DATU-EGITURAK ETA ALGORITMOAK I **2. maila**
2006ko otsailaren 6a
Epazpena

1. ariketa: (3 puntu)

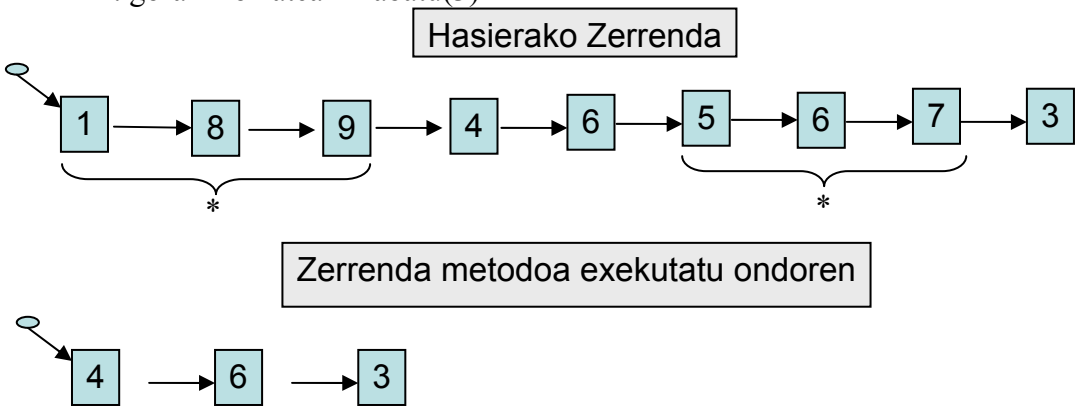
Elementu osoko esteka sinpleko zerrenda batean, diseina eta inplementatu hurrengo metodoa LinkedListItr klasean:

void **goranzkoKateakEzabatu**(int luz)

Metodoak *luz* luzeerako goranzko elementu sekuentziak ezabatzen ditu.

Adibidea:

z. goranzkoKateakEzabatu(3)



Diseinua

```
Hasierako nodoan kokatu
hasierakoJarraian=zerrendaren hasierako nodoa
while (elementuak zerrendan)
 jarraian=0
 while (uneko!=null eta uneko.hurrengo!=null eta uneko.balioa<uneko.hurrengo.balioa)
    jarraian=jarraian+1
    hurrengora joan
 if (jarraian==luzeera)
   if (hasierakoJarraian==zerrendaren hasierako nodoa)
     zerrendaren hasiera=unekoa.hurrengoa
     hasieraJarraian= zerrendaren hasiera
    } else
       ezabatu jarraizko sekuentzia (hasieraJarraian.next=unekoa.next)
   else // jarraian!=luzeera
      hasieraJarraian=unekoa
 zerrendaren hurrengora joan
 } //endWhile
}//end while
```

Inplementazioa

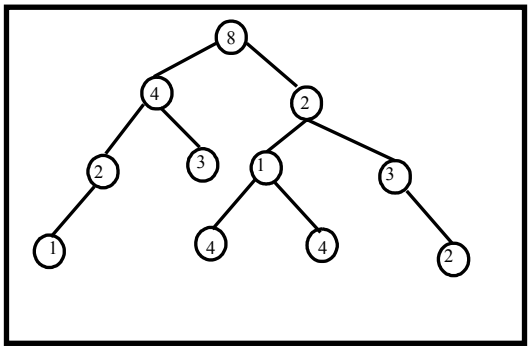
```
public void goranzkoKateakEzabatu(int luz){
  this.goFirst();
  Node<T> hasiera=list.top;
  int jarraian=0;
  while (actual!=null){
     jarraian=1;
    while ((actual!=null) && (actual.next!=null) &&
          ((actual.getElem()).intValue()<(actual.next.getElem()).intValue())) {
      jarraian++;
      this.goNext();
    }
  if (jarraian==luz)
    if (hasiera==top) {
      top=actual.next;
      hasiera=top;
    }
    else
     hasiera.next=actual.next;
   else
     hasiera=actual;

   this.goNext();
  }
  }
  }
```

2. ariketa: (3 puntu)

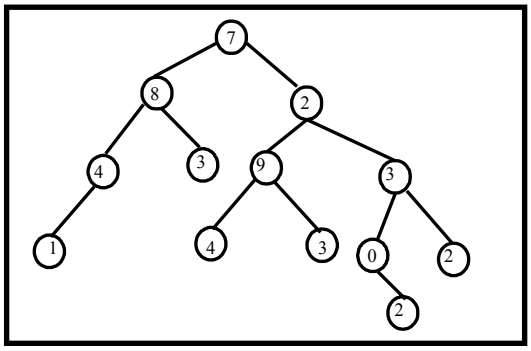
Osoko zenbakien zuhaitz bitar ez-huts bat emanda, zuhaitz hori **bideratuta** dagoen ala ez erabakitzen duen metodo bat espezifikatu, diseinatu eta inplementatu.

Zuhaitz bat **bideratuta** dago baldin eta soilik baldin errotik hostoetara doazen bere bide guztiek pisu berdina badute. Bide baten pisua bere adabegi guztien pisuen batura da. Zuhaitz hutsa bideratuta dago.

Zuhaitz bideratua



Zuhaitz EZ bideratua



Diseinua

Parametrizazioa: 2 irteera parametroak, pisua eta bideratua den ala ez

Kasu nabariak:

Hostoa bada **itzuli** pisu hostoa, True

Kasu orokorrak

```
If (erroa.ezk!=null) && (erroa.esk!=null)
        (pisuEzk, bidEzk)=bideratuta(erroa.ezk)
        (pisuEsk, bidEsk)= bideratuta(erroa.esk)
        If (pisuEsk > pisuEzk)
           max= pisuEsk
         else
           max= pisuEzk
        isBideratua= bidEzk && bidEsk && pisuEzk == pisuEsk
        itzuli max+erroa.pisua, isBideratua

If (erroa.ezk==null) && (erroa.esk!=null)
        (pisuEsk, bidEsk)= bideratuta(erroa.esk)
        itzuli pisuEsk+erroa.pisua, bidEsk

If (erroa.ezk!=null) && (erroa.esk==null)
        (pisuEzk, bidEzk)= bideratuta (erroa.ezk)
        itzuli pisuEzk +erroa.pisua, bidEzk
```

Inplementazioa

```
class Data{
  int pisua;
  boolean bideratua;
  Data(int p, boolean b){
        pisua=p;
        bderatua=b;
   }
}


class Bintree{
    public boolean isBideratua(){
       Data d=isBideratua(root);
       return d.bideratua;
    }
```

```
    public Data isBideratua(BTNode<Integer> pNode){
        int nodeBalioa=(pNode.content).intValue();
        if(pNode.left==null) && (pNode.right==null) //hostoa
          return new Data(balioa, true);

        if(pNode.left!=null) && (pNode.right!=null) {
          Data dEzk=isBideratua(pNode.left);
          Data dEsk=isBideratua(pNode.right);
          int max;
          if (  dEzk.pisua > dEsk.pisua)
             max= dEzk.pisua;
           else
             max= dEsk.pisua
          boolean bideratua=dEzk.bideratua && isEzk.bideratua &&
                                                dEzk.pisua==dEsk.pisua
          return new Data(max+nodeBalioa, bideratua);
        }
        if(pNode.left==null) && (pNode.right!=null) {
          Data dEsk=isBideratua(pNode.right);
          return new Data(dEsk.pisua+max, dEsk.bideratua)
        }

        if(pNode.left!=null) && (pNode.right==null) {
          Data dEzk=isBideratua(pNode.left);
          return new Data(dEzk.pisua+max, dEzk.bideratua)
        }
    }


}
```