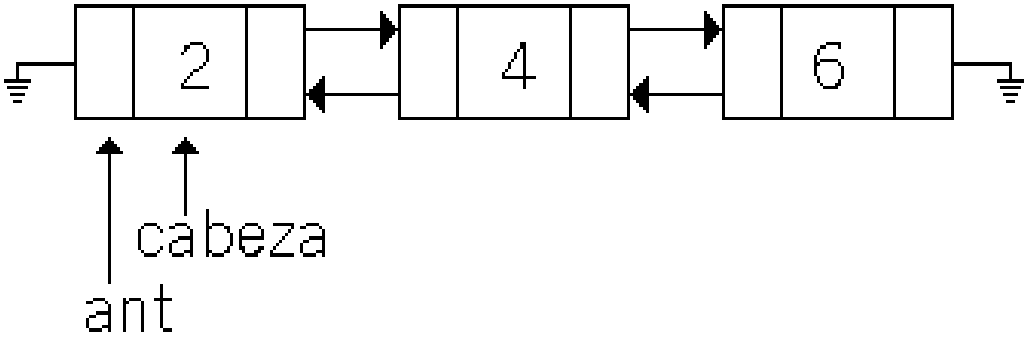


Operaciones

- Las operaciones de Creación y determinar si la lista está vacía, son las mismas que para las listas Ligadas

Búsqueda

- Buscar un elemento es igual que en las listas ligadas, con la ventaja de que si se quiere insertar en orden, ya se tiene acceso a la posición en dónde se colocaría el nuevo elemento

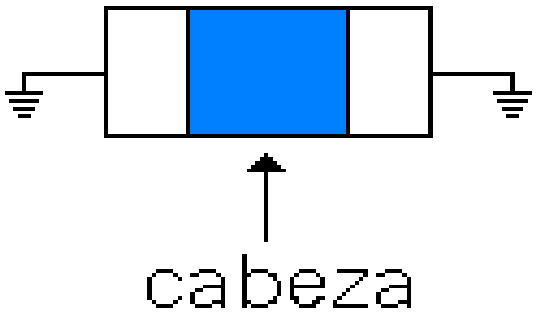


Insertar

- Al igual que en las listas ligadas, se tienen diferentes casos:
  - Insertar en una lista vacía
  - El elemento es la cabeza de la lista
  - El elemento es la cola de la lista
  - El elemento se inserta en una posición intermedia

Insertar en una lista vacía

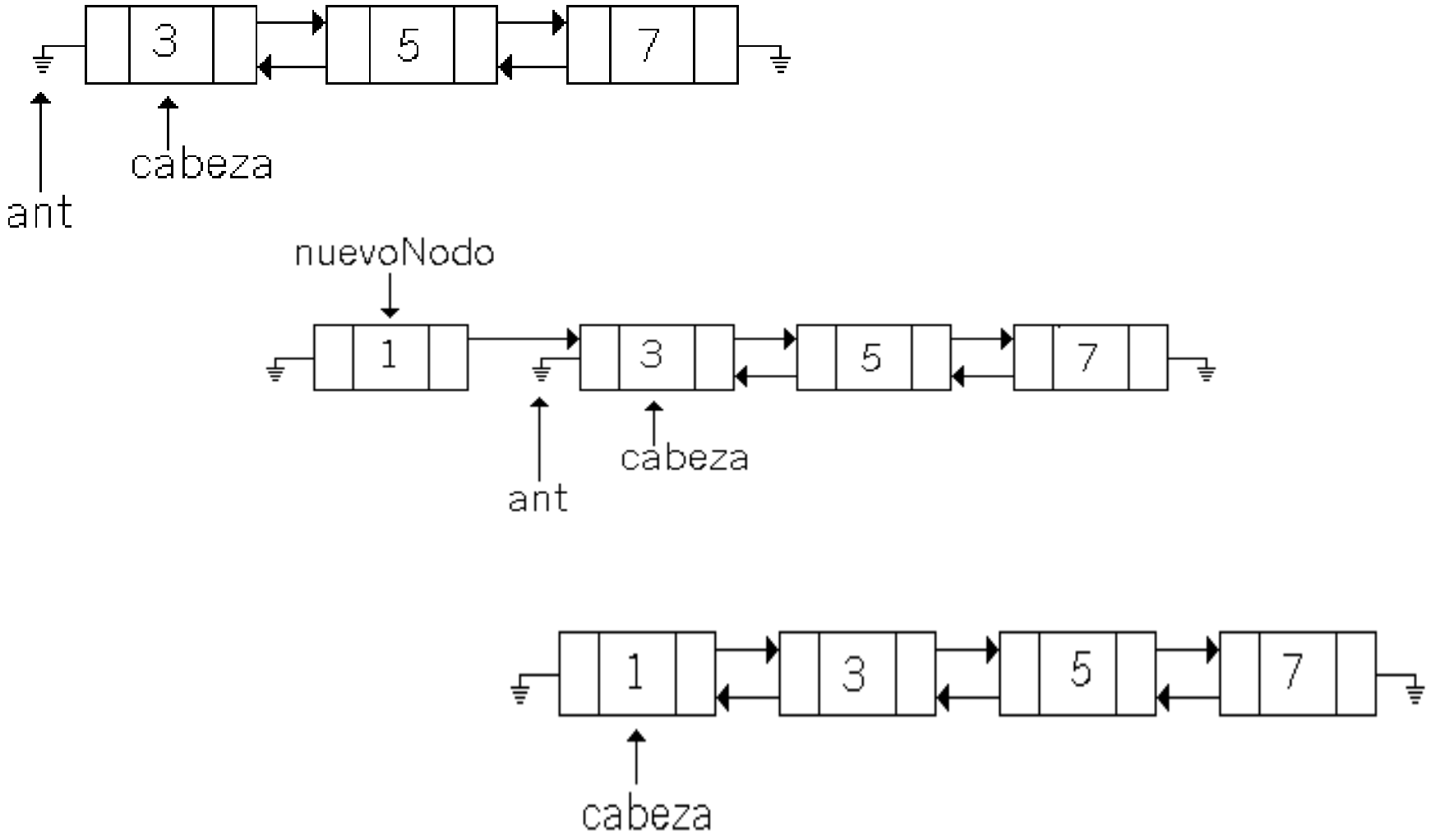
- Se crea un nuevo elemento y se le asigna a la cabeza de la lista
- Las dos referencias (anterior y siguiente) apuntarán a nulo



Insertar un Elemento como Cabeza

- Se crea un nuevo nodo
- La liga siguiente de este apuntará a la cabeza de la lista
- La liga anterior de la cabeza de la liga apuntará al nuevo nodo
- La liga anterior del nuevo nodo apuntará a nulo

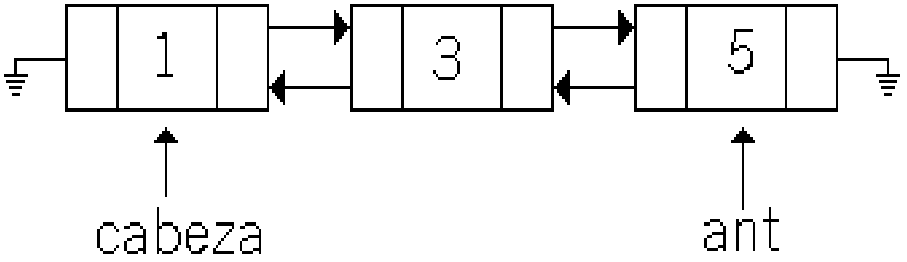
Insertar un Elemento como Cabeza

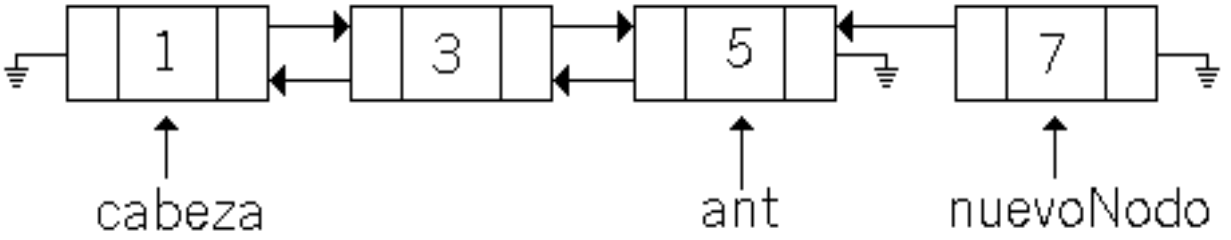


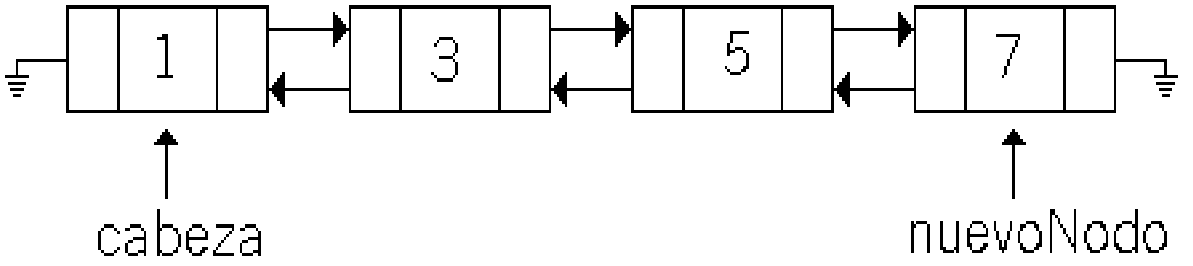
Insertar como Cola de la Lista

- Se crea un nuevo nodo
- La liga anterior del nuevo nodo apunta a la cola de la lista
- La liga siguiente de la cola de la lista apunta al nuevo nodo
- La liga siguiente del nuevo nodo, apunta a nulo

Insertar como Cola de la Lista



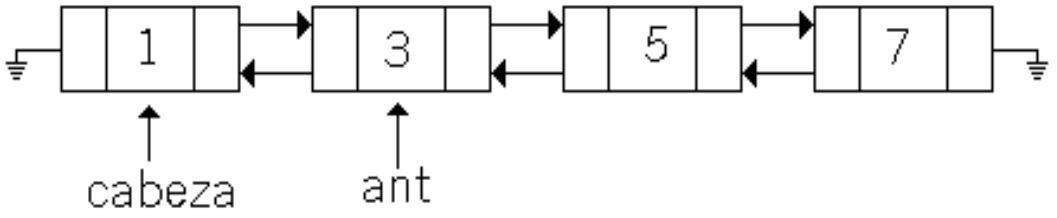


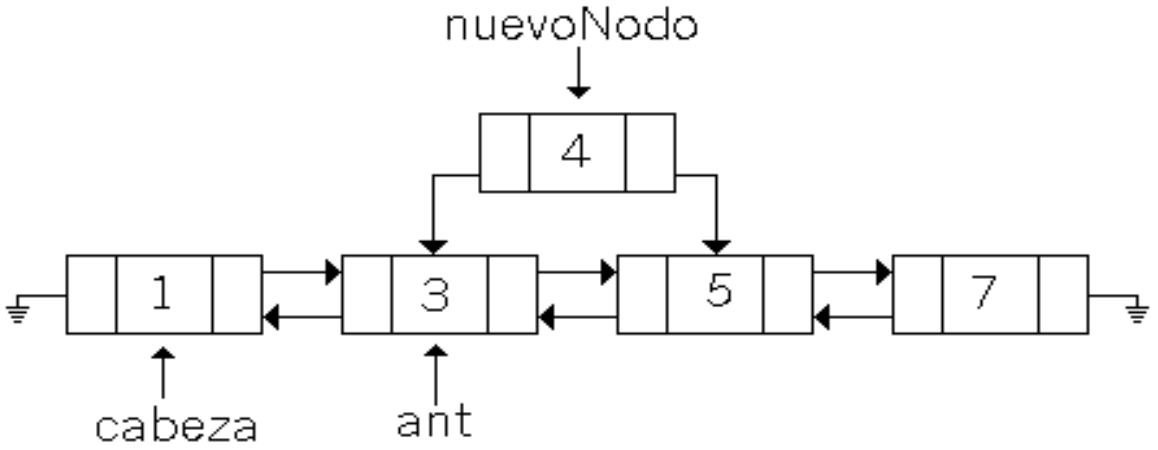


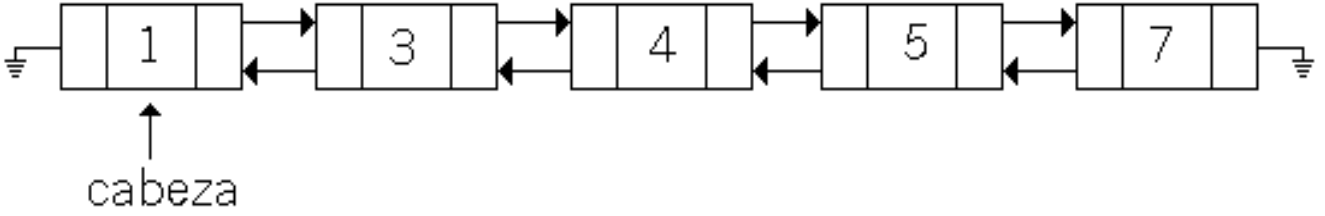
Insertar en Medio de dos Nodos

- Se crea un nuevo nodo
- Se busca la posición indicada
- La liga anterior del nuevo nodo apunta al nodo anterior
- La liga siguiente del nodo anterior apunta al nuevo nodo
- La liga siguiente del nuevo nodo apunta al nodo siguiente
- La liga anterior del nodo siguiente apunta al nuevo nodo

Insertar en Medio de los Nodos







Eliminar Elementos

- De manera similar a la lista ligada, se tienen las siguientes opciones:
  - Borrar un único elemento
  - Borrar la cabeza de la lista
  - Borrar la cola de la lista
  - Borrar un nodo intermedio

Borrar el único Elemento

- Es el caso más sencillo, la cabeza de la lista se referencía a nulo

Borrar la Cabeza de la Lista

- Se asigna como cabeza el elemento siguiente a la cabeza actual
- La liga anterior de la nueva cabeza apuntará a nulo

Borrar la Cola de la Lista

- La liga siguiente del elemento anterior de la cola actual se asignará a nulo

Borrar un Elemento Intermedio

- Se encuentra la posición del elemento a borrar
- La liga siguiente del nodo anterior apuntará al nodo siguiente del nodo a eliminar
- La liga anterior del nodo siguiente apuntará al nodo anterior del nodo a eliminar

Pilas y Colas

Listas Especiales

- Las Pilas y las Colas son dos casos especiales de listas ligadas, aunque también pueden implementarse con arreglos
- Su diferencia se basa en la forma en que se insertan elemento (Push) y se sacan de la misma (Pop)

Pilas

- Las pilas (*stack*) son una estructura muy utilizada en computación con acceso de tipo LIFO
- LIFO significa "*Last In First Out*" es decir "*Último en Entrar Primero en Salir*"

Operaciones de una Pila

- Las operaciones de una Pila son:
  - Crear
  - Tamaño
  - Insertar (*Push*)
  - Quitar (*Pop*)
  - Mostrar último elemento insertado
  - Vacía

Push y *Pop*

- Las operaciones *Push* y *Pop* son inserciones y eliminaciones en la pila
- *Push* representa una inserción al inicio de la pila / lista
- *Pop* representa eliminar el primer elemento de la pila / lista

Colas

- Las colas / filas (*queue*) son una estructura muy utilizada en computación con acceso de tipo FIFO
- FIFO significa "*First In First Out*" es decir "*Primero en Entrar Primero en Salir*"

Operaciones de una Cola

- Las operaciones de una Cola son:
  - Crear
  - Tamaño
  - Insertar (*Push*)
  - Quitar (*Pop*)
  - Mostrar primer elemento insertado
  - Vacía

Push y *Pop*

- Las operaciones *Push* y *Pop* son inserciones y eliminaciones en la cola
- *Push* representa una inserción al final de la cola / lista
- *Pop* representa eliminar el primer elemento de la pila / lista