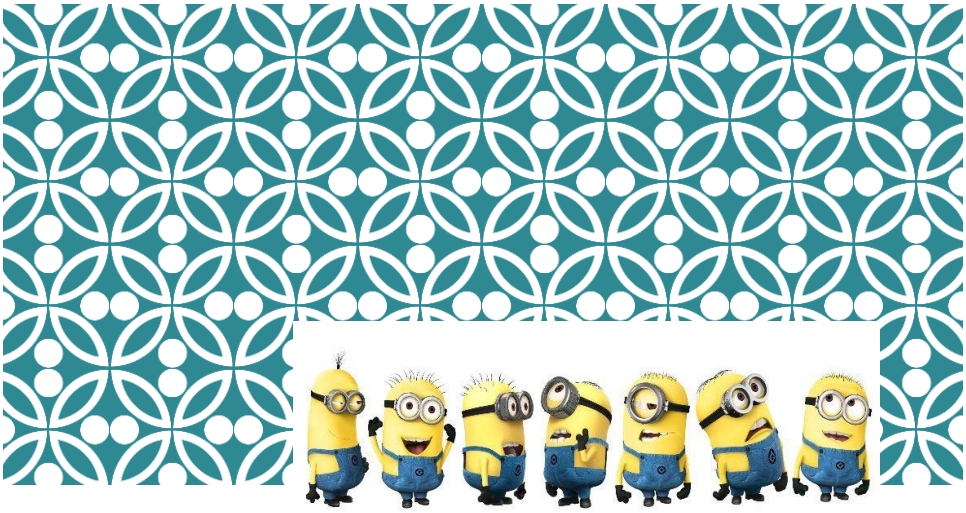
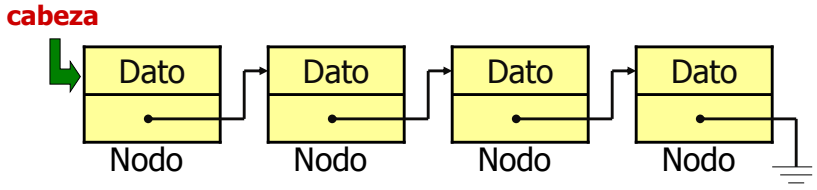


UNIDAD 2: ESTRUCTURAS DE DATOS DINÁMICAS

Listas ligadas simples
y dobles
Pilas y colas

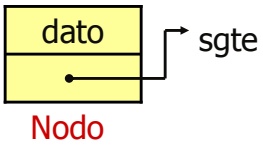
LISTAS LIGADAS SIMPLE

- Una lista ligada o enlazada es una estructura de datos dinámica
- Es una colección lineal de objetos, conocidos como nodos, que están conectados mediante ligas o enlaces de apuntadores.



LISTAS LIGADAS

- Cada nodo esta constituido por dos campos: Información (que puede ser cualquier tipo de dato o un objeto) y enlace (referencia o apuntador).
- Un nodo se implementa usando una clase auto-referenciada. Una clase autoreferenciada contiene una referencia que apunta a un objeto de la misma clase.



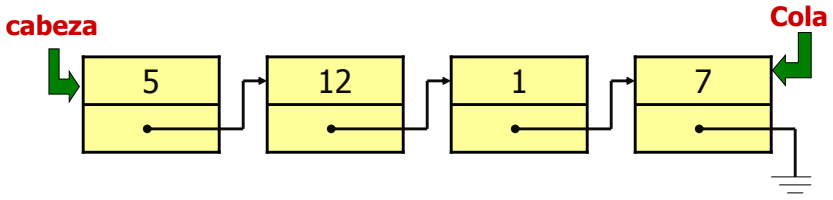

```
Class Nodo {
    int dato; //información
    Nodo sgte; //enlace

    Nodo( );
    Nodo(int x);
};
```

3

LISTAS LIGADAS SIMPLES

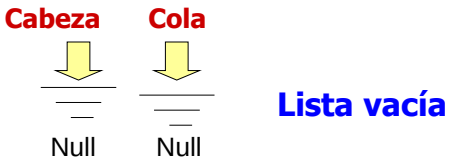
- Para marcar el inicio o cabeza de la lista se utiliza una **referencia** a nodo, llamada normalmente cabeza, inicio o principio.
- Opcionalmente se puede contar con una referencia al último nodo de la lista, llamado generalmente cola, final o último.
- La cabeza de la lista es la única forma de acceder a los elementos de esta.



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 4

LISTAS LIGADAS

- Una lista ligada sin ningún elemento se llama lista vacía, y se representa poniendo la variable cabeza igual a Null.
- Si existe cola también debe ser igual a Null.



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 5

OPERACIONES EN LISTAS LIGADAS

- Las operaciones básicas que podemos realizar en una lista son:
  - **Creación**: crear una lista vacía.
  - **Inserción**: añadir elementos (nodos) a la lista en alguna posición específica:
    - Al inicio
    - Al final
    - Entre dos nodos de la lista
  - **Recorrido**: Moverse sobre los elementos de la lista, partiendo del inicio y llegando al final de la misma.

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 6

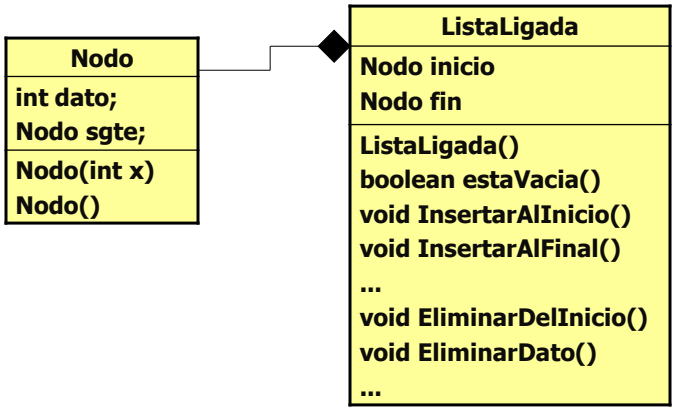
OPERACIONES EN LISTAS LIGADAS

- **Eliminación**: quitar un nodo de la lista de alguna posición específica
  - Al inicio
  - Al final
  - Dato específico x (que puede estar entre dos elementos)
- **Mostrar**: Recorrer la lista mostrando los elementos en esta.
- **Búsqueda**: Recorrer la lista para verificar la existencia de un elemento dado dentro de la lista.
- **Ordenamiento**: colocar en algún orden específico (ascendente o descendente) los elementos de la lista.

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 7

CLASE LISTA LIGADA

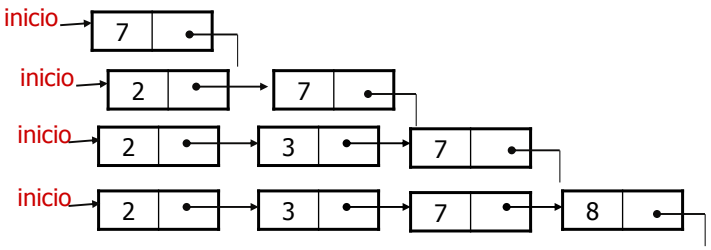
- Dadas las operaciones de una lista enlazada, podemos modelar la clase ListaLigada de la siguiente forma



8

ACTIVIDAD COLABORATIVA

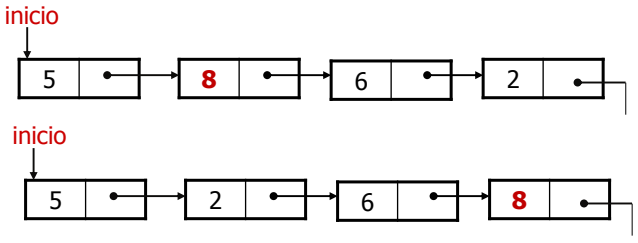
- En Binas, implementar en Java un método que inserte ordenadamente cada elemento como va llegando.
- Por ejemplo, si la secuencia de entrada es: 7, 2, 3, 8



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 9

ACTIVIDAD COLABORATIVA

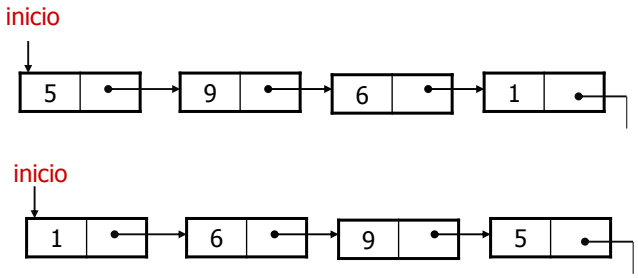
- En Binas, implementar en Java un método que busque el dato mayor de la lista ligada y lo inserte al final de la misma. Sólo si tal valor es mayor que el número de nodos actual de la lista.
- Ejemplo: el nodo con valor 8 se debe pasar después del nodo con valor 2.



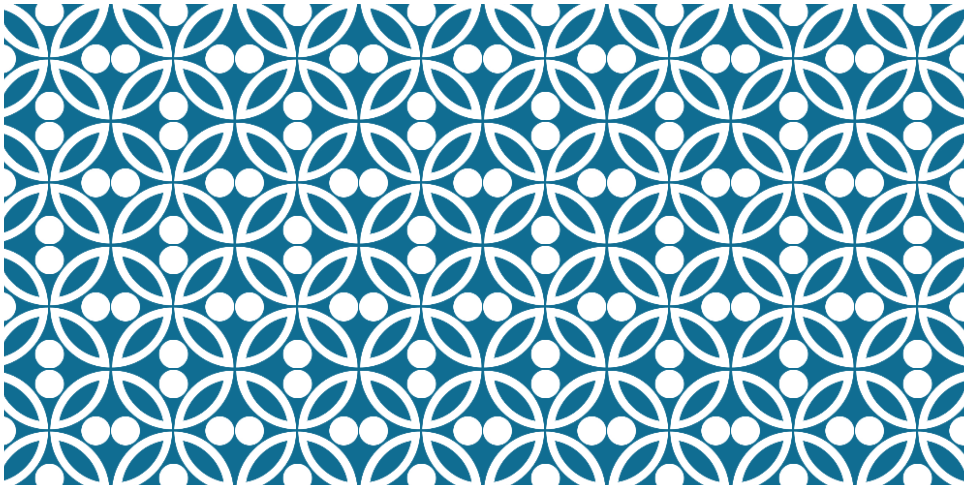
M.C. YALU GALICIA HDEZ. (FCC/BUAP) 10

ACTIVIDAD COLABORATIVA

- En Binas, implementar en Java un método que invierta el contenido de una lista, moviendo únicamente referencias



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 11

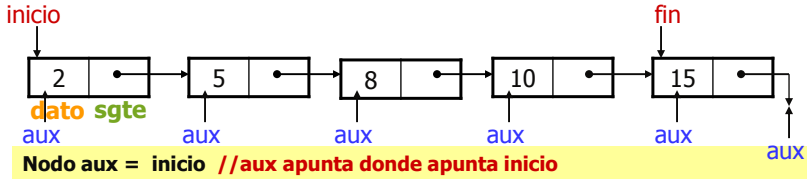


DIAPOSITIVAS DE REPASO!

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 12

MOSTRAR EL CONTENIDO

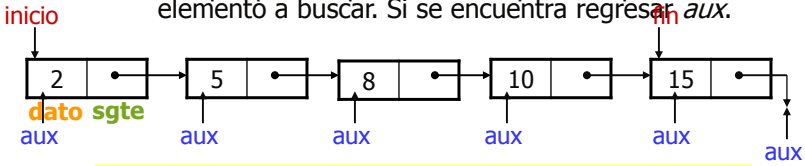
- Para mostrar el contenido de la lista tenemos que considerar dos situaciones:
  - Si la lista esta vacía (inicio y fin son Null), entonces no hacer nada solo indicar "lista vacía"
  - Si la lista tiene al menos un elemento
    - Crear una referencia auxiliar e inicializarla con el inicio de la lista (el apuntador inicio nunca se debe perder).
    - Recorrer la lista utilizando la variable auxiliar hasta que sea null (fin de la lista)




```
Nodo aux =  inicio  //aux apunta donde apunta inicio
while  (aux != null)  {
        System.out.println( aux. dato); //dato del nodo apuntado por aux
        aux = aux. sgte; //aux apunta donde apunta sgte
}
```

BÚSQUEDA DE UN DATO

- Para buscar un elemento de la lista tenemos que considerar dos situaciones:
  - Si la lista esta vacía (inicio y fin son null), entonces regresar null para indicar que no se encontró
  - Si la lista tiene al menos un elemento
    - Inicializar un apuntador auxiliar con el inicio de la lista
    - Recorrer la lista hasta que *aux* sea null al mismo tiempo que compara el nodo apuntado por *aux* con el elemento a buscar. Si se encuentra regresar *aux*.

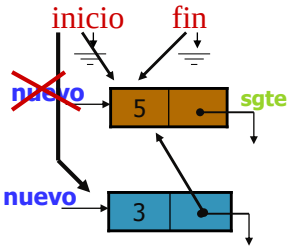



```
Nodo aux =  inicio  //aux apunta donde apunta inicio
while  (aux != null && aux.dato != x) //busca el dato
        aux = aux.sgte;
return aux;
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 14

INSERTAR AL INICIO

- Para insertar al inicio de una lista enlazada tenemos que considerar dos situaciones
  - Si la lista esta vacía (inicio y fin son null)
  - Si la lista tiene al menos un elemento

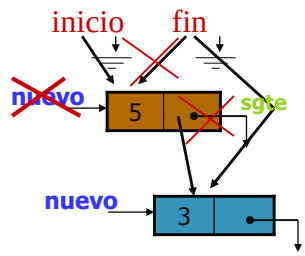



```
Si estaVacia() entonces
   Nodo nuevo = new Nodo(dato);
   inicio = nuevo; //inicio apunta donde apunta nuevo
   fi = nuevo;
Sino
    Nodo nuevo = new Nodo(dato);
    nuevo.sgte = inicio;
    inicio = nuevo;
Fin_si
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 15

INSERTAR AL FINAL

- Para insertar al final, nuevamente tenemos que considerar dos situaciones
  - Si la lista esta vacía (inicio y fin son Null)
  - Si la lista tiene al menos un elemento

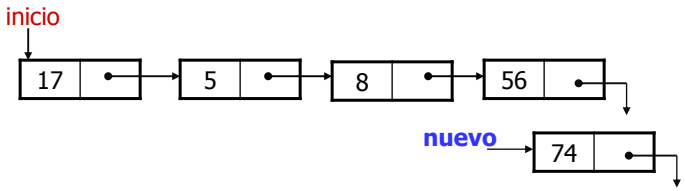



```
Si estaVacia() entonces
   Nodo nuevo = new Nodo(dato);
   inicio = nuevo;
   fin = nuevo;
Sino
    Nodo nuevo = new Nodo(dato);
    fin.sgte = nuevo; //el nodo final se liga al nuevo
    fin = nuevo; //se recorre fin
Fin_si
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 16

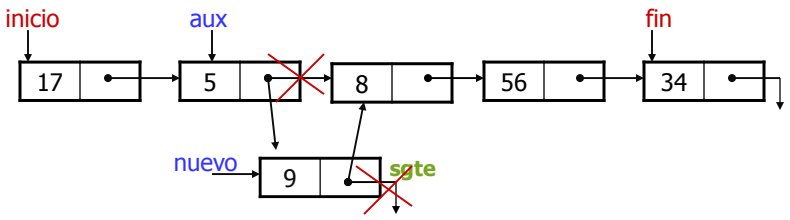
ACTIVIDAD COLABORATIVA

- En binas, insertar un elemento al final de la lista, suponiendo que no se tiene la variable fin.



INSERTAR DESPUÉS DE UN NODO ESPECÍFICO

- Para insertar entre dos nodos, tenemos que considerar dos situaciones:
  - Si la lista esta vacía (inicio y fin son Null)
  - Si la lista tiene al menos un elemento, localizar posición del nuevo nodo a insertar. Por ejemplo después del 5



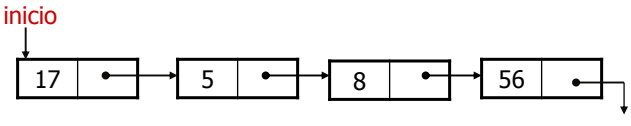
SOLUCIÓN

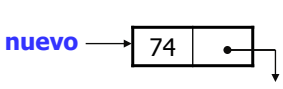
```
void insertarDespuesDe(int d, int y)
{
 Nodo aux =  inicio  //aux apunta donde apunta inicio
 while  (aux.sgte != null && aux.sgte.dato != d) //busca el dato
       aux = aux.sgte;
 if(aux.sgte != null)  {// si existe el dato
       Nodo nuevo = new Nodo(y); //crea nuevo nodo
       nuevo.sgte = aux.sgte;  //inserta primero el nuevo en la lista
       aux.sgte = nuevo;  //actualiza la lista
 }
 else
       System.out.println ("Dato no encontrado, no se insertó nuevo ");
}
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 19

ACTIVIDAD COLABORATIVA

- En binas, insertar un nuevo elemento **antes** de un dato específico. Por ejemplo antes del 8
- Considerar el caso en que la lista no está vacía y cuando el dato está al inicio.

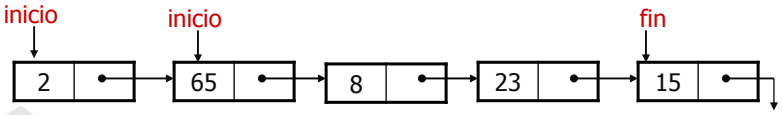


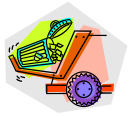


M.C. YALU GALICIA HDEZ. (FCC/BUAP) 20

ELIMINAR DEL INICIO

- Para eliminar del inicio tenemos que considerar tres situaciones:
  - Si la lista esta vacía (inicio y fin son Null), no hacer nada, solo indicar "lista vacía"
  - Si la lista tiene solo UN elemento (inicio == fin), entonces eliminarlo y actualizar inicio y fin a Null.
  - Si la lista tiene más de un elemento



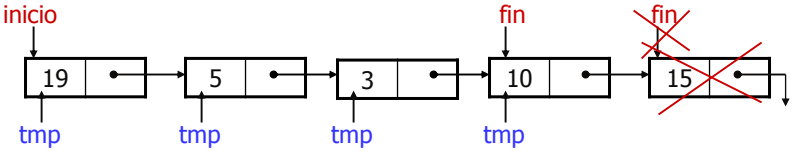


```
inicio = inicio.sgte; //inicio se recorre uno
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 21

ELIMINAR DEL FINAL

- Para eliminar del final tenemos que considerar tres situaciones:
  - Si la lista esta vacía (inicio y fin son null), no hacer nada, solo indicar "lista vacía"
  - Si la lista tiene solo UN elemento (inicio == fin), entonces eliminarlo y actualizar inicio y fin a Null.
  - Si la lista tiene más de un elemento

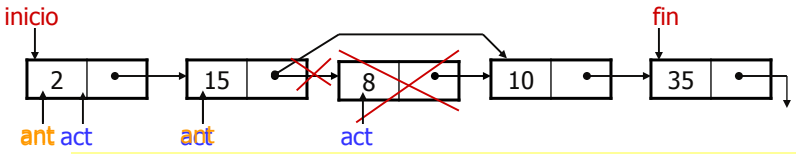



```
Nodo tmp =  inicio;  //tmp apunta donde apunta inicio
while (tmp.sgte != fin)  tmp = tmp.sgte;
fin = tmp; //fin apunta donde apunta tmp
fin.sgte = null;
```

22 22

ELIMINAR UN DATO

- Para eliminar un elemento específico, tenemos que considerar varias situaciones:
  - Localizar el elemento a borrar (por ejemplo el 8)
    - Cuantos apuntadores auxiliares se necesitan?
    - Donde se deben colocar?, en el nodo con el dato o en el nodo anterior?

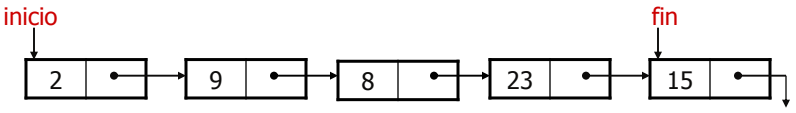



```
Nodo act, act;   ant = act = inicio;
while (act != null && act.dato != x) { ant=act;  act=act.sgte; }
if (act != null) {   ant.sgte = act.sgte;
                     if(act == fin) fin = ant;
 }
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 23

ELIMINAR UN DATO

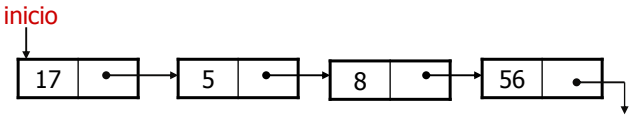
- Considerar:
- Si la lista esta vacía (inicio y fin son Null), no hacer nada, solo indicar "lista vacía"
- ¿Qué pasa si el elemento a borrar está al inicio? (por ejemplo el 2)
- ¿Qué pasa si el elemento a borrar está al final? (por ejemplo el 15)



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 24

ACTIVIDAD COLABORATIVA

- En binas, eliminar un elemento **antes** de un dato específico. Por ejemplo el nodo que está antes del 8



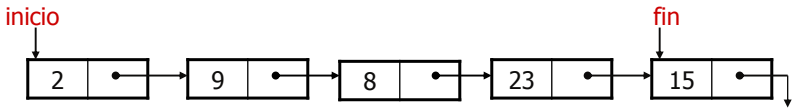
M.C. YALU GALICIA HDEZ. (FCC/BUAP) 25

TIPOS DE LISTAS LIGADAS

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 26

TIPOS DE LISTAS LIGADA

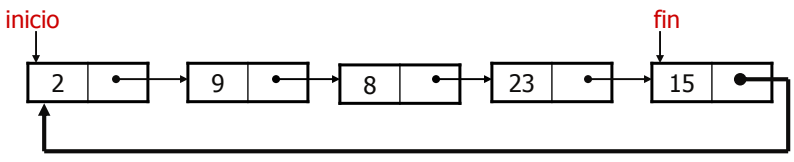
- De acuerdo con su conformación, las listas se pueden clasificar en:
  - Lista ligada simple
    - Es aquella en la que cualquiera de sus nodos tiene un único inmediato sucesor y un único inmediato antecesor, exepto el primero y el último nodo que no tienen antecesor ni sucesor respectivamente.



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 27

TIPOS DE LISTAS LIGADAS

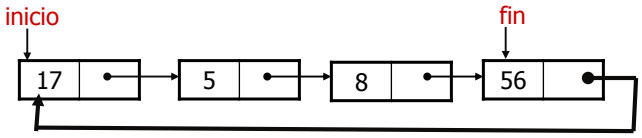
- Lista circular (o en anillo)
  - Es aquella en la que en lugar de almacenar un apuntador a null en el campo sgte del último nodo de la lista, se hace que el último elemento apunte al principio de la lista.
  - En una lista circular cada nodo de la lista es accesible desde cualquier otro nodo de ella. Es decir, dado un nodo se puede recorrer toda la lista completa. En una lista enlazada simple solo es posible recorrerla por completo si se parte del primer nodo.
  - Las operaciones de concatenación y división de listas son mas eficaces con listas circulares



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 28

ACTIVIDAD COLABORATIVA

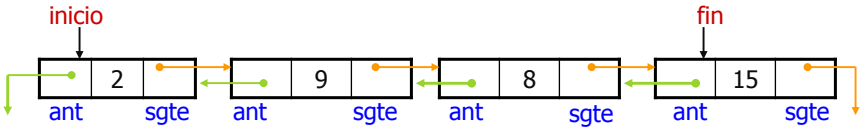
- En binas, insertar un elemento **al inicio** de la lista
- Insertar un elemento al final



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 29

TIPOS DE LISTAS LIGADAS

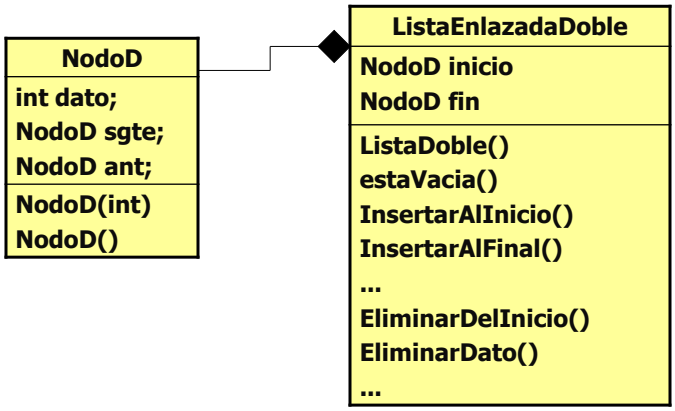
- **Lista doblemente enlazada**
  - En las listas anteriores el recorrido de ellas solo podía hacerse en un único sentido: de izquierda a derecha (principio a final). En numerosas ocasiones se requiere recorrer las listas en ambas direcciones
  - Las listas que pueden recorrerse en ambas direcciones se denominan listas doblemente enlazadas
  - En estas listas cada nodo consta del campo de información (dato) y de dos campos de enlace o apuntadores: anterior(ant) y siguiente (sgte), que apuntan hacia adelante y hacia atrás.



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 30

CLASE LISTA DOBLEMENTE ENLAZADA

- Dadas las operaciones de una lista doblemente enlazada, podemos modelar la clase listaDoble de la siguiente forma



31

CLASE LISTA DOBLEMENTE ENLAZADA

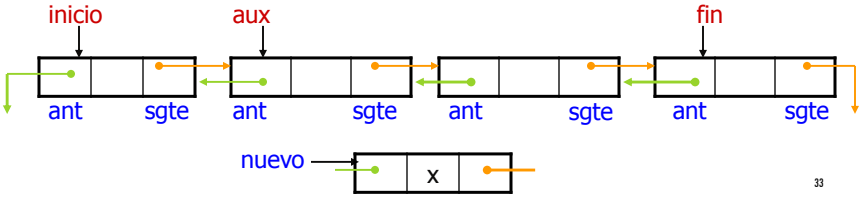
```
class NodoD {
 int dato;
 NodoD sgte;
 NodoD ant;
 //por convención sgte siempre es iniciado a null
 NodoD( ) { dato = 0; sgte = null; ant = null;}
 NodoD( int x) { dato = x; sgte = null; ant = null}
};

class ListaDoble {
 NodoD inicio, fin;  // inicio y fin de la cola
 ListaDoble( ) { inicio = null; fin = null;  }  //Lista Vacia
 bool estaVacia() { if (inicio == null && fin == null) return true;
                                return false; }
 void Insertar (int dato);
 void EliminarDato( int x);
 ....
};
```

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 32

INSERCIÓN

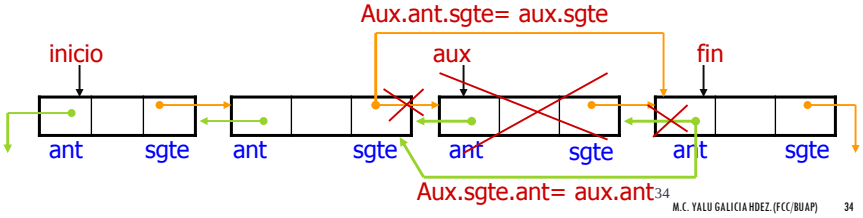
- La inserción de un nodo a la derecha de un nodo especificado, apuntado por aux, puede presentar varios casos
  - La lista esta vacía, entonces actualizar inicio y fin al nuevo nodo
  - Insertar dentro de la lista: existe un elemento anterior y otro posterior de X
  - Insertar al inicio de la lista. Se requiere que inicio sea modificado
  - Insertar a la derecha del nodo al final de la lista. Se requiere que fin sea modificado



33

ELIMINACIÓN

- La eliminación es directa
  - Si la lista tiene un simple nodo, entonces los apuntadores inicio y fin son puestos a NULL.
  - Si el nodo al final de la lista debe ser eliminado, el fin debe modificarse para apuntar al predecesor del nodo que se va a eliminar de la lista
  - Si el nodo al inicio de la lista debe ser eliminado, el inicio debe modificarse y apuntar al siguiente de la lista
  - La eliminación dentro de la lista del nodo aux es como sigue:



34

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 34

¿QUE HEMOS APRENDIDO?