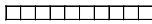


Clase25: Listas Enlazadas

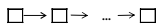
¿Estructuras de datos?

Formas de organización y almacenamiento de datos (en memoria). Ejemplos:

arreglo



lista enlazada



árbol



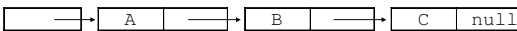
Arreglo versus Lista enlazada

Característica	Arreglo	Lista enlazada
Memoria elementos	contigua	separada
Elemento	valor	nodo (valor + enlace)
Máximo elementos	si	no
Acceso a elementos	índice (posición relativa)	referencia (dirección)
Tiempo de acceso	constante	variable

Aplicación. Representación de objetos de múltiples valores
Ejemplos: stack, queue, diccionario, conjunto, polinomio, etc

¿lista enlazada?

ejemplo:



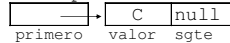
primero valor sgte valor sgte valor sgte

- n° indeterminado de objetos (nodos) distribuidos en la memoria
- cada nodo contiene un valor y un enlace (referencia) al sgte nodo
- referencia al primer nodo se mantiene en un lugar especial (primero/first, cabeza/head)
- último nodo contiene una referencia con el valor null

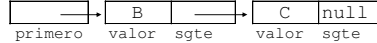
Construcción “artesanal” de lista enlazada

```
class Nodo{  
    public Object valor; //valor del nodo  
    public Nodo sgte; //ref(enlace) a sgte nodo  
    public Nodo(Object x, Nodo y){valor=x; sgte=y;  
}
```

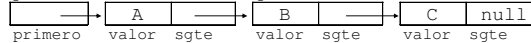
```
Nodo primero=new Nodo("C",null);
```



```
primero=new Nodo("B",primero);
```



```
primero=new Nodo("A",primero);
```



Algoritmo de construcción de lista enlazada

```
//crear lista vacía  
Nodo primero=null;  
//recorrer valores  
String[] a={"C","B","A"}; //ejemplo  
for(int i=0; i<a.length; ++i){  
    //crear nuevo nodo  
    Nodo r=new Nodo(a[i],null); //r -> |ai|null|  
    //enlazarle lista anterior  
    r.sgte=primero; //r -> |ai| -> |ai-1| -> . . .  
    //dejarlo como primero de la lista  
    primero=r; //1° -> |ai| -> |ai-1| -> . . .  
}  
abreviando  
Nodo primero=null;  
for(int i=0; i<a.length; ++i)  
    primero=new Nodo(a[i],primero); //1° -> |ai| -> ...
```

Algoritmo de recorrido de lista

//posicionarse en (apuntar a) primer nodo

Nodo r=primero;

//repetir hasta fin de lista

while(r!=null){

//escribir valor del nodo

U.println(r.valor);

//apuntar a nodo siguiente

r=r.sgte;

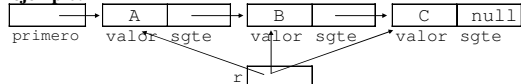
}

Abreviando

for(**Nodo** r=primero; r!=null; r=r.sgte)

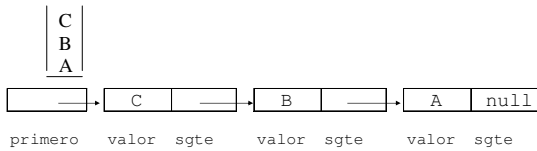
U.println(r.valor);

ejemplo:



Clase25: Listas Enlazadas

Stack: representación con lista enlazada



```
class Nodo
{
    public Object valor; //visible
    public Nodo sgte;    //visible
    public Nodo(Object x,Nodo y){
        valor=x; sgte=y;
    }
}
```

```
class Stack
{
    //representación: ref a primer nodo
    protected Nodo primero;

    public Stack(){
        primero=null;
    }
    public boolean empty(){
        return primero == null;
    }
    public boolean full(){
        return false; //nunca está lleno
    }
    public void reset(){
        primero=null;
    }
}
```

```
public void push(Object x)
throws StackFull
{
    //crear un nodo con valor x y sgte nulo
    Nodo r=new Nodo(x,null);//r→| x |null|

    //enlazarle lista anterior
    r.sgte=primero;//r→| x | → | y | → ...

    //dejarlo como primero de la lista
    primero=r;//1°→| x | → | y | → ...
}
```

Alternativamente:

```
public void push(Object x)
throws StackFull
{
    //agregar al comienzo
    primero=new Nodo(x, primero);

    //1°→| x | → | y | → ...
}
```

```
public Object pop()
throws StackEmpty
{
    //excepción si lista está vacía
    if(primero==null) throw new StackEmpty();

    //recuperar primer valor
    Object aux=primero.valor;//1°→| x | → | y |

    //eliminar primer nodo
    primero=primero.sgte; //1°→| y | →...

    //devolver primer valor
    return aux;
}
} //fin class Stack
```

Alternativamente

```
public Object pop()throws StackEmpty{
    try{
        Object aux=primero.valor;//excepcion si null
        primero=primero.sgte;
        return aux;
    }
    catch(NullPointerException x)
    {
        throw new StackEmpty();
    }
}
} //fin class Stack
```

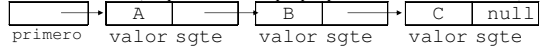
Clase25: Listas Enlazadas

Problema: implementar la clase Queue.

Operación	significado
new Queue()	_____
void enqueue(Object x) throws QueueFull	_____ ← x
Object dequeue() throws QueueEmpty	← _____
void reset()	_____
boolean empty()	¿ _____ ?
boolean full()	¿ _____ ?

Representación de una Queue con una lista enlazada

Alternativa 1 : (enqueue toma tiempo proporcional a N° de nodos)



```

class Queue
{
protected Nodo primero;

public Queue() {primero=null;}

public void reset() {primero=null;}

public boolean empty() {return primero==null;}

public boolean full() {return false;}

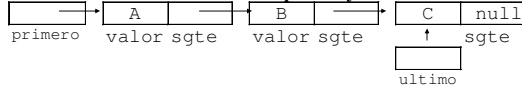
```

```

public Object dequeue() throws QueueEmpty{
    if(primero==null) throw new QueueEmpty();
    Object aux=primero.valor;
    primero=primero.sgte;
    return aux;
}
public void enqueue(Object x) throws QueueFull{
    //crear nuevo nodo
    Nodo r=new Nodo(x,null);
    //si lista vacía, agregar al comienzo
    if(primero==null){primero=r; return;}
    //agregar al final
    Nodo ultimo=primero;
    while(ultimo.sgte!=null) ultimo=ultimo.sgte;
    ultimo.sgte = r;
}
}
Nota. operación O(n)

```

Alternativa 2: con referencias al primer y último nodo



```

class Queue{
protected Nodo primero, ultimo;
public Queue(){
    reset();
}
public void reset(){
    primero=ultimo=null;
}
public boolean empty(){
    return primero==null;
}
public boolean full(){
    return false;
}
}

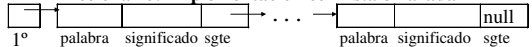
```

```

public void enqueue(Object x) throws QueueFull
{
    Nodo r=new Nodo(x, null);
    if(ultimo==null) //si lista vacía
        ultimo=primero=r; // dejar con un nodo
    else //si no
        ultimo=ultimo.sgte=r; //enlazar al final
}
public Object dequeue() throws QueueEmpty
{
    if(primero==null) throw new QueueEmpty();
    if(primero==ultimo) ultimo=null; //un nodo
    Object aux=primero.valor;
    primero=primero.sgte;
    return aux;
}
}
Nota. Operaciones enqueue y dequeue O(1)

```

TDA Diccionario: Implementación con lista enlazada



```

class Nodo{
public Comparable palabra;
public Object significado;
public Nodo sgte;
public Nodo(Comparable x, Object y, Nodo z){
    palabra=x; significado=y; sgte=z;
}
}
class Diccionario{
protected Nodo primero;
public Diccionario(){primero=null; }
public Object buscar(Comparable x){
    Nodo r=referencia(x,primero); //propuesto
    return r==null ? null : r.significado;
}
}
...

```