

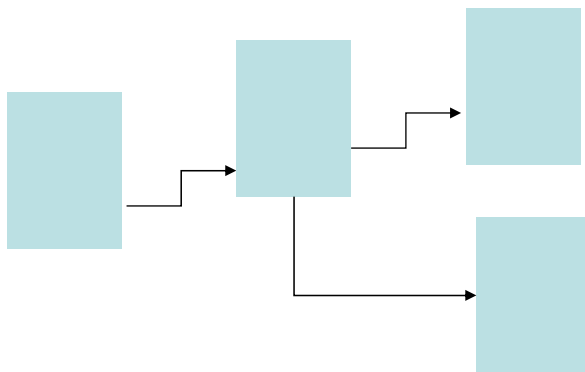
2.0.2. Egitura estekatuak

[Creative Commons](#) lizentziapean



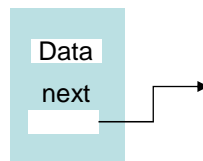
Egitura estekatu baten ezaugarriak

- Datu-egiturak dira, objektuen erreferentzia-aldagaiak erabiltzen dituzte beste objektuen estekak izan ahal izateko



Egitura estekatu baten diseinua

- Orokorrean, **adabegien klase bat** egongo da. Adabegi batek hau izango du:
 - Datuak eta
 - Adabegi klasearen erreferentzia bat edo gehiago (definizio errekurtsiboa)



Lehen hurbilpena(I)

```

public class Pertsona {
    private String name;
    private String na;
    private Pertsona next; // hurrengo pertsonaren atzipena!

    public Pertsona(String pName, String pNa) { // Eraikitzailea
        name = pName;
        na= pNa;
        next = null;
    }
    public void setNext(Pertsona next) { this.next = next; }

    public void print() { // Dena idazten du
}
  
```

Lehen hurbilpena (II)

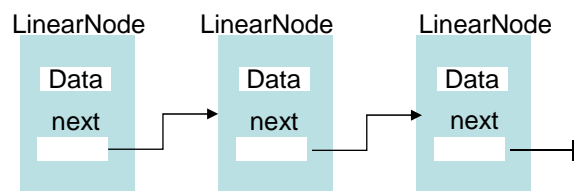
```
public static void main(String[] args) {  
  
    Pertsona p1 = new Pertsona("pepe", "1111");  
    Pertsona p2 = new Pertsona("ana", "2222");  
    Pertsona p3 = new Pertsona("jon", "3333");  
    Pertsona p4 = new Pertsona("amaia", "1212");  
  
    p1.setNext(p2);  
    p2.setNext(p3);  
    p3.setNext(p4);  
}
```

Lehen hurbilpena(III)

- Pertsona klasea,
 - Pertsona bat da?
 - Edo pertsona-multzo bat?
 - Zer idazten du print-ek? Zer idatzi beharko luke?
- Aurreko inplementazioa badabil, baina diseinu ez egokia du
 - Pertsona klasea eta PertsonenZerrenda klaseak desberdinak dira

Adabegi baten definizioa

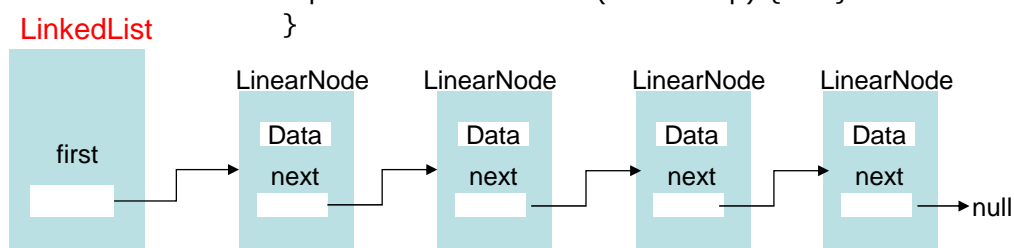
```
public class LinearNode
{
    public Pertsona data;    // datuak adabegian
    public LinearNode next; // hurrengo adabegia
}
```



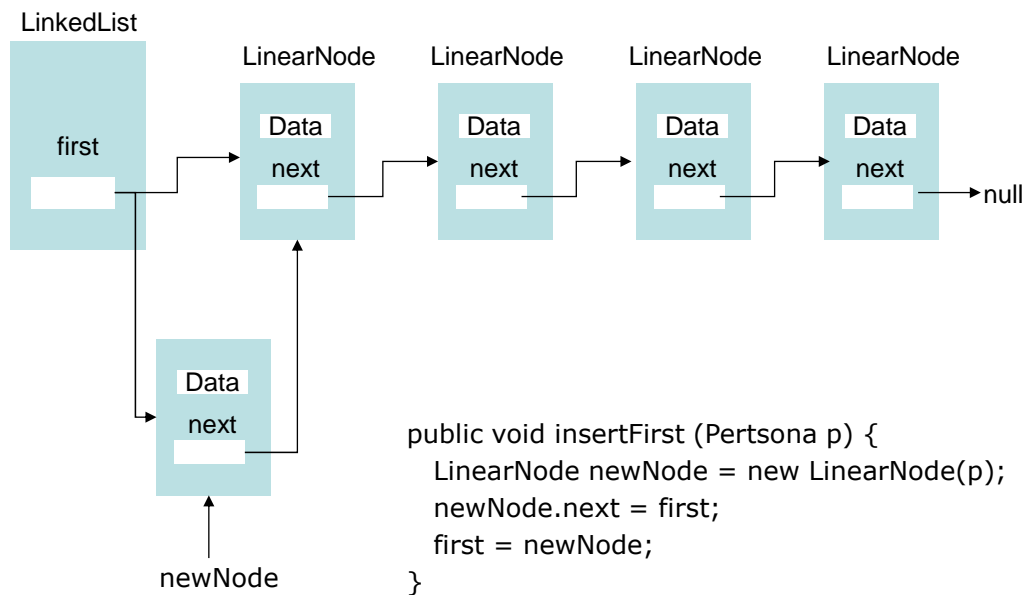
Egitura estekatuaren definizioa

```
class LinkedList
{
    private LinearNode first; // adabegi berezi baten erreferentzia
    // aukerazko atributuak: int size, ...

    public LinkedList() {          // eraikitzailea
        first = null;
    }
    public boolean isEmpty() {.....}
    public void insertFirst(Pertsona p) {.....}
    public Pertsona deleteFirst() {.....}
    public void displayList() {.....}
    public Pertsona find(Pertsona p) {.....}
    public Pertsona delete(Pertsona p) {.....}
}
```

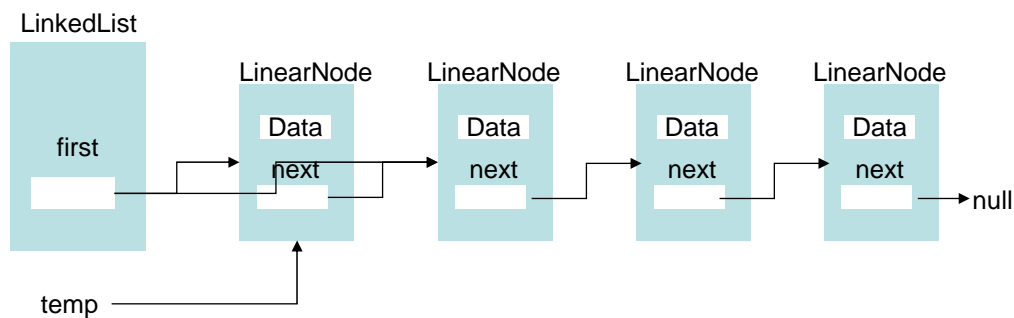


Txertaketa hasieran



Kostua: $O(1)$

Lehenengoa ezabatu



```

public Pertsona deleteFirst ( ) {
    Pertsona temp = first.data;
    first = first.next;
    return temp;
}

```

Kostua: $O(1)$

Egiturako datuak inprimatu

```
public void displayList()
{
    System.out.print("List (first-->last): ");
    LinearNode current = first; // start at beginning of list
    while(current != null)      // until end of list,
    {
        current.displayLink(); // print data
        current = current.next; // move to next node
    }
    System.out.println("");
}
```

Kostua: $O(n)$

n : egiturako adabegi-kopurua

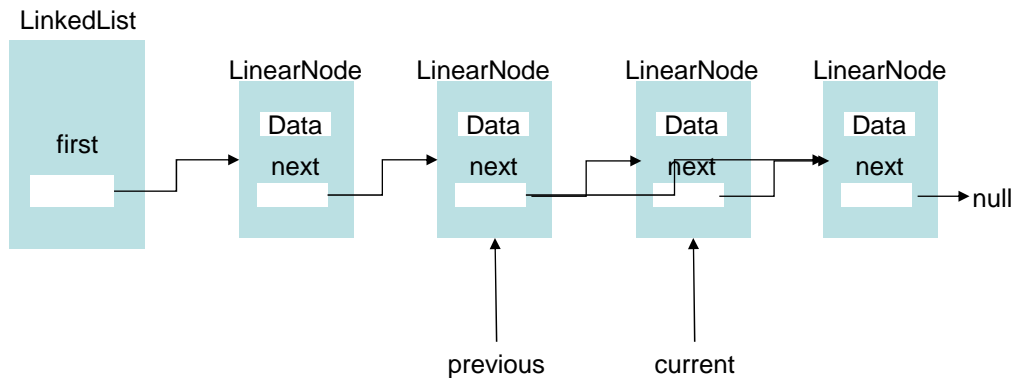
- Ikusi Link.java, LinkList.java eta LinkListApp.java

Bilatu balio jakin bateko adabegia

```
public Pertsona find(Pertsona p) // find node with given key
{
    // (assumes non-empty list)
    LinearNode current = first; // start at 'first'
    while(!current.data.equals(p)) // while no match,
    {
        if(current.next == null) // if end of list,
            return null;        // didn't find it
        else // not end of list,
            current = current.next; // go to next link
    }
    return current.data; // found it
}
```

Kostua: $O(n)$

Ezabatu balio jakin bateko adabegia



Ezabatu balio jakin bateko adabegia

```

public Pertsona delete(Pertsona p) // delete link with given key
{
    // Precondition: there exists an element with the given key
    // (assumes non-empty list)

    LinearNode current = first; // search for link
    LinearNode previous = first;

    while (!current.data.equals(p)) {
        if (current.next == null)
            return null; // didn't find it
        else
        {
            previous = current; // go to next link
            current = current.next;
        }
    }

    // found it
    if (current == first) // if first link,
        first = first.next; // change first
    else // otherwise,
        previous.next = current.next; // bypass it
    return current.data;
}
  
```

Kostua: $O(n)$

Iteradore bat eskaintzeko?

```

/** Return an iterator to the stack that iterates through the items . */
public Iterator iterator() { return new ListIterator(); }

// an iterator, doesn't implement remove() since it's optional
private class ListIterator implements Iterator {

    private Link current = first;

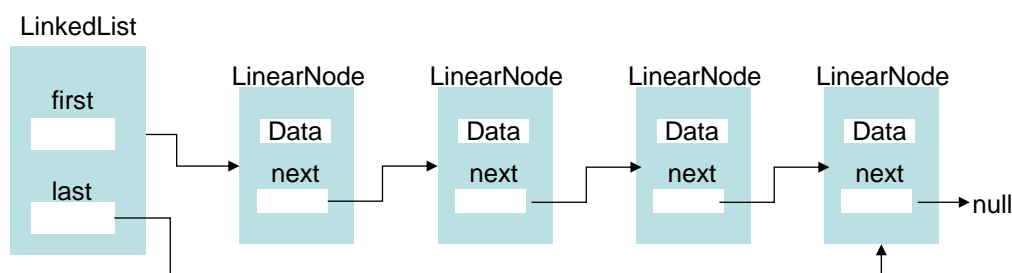
    public boolean hasNext() { return current != null; }

    public void remove() {
        throw new UnsupportedOperationException(); }

    public Pertsona next() { // recorre el campo clave
        if (!hasNext()) throw new NoSuchElementException();
        Pertsona item = current.iData;
        current = current.next;
        return item;
    }
} // private class

```

Bukaeran txertatzeko, komeni da azkenaren atzipena izatea



- Ikusi FirstLastList.java
- 4. kapitulua aztertu: Linked structures [Lewis eta Chase 2010]