Ariketa guztien ebazpenerako Zerrenda DMA inplementazioa daukagu LinkedList klasearen bitartez. Jarraian beren metodoak aurkezten dira:

Klase LinkedList<T>

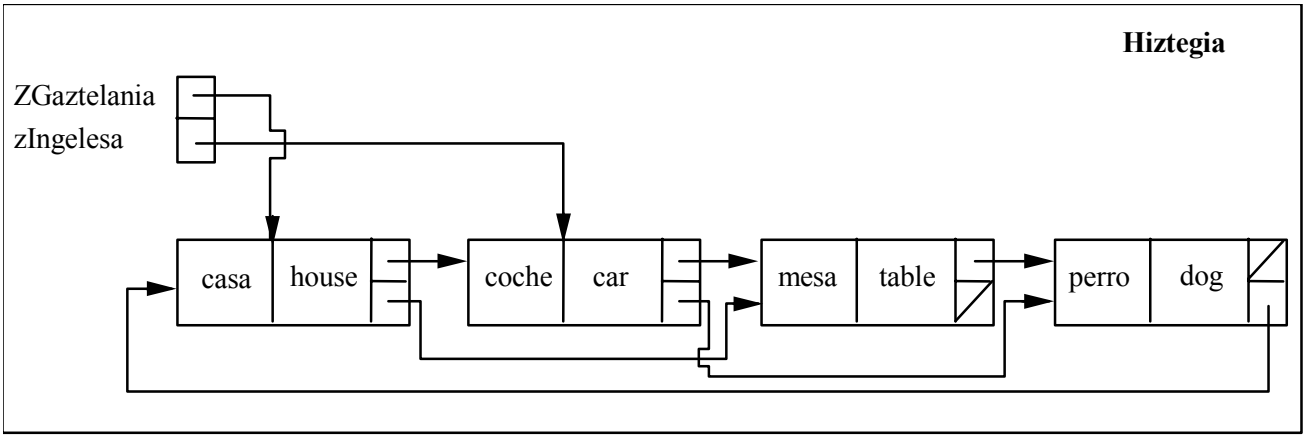
********** METODO PUBLIKOAK **********

boolean isEmpty( ) --> True zerrenda hutsa bada
void emptyList ( ) --> Zerrenda husten du
void insert(T x) → x txertatu actual posizioaren jarraian
void remove(T x) → ezabatu x element. lehendabiziko agerpena
void removeActual() → ezabatu actual elemetua
boolean find(T x) → true x existitzen bada, eta actual x elemt. posizioan
void goFirst() → actual lehendabiziko posizian kokatu
void goNext() → actual hurrengo posioan kokatu
boolean hasMoreElements() → actual!=null
T getActual() → itzuli actual elementuaren edukia

(J94) Gaztelania-ingelesa hiztegi elebidun bat errepresentatzen duen klase bat daukagu. Klase honek**, bi lengoaietatik hitz ordenatuen bi zerrenda mantentzen ditu**

Gaztelerazko hitz bakoitza (eta ingelesekoa hurrenez hurren) aldi bakar bat agertu daiteke hiztegian, eta ondorioz, hitz bakoitzak ezin dezake bi itzulpen desberdin eduki.

Hurrengo adibidean ikusi daiteke, gaztelerazko zerrendan *casa, coche*, *mesa* eta *perro* (alfabetikoki gaztelaniaz ordenatuta) hitzak daudela eta ingeleseko zerrendan *car, dog*, *house* eta azkenik *table* (alfabetikoki ingelesez ordenatua) hitzak.



Eskatzen da:

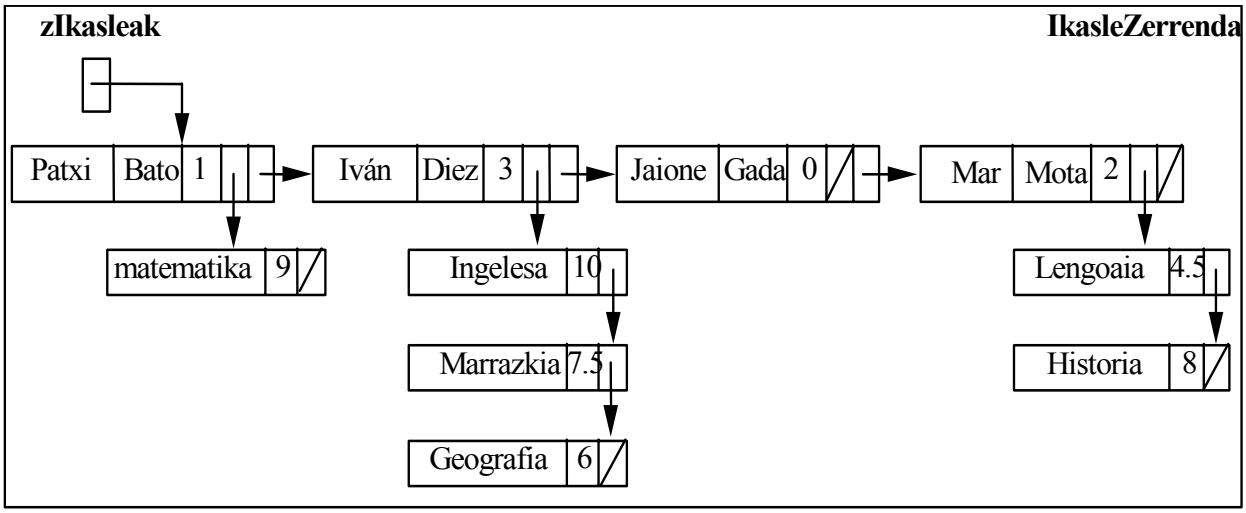
a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) ***Hiztegi* klasearen *itzuli* metodoaren diseinu eta inplementazioa.** Metodo honek, hitz bat eta bere lengoaia (gaztelaniaz true, ingelesez false) jasoko du parametro bezala, eta emaitza bezala hitz honen itzulpena beste lengoaiara itzuliko du. Hitza agertzen ez bada, null itzuliko da.

```
public String itzuli(String p, boolean b)
```

(S94) Ikasleen zerrenda bat, aurkeztutako asignaturen kalifikazioekin errepresentatzen duen IkasleZerrenda klase bat daukagu. Ikasleen zerrenda, ikasleen lehenengo abizenengatik ordenatuta dago, baina irakasgaik ordenarik gabe kokatu dira.

Hurrengo irudian aurkezten da, ListaAlumnos zerrendaren adibide bat, abizenengatik soilik ordenatuta.



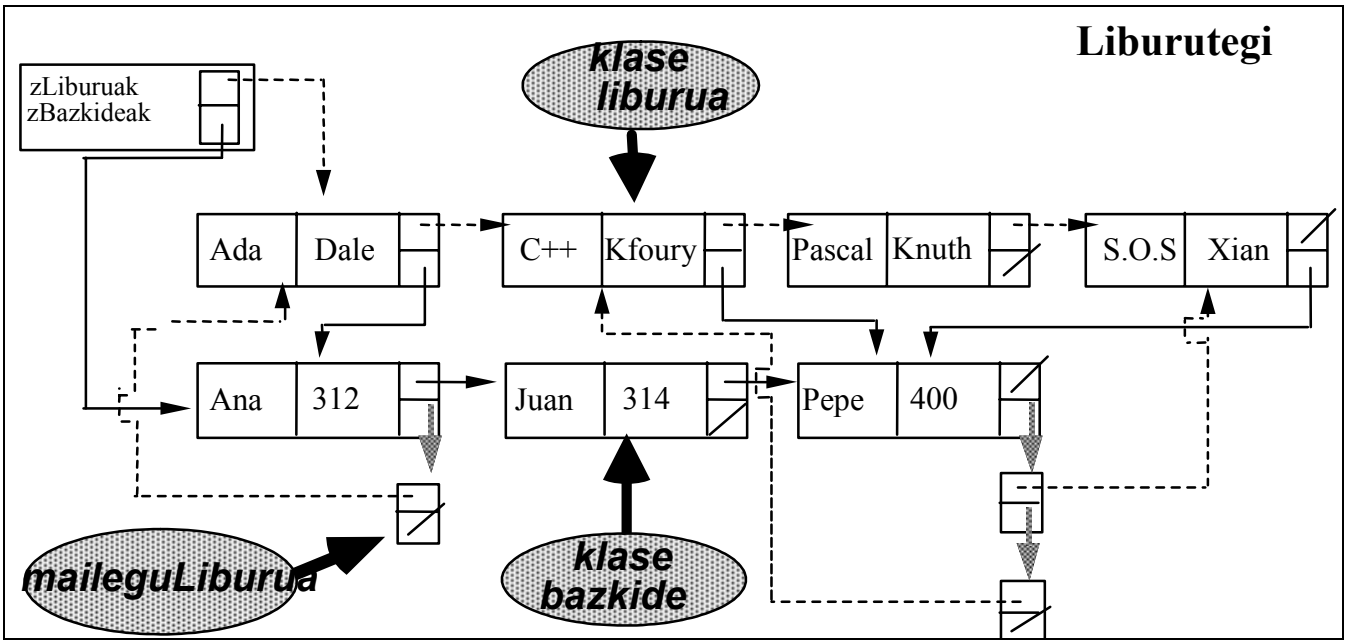
Eskatzen da:

a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) IkasleZerrenda klasearen **gaindituakInprimatu** metodoaren diseinua eta inplementazioa Java-n. Metodo honek, irakasgai bat jasotzen du parametro bezala, eta irakasgai hori gainditutako ikasleen izena eta abizena inprimatzen ditu.

```
public class IkasleZerrenda extends Object {
 public void gaindituakInprimatu (String asign) {
 ..............
 ..............
 }
}
```

(J95) Liburutegi bat errepresentazen duen klase bat daukagu. Klase honek, bi zerrenda dauzka: liburuen zerrenda eta bazkideen zerrenda. Bi zerrendak ordenatuta daude, lehenengoa alfabetikoki liburuaren izenburuagatik, eta bigarrena bazkideen karneta zenbakiagatik.
Hurrengo irudian, liburutegi bat aurkezten da 4 liburuekin eta 3 bazkideekin.



Liburu zerrendako elementu guztiak, izenburua, idazlea eta erakusle bat maileguan duen bazkideari dauka. (liburua, liburutegian badago erakusle hori null balioa edukiko du). Bazkide zerrrendako elementu guztiak, izena, karneta zenbakia eta bazkideen mailegu zerrenda (liburutegitik atera dituen liburuak) dauka. Mailegu zerrendako elementu bakoitza, erakusle bat dauka maileguan daukan liburuari.

Eskatzen da:

a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) Liburutegi klasearen *mailegu* metodoaren diseinua eta inplementazioa Java-n. Metodo honek, bazkide baten zenbakia eta liburu baten izenburua jasoko ditu parametro bezala, eta hurrengo portaera izengo du:

- Liburua liburutegian ez badago, inprimatu errorezko mezu bat
- Liburua maileguan badago, inprimatu liburu hori maileguan daukan bazkidearen izena.
- Liburua liburutegian badago, liburutegia eguneratu eskatzen den maileguarekin.

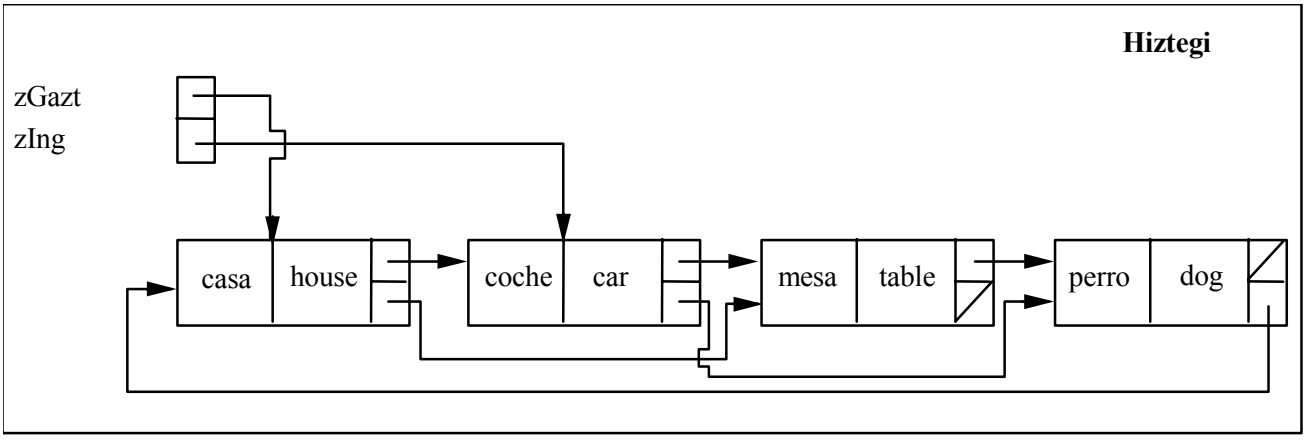
Suposatzen da, bazkidearen zenbakia existitzen dela liburutegian. Hau da, ez da konprobatu behar bazkidearen existentzia.

```
public  void mailegu (int pBaz, String pLib)
```

(S95) Gaztelania-ingelesa hiztegi elebidun bat errepresentatzen duen klase bat daukagu. Klase honek**, bi lengoaietatik hitz ordenatuen bi zerrenda mantentzen ditu**

Gaztelerazko hitz bakoitza (eta ingelesekoa hurrenez hurren) aldi bakar bat agertu daiteke hiztegian, eta ondorioz, hitz bakoitzak ezin dezake bi itzulpen desberdin eduki.

Hurrengo adibidean ikusi daiteke, gaztelerazko zerrendan *casa, coche*, *mesa* eta *perro* (alfabetikoki gaztelaniaz ordenatuta) hitzak daudela eta ingeleseko zerrendan *car, dog*, *house* eta azkenik *table* (alfabetikoki ingelesez ordenatua) hitzak.

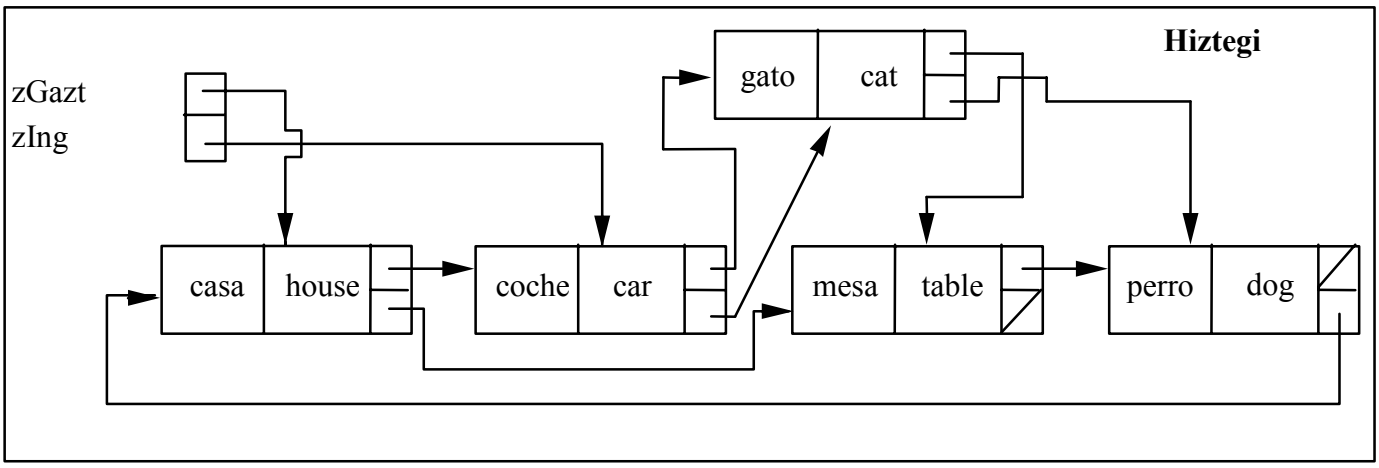


Eskatzen da:

a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) Hiztegi klasearen *bikoteTxertatu* *metodoaren diseinua eta inplementazioa Java-n.* **Metodo honek, gaztelaniazko eta ingelesezko hitz bikote bat jasoko du parametro bezala. Hitzak agertzen ez badira (inongo biak), hiztegian sartuko ditu alfabetiko ordena mantenduz. Baten bat agertzen bada, bai ingelesezkoa bai gaztelaniazko, errore bat inprimatuko du.**

Hurrengo irudian *gato/cat* bikotea txertatu ondoren, hiztegia nola geratuko litzateken aurkezten da.



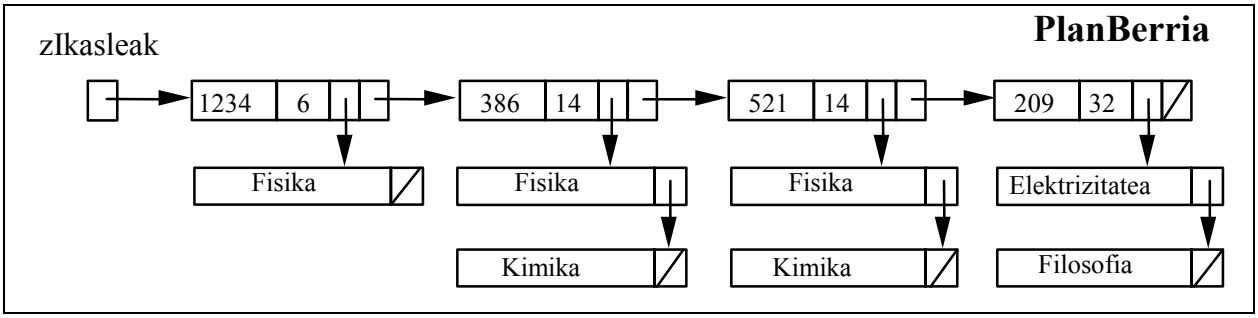
Metodoaren definizioa hurrengoa da:

```
public void bikoteTxertatu(String p1, String p2)
```

(J96) Plan Berriko ikasleei buruzko informazioa errepresentatzen duen klase bat ematen digute

Ikasle bakoitzak, *espediente zenbakia*, gainditu dituen *kreditu kopurua* eta gainditu dituen *irakasgai zerrenda* bat gordetzen du. Ikasleak *kreditu kopurutik* ordenatuta daude, eta kreditu berdinen artean *espediente zenbakitik*

Adibidea:



Eskatzen da:

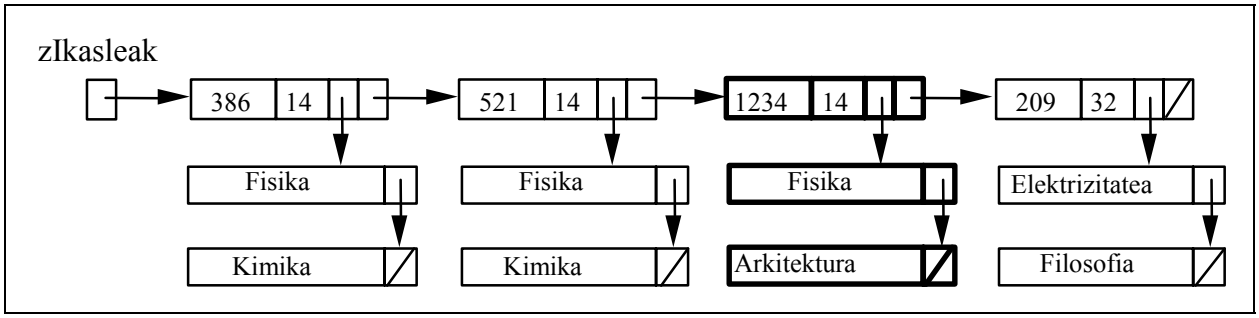
a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

***b)* PlanBerria klaseko *irakasgaiGainditu* metodoa**

```
 public  void irakasgaiGainditu(int pzEsp, String
pIrak, int pKred)
```

- Metodoak *espediente zenbakia*, *irakasgai* bat eta *irakasgaiaren kreditu kopurua* jasotzen du parametro bezala.
- Zerrendaren eguneraketa mantendu behar du ordenaziozko baldintzak (ikasleak, kredito zenbakiaz ordenatuta, eta kreditu berdinentzat espediente zenbakiagatik).
- Ikaslearen *espediente zenbakia* agertzen ez bada, ikasle objektu berri bat sortu beharko da eta txertatu zerrendan.
- Ikaslea badago, baina *kreditu kopuru maximoa* gainditu badu (300), errore bat inprimatuko da.
- Errore bat inprimatuko da, ikasle batek *lehendik gainditutako* irakasgai bat, berriz gainditu nahi badu.

Hurrengo irudian aurkezten da ikasleen zerrenda **`irakasgaiGainditu (1234,"Arkitektura", 8)` exekutatu ondoren**



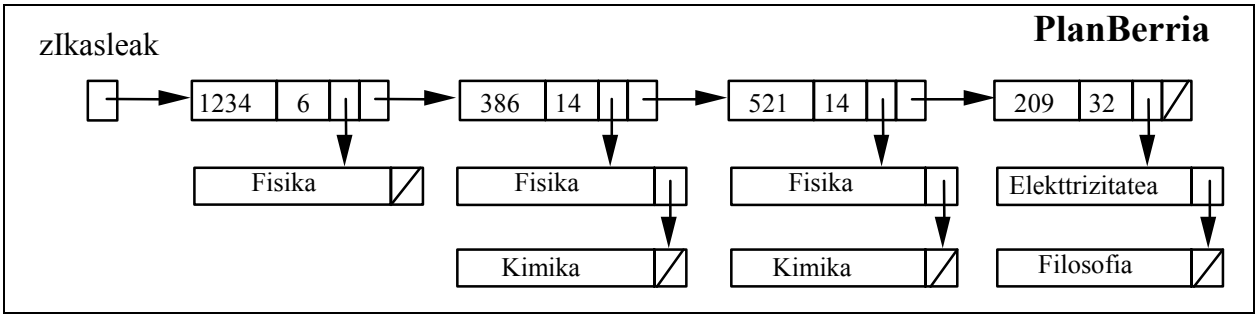
Suposatzen dugu PlanBerria klasearen ikasleKokatu() metodoa eskuragarri dagoela.

```
public class PlanBerria {
   public void ikasleaKokatu(Ikale pIk)
-- Pre    “pIk” Ikaslea klaseko objektu bateri
erreferentziatzen dio
-- Pos: “pIk” erreferentziatuko ikaslea, zerrendan
txertatu da, ordenaziozko legeak mantenduz.
```

(S96) Plan Berriko ikasleei buruzko informazioa errepresentatzen duen klase bat ematen digute

Ikasle bakoitzak, *espediente zenbakia*, gainditu dituen *kreditu kopurua* eta gainditu dituen *irakasgai zerrenda* bat gordetzen du. Ikasleak *kreditu kopurutik* ordenatuta daude, eta kreditu berdinen artean *espediente zenbakitik*

Adibidea:



Eskatzen da:

a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) PlanBerria klaseko *ikasleKokatu* metodoa

```
public void ikasleKokatu (Ikasle ikBerria, Ikasle ikZaharra)
```

Espezifikazioa:

SARRERA:

- *ikBerria*: txertatu nahi den ikasle objektua

- *ikZaharra*: PlanBerria zerrendaren ikasle objektua, ikBerria ikaslearen espediente zenbaki berberarekin.
  Zerrendan ez badago inongo ikaslerik espediente zenbaki horrekin, orduan ikZaharraren balioa null izango da.

IRTERA

ikZaharra null bada, orduan *ikBerria* PlanBerria zerrendan txertatu da hurrengo ordenean:

goranzko ordenean **kreditu kopuruagatik**, eta kreditu berdin dutenen artean, **espediente zenbakiagatik** goranzko ordenean.

Ordea, *ikZaharra* **null ez bada**, objektua zerrendan zegoelako izango da. Orduan, bi kasu aztertu behar dira:

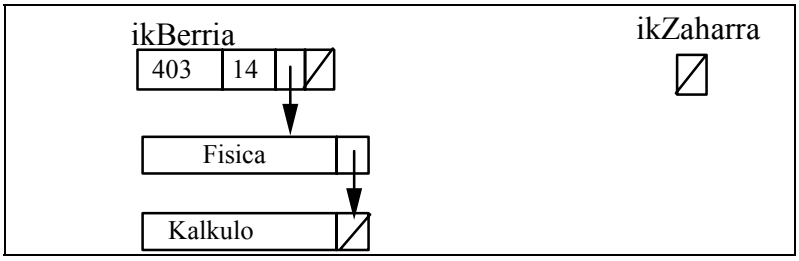
• Zerrendan zegoen informazioa eta txertatu nahi duguna berdinak badira (gainditutako irakasgaik orden berberan egin behar dira):

ez da ezer egiten

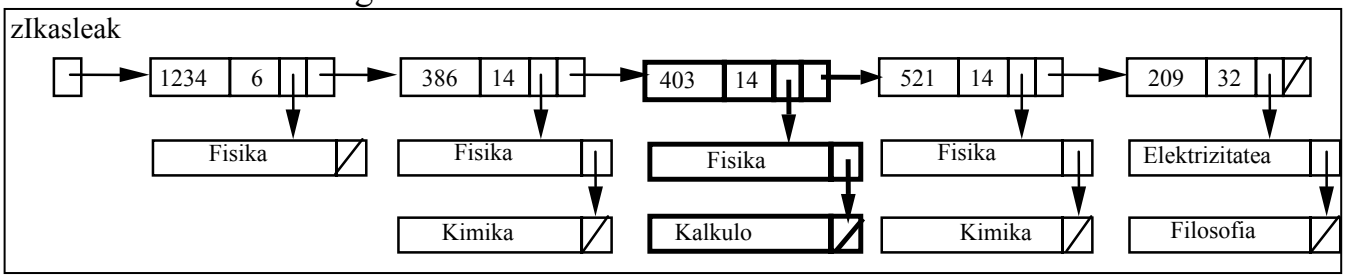
• Zerrendan zegoen informazioa eta txertatu nahi duguna desberdinak badira:

errore bat inprimatzen da

Adibidez, hurrengo *ikBerria objektua txertatuko balitz* zerrendan, *ikZaharra*=null balioarekin:

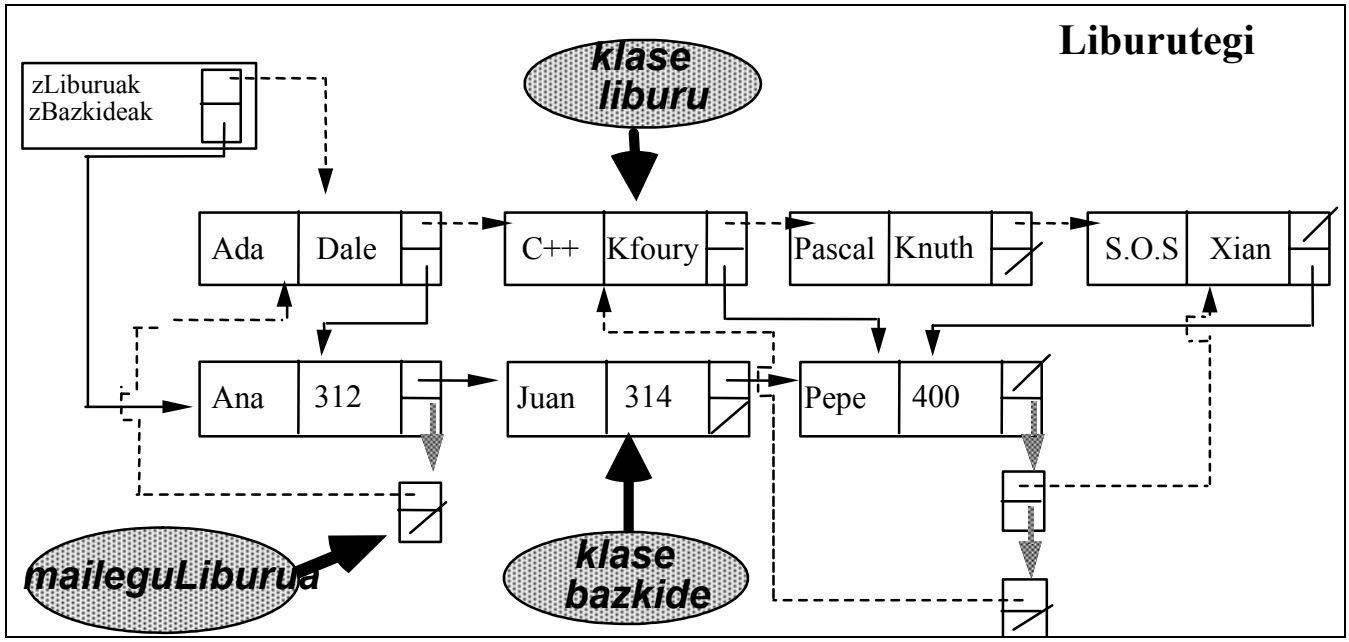


Ikasle zerrenda horrela geratuko litzateke:



(J97) Liburutegi bat errepresentazen duen klase bat daukagu. Klase honek, bi zerrenda dauzka: liburuen zerrenda eta bazkideen zerrenda. Bi zerrendak ordenatuta daude, lehenengoa alfabetikoki liburuaren izenburuagatik, eta bigarrena bazkideen karneta zenbakiagatik.

Hurrengo irudian, liburutegi bat aurkezten da 4 liburuekin eta 3 bazkideekin.



Liburu zerrendako elementu guztiak, izenburua, idazlea eta erakusle bat maileguan duen bazkideari dauzka. (liburua, liburutegian badago erakusle hori null balioa edukiko du). Bazkide zerrrendako elementu guztiak, izena, karneta zenbakia eta bazkideen mailegu zerrenda (liburutegitik atera dituen liburuak) dauzka. Mailegu zerrendako elementu bakoitza, erakusle bat dauka maileguan daukan liburuari.

Eskatzen da:

a) Behar diren klaseak, definitutako espezifikazioa gordetzeko.

b) Liburutegi klaseko *bazkideEzabatu* metodoaren diseinua eta inplementazioa Java-n. Metodoak, bazkide baten karneta zenbakia jasoko du parametro bezala eta:

- Bazkidea, bazkide zerrendan ez bada agertzen, errore bat inprimatuko da.
- Bazkidea badago, lehendabizi maileguan zeuzkan liburuak itzultzen ditu (librutegian egongo balitz bezala ipintzen ditu) eta jarraian bazkidea ezabatzen du bazkideen zerrendatik.

```
public void bazkideEzabatu(Bazkide pBaz)
```

(S97) Lehenengo orriaren *LinkedList* klasea emada, `errepikatuakEzabatu` metodoaren diseinua eta inplementazioa eskatzen da. Metodo honen portaera hurrengoa da:

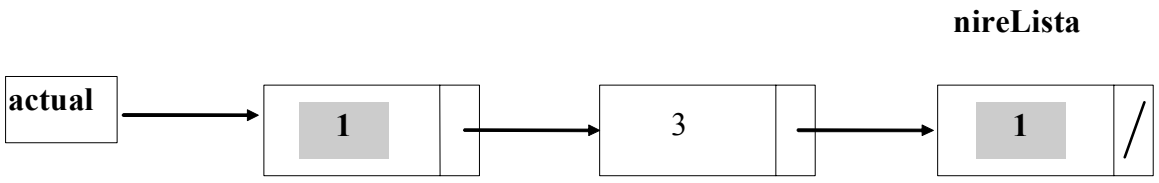
- Elementua zerrendan agertzen ez bada, errorea.
- Elementua bat baino gehiagotan agertzen bada, lehenengo agerpena utziko du soilik. (Beste guztiak ezabatuz)
- Elementu bakar bat agertzen bada, ez du ezer egiten.

```
public void errepikatuakEzabatu(T elem)
```

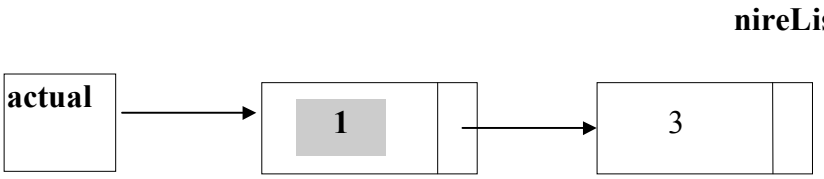
Adibidea:

LinkedList klasearen nireLista instantzia bat emanda:

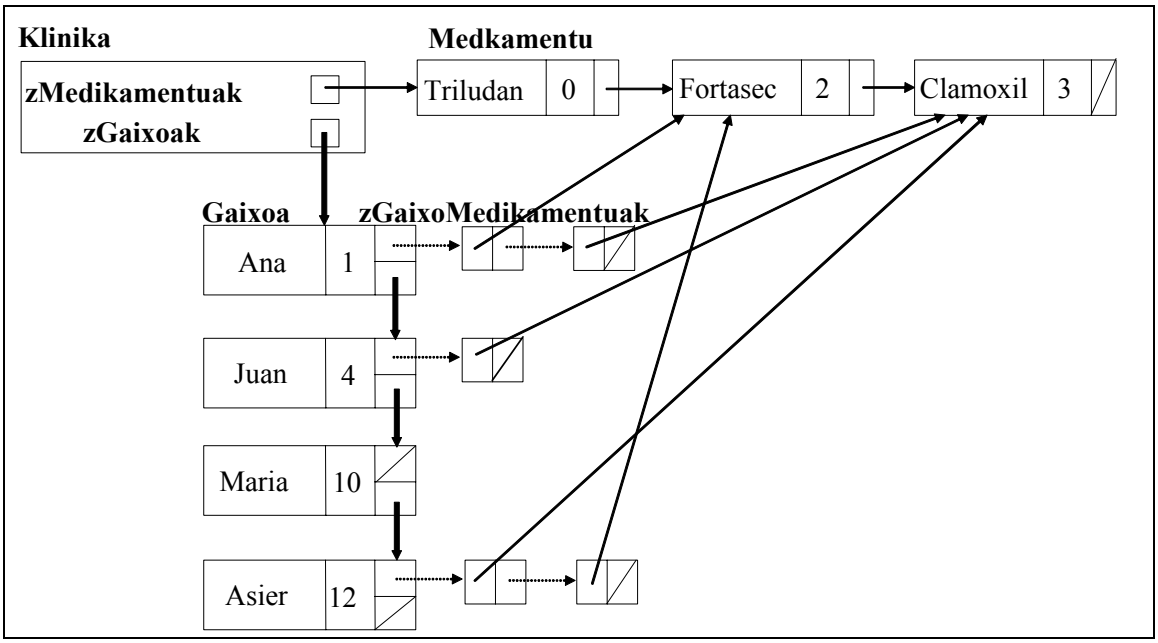
- `nireLista.errepikatuakEzabatu(new Integer(1))`



metodoa exekutatu ondoren:



(J98) *Klinika* klase bat daukagu gaixo zerrenda, eta medikamentu zerrenda baten informaziorekin. Gaixo zerrenda **ordenatua** dago **gaixoak dauden ohe zenbakiarekiko**.
Hurrengo irudian klinika bat aurkezten da 4 gaixoekin eta 3 medikamentuekin.



Gaixo zerrendaren elementu bakoitzak, izena, ohia zenbakia eta beren medikamentu zerrenda bat gordetzen du (zerrenda null izango da medikamendurik hartzen ez badu). Medikamentu zerrendako elementu bakoitzak, izena, eta hartzen duen gaixo kopurua gordetzen du.
Gaixo konkretu batentzat, medikamentu zerrendako elementu bakoitzak, erakusle bat gordetzen du medikamentu objektu (medikamentu zerrendako objektu bat) bateri
Eskatzen da:

a) **Behar diren klaseak, definitutako espezifikazioa gordetzeko.**
b) Klinika klasearen **medikamentuEsleitu** metodoaren **diseinua eta implementazioa Java-n**. Metodoak, existitzen den gaixo objektu bat eta medikamentu bat jasotzen du parametro bezala eta:
    - Medikamentu zerrendan, medikamentu berria agertzen ez bada, zerrendan txertatzen da, gaixo kopuruarekin = 1.
    - Medikamentu zerrendan, medikamentua agertzen bada, gaixo kopurua unitate batean gehitzen da.
    - Gaixoaren medikamentu zerrendan, medikamentu berria txertatzen da.

```
public void medikamentuEsleitu(Gaixo pGai,
                               String pMedikamentu)
--Pre: pGai klinika gaixo zerrendaren gaixo bat da.
```

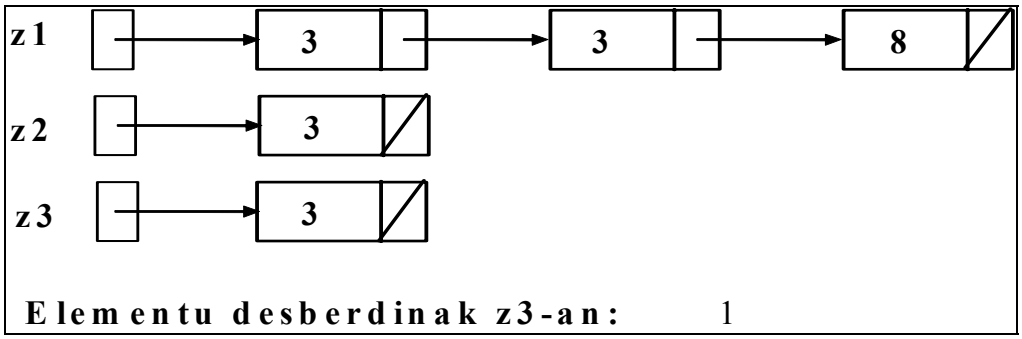
(S98)) Lehengo orriaren *LinkedList* klasea ematen dizgute. Zerrendaren elementuak goranzko ordenean ordenatuta daude.

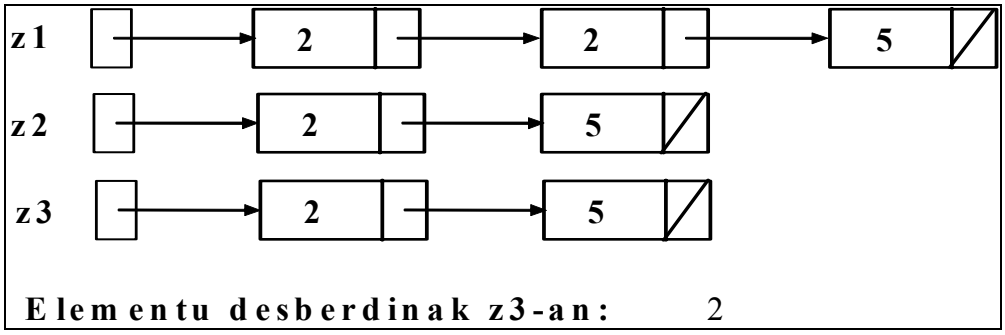
Diseina eta inplementatu *ebakidura* metodoa. Metodak, LinkedList-ko hiru objektu jasoko ditu parametro bezala (jatorrizko bi zerrendak, eta emaitzarena), eta **emaitzaren zerrendak dauzkan elementu desberdinen kopurua itzuliko du**.

int ebazpena (LinkedList z1, LinkedList z2, LinkedList z3)

Kontuan hartu beharko da, elementuak aldi bat baino gehiagotan agertu daitezkela zerrenda batean.

Adibideak **(z3 zerrenda z1 eta z2-ren ebakidura da):**





(1) J94 eta (S95) egitura berdina emanda, diseina eta inplementatu Java-n Liburutegi klaseko **bikoteEzabatu** metodoa. Metodo honek bi hitz jasoko ditu parametro bezala, bat gaztelaniaz eta bestea ingelesez eta:

- Biak osagai berdinean badaude, osagaia ezabatu egiten da.
- Osagai desberdinetan badaude, errore bat itzuliko du.
- Hitzaren bat agertzen ez bada, beste errore desberdin bat aurkeztuko du

(2) J95 (eta **J97**) egitura berdina emanda, diseinatu eta inplementatu Java-n Liburutegi klaseko itzuli (String pLib) metodoa. Metodo honek, liburu bateko informazioa jasoko du eta:

- Liburua Liburutegira itzuliko du, eta maileguan zeukan bazkideraren izena itzuliko du.
- Errore bat inprimatuko da, liburua liburutegian ez badago
- Errore bat, liburua maileguan ez badago (Suposatzen dugu, liburu bakoitzagatik, unitate bakar bat dagoela)

(3) S94 egitura berdina emanda, diseinatu eta inplementatu Java-n zIkasleak klaseko metodo bat. Metodo honek, irakasgai bat, nota bat eta ikasle baten izena eta abizena jasoko ditu eta:

- Irakasgai hori txertatuko du ikaslearen zerrendan. Suposatuko dugu kasu honetan, irakasgaiak alfebetikoki ordenatuta daudela.
- Errore bat, ikaslea agertzen ez bada.
- Irakasgai hori ikaslearen zerrendan ez gainditu bezala (<5) badago, aldatuko da irakasgairen nota.
- Irakasgai gaindituta bazegoen, errore bat inprimatu.

(4) S94 egitura berdina emanda, diseinatu eta inplementatu Java-n zIkasleak klaseko metodo bat. Metodo honek, irakasgai bat eta ikasle baten izena eta abizena jasoko ditu eta:

- Irarakasgai hori ezabatuko du ikaslearen zerrendatik. Suposatzen dugu ere, irakasgaiak alfabetikoki ordenatuta daudela.
- Errore bat, ikaslea agertzen ez bada
- Errore bat, irakasgai hori ikaslearen zerrendan agertzen ez bada.

(5) S97 ariketan, zerrendako zenbakiak ordenaturik gabe daude. Diseina eta inplementatu metodo bat, elementuak goranzko ordenean egongo balira.

(6) J98 egitura berdina emanda, diseinatu eta inplementatu Java-n Klinika klaseko metodo bat. Metodo honek, medikamendu bat jasoko du parametro bezala, eta existitzen bada ezabatuko du. Agertzen ez bada, errore bat inprimatuko da.

public void medikamentuEzabatu(Medikamentu pMed)

(7) LinkedList bi objektu emanda (suposatzen dugu elementuak goranzko ordenenan txertatu dira), diseina eta implementatu **mergeList** metodoa. Metodo honek LinkedList objektu bat itzuliko du, aurreko bi zerrendetan agertzen diren elementu guztiekin goranzko ordenean kokaturik. Zerrenda berri honetan, ez daude elementu errepikatuak.

(8) **J95** egitura berdina emanda, diseinatu eta inplementatu Java-n Liburutegi klaseko **aldatuBazkide** metodoa. Metodo honek, liburu izenburu bat eta bazkide zenbaki bat jasotzen ditu parametro bezala eta:

- Liburua bazkide batek maileguan zeukan eta bi eragiketa egiten dira:
    1. Liburua maileguan zeukan bazkiderari ezabatzen zaio
    2. Liburu hori, bazkide zenbakia eman diguten bazkideari mailegutzen zaio
- Eman diguten bazkidea, liburu hori lehendik maileguan bazeukan, ez da ezer egiten.
- Liburua liburutegian egongo ez balitz, errore bat inprimatuko da
- Liburua maileguan egongo ez balitz, beste errore bat inprimatuko da.
- Eman diguten bazkide zenbakia, inongo bazkidena bada, errore bat inprimatuko da.

```
public void aldatuBazkide (String pLiburu,
                                    int pBazkide)
```

EBAZPENAK

(J94)

1. Klaseen espezifikazioa

```
public class Hiztegi {
 LinkedList<Bikote> zGazt = new LinkedList();

 LinkedList<Bikote> zIng = new LinkedList();
}

public class Bikote {
     String pGaztelania;
     String pIngelesa;
}
```

2. Diseinu eta inplementazioa

```
public String itzuli(String p, boolean b) {
```
-- Pre: hutsa.
-- Pos: "p" hitzaren itzulpena itzultzen du.
-- b true denean, itzulpena gaztelaniatik ingelesera egiten da, eta b
-- false denean ingelesetik gaztelelaniara.
-- p ez bada agertzen, null itzultzen da.

Algoritmoa:

If b is true
1.Bilatu hitza Gaztelaniazko zerrendan
Aurkitzen bada itzuli ingeleseko itzulpena
else itzuli null
else
2. Bilatu hitza Ingeleseko zerrendan
Aurkitzen bada itzuli gaztelaniazko itzulpena
else itzuli null
endIf

Inplementazioa:

```
public String itzuli(String p, boolean b) {

   Bikote p2=null;
   boolean aurk;
   if (b) // Gaztelaniaz
```

```
{// Bilatu hitza gaztelaniazko zerrendan
  aurk=false;
   zGazt.goFirst();

   while ( (zGazt.hasMoreElements()) && !aurk) {
     p2=zGazt.getActual();
     if (p2.pGaztelania.equals(p)) aurk=true;
      else
       zGazt.goNext();
   }
   if (!aurk) //Aurkitzen ez bada, inprimatu mezua
    return null;
   else //Aurkitzen bada, ingeleseko itzulpena inprimatu:
      return(p2.pIngelesa);
 } else
   { //Bilatu hitza ingeleseko zerrendan
   aurk=false;
    zIng.goFirst();

    while ( (zIng.hasMoreElements()) && !aurk) {
      p2= zIng.getActual();
      if (p2.pIngelesa.equals(p)) aurk=true;
       else
        zIng.goNext();
    }
    if (!aurk) Aurkitzen ez bada, inprimatu mezua
     return null;
     else // Aurkitzen bada, gaztelaniazko  itzulpena inprimatu
      return(p2.pGaztelania);

  }}
```

(S94)

1. Klaseen espezifikazioa

```
 public class zIkasleak {
  LinkedList<Ikaslea> zIkasleak = new LinkedList( );
}

public class Ikaslea {
 String izena;
 String abizena;
 int zNotak =0;
 LinkedList<Irakasgaia>zIrakasgaiak=
                                    new LinkedList();
}

public class Irakasgaia {
    String izena;
    float kalifikazioa;
}
```

2. Diseinu eta inplementazioa

```
public void gaindituakInprimatu (String irak)
```

--Pre: hutsa.

--Pos: “irak” irakasgaia gainditu duten ikasleen izena eta abizena inprimatzen du

Algoritmoa:

```
Ikasleen zerrendako lehen elementuan kokatu
While  ikasleak zerrendan
        1. Bilatu irakasgai ikasleen zerrendan
          If irakasgai aurkitua eta gaindituta
                Ikasleen izena eta abizena inprimatu
          endIf
endWhile
```

```
1.  Bilatu irakasgai ikasleen zerrendan
Ikasleen zerrendako lehenengo irakasgaian kokatu
aurkitua=false
While irakasgaiak zerrendan eta ez aurkitua
        1. If egungoIrakasgai=Bilatutako irakasgaia
                aurkitua=true
           else
              aurreratu zerrendaren hurrengo irakasgaira
endWhile
```

Inplementazioa:

```
public void gaindituakInprimatu (String pIrak) {
   zIkasleak.goFirst();
   Ikaslea ik=null;
   //While ikasleak zerrendan
   while (zIkasleak.hasMoreElements()) {
    ik=zIkasleak.getActual();
    ik.zIrakasgaiak.goFirst();
    boolean aurk=false;
    Irakasgaia i=null;
        // Irakasgaia bilatu ikasleen zerendan:
    while ((ik.zIrakasgaiak.hasMoreElements())&&!aurk ) {
        i=ik.zIrakasgaiak.getActual();
        if (i.izena.equals(pIrak)) aurk=true;
         else
          ik.zIrakasgaiak.goNext();
    }
    if (aurk && (i.kalifikazioa>=5))
     System.out.println(ik.izena+" "+i.kalifizazioa);
    zIkasleak.goNext();
   }
}
```

(J95)

1. Klaseen espezifikazioa

```
public class Liburutegi {
 LinkedList<Bazkide> zBazkideak = new LinkedList( );

 LinkedList<Liburu>  zLiburuak = new LinkedList( );

}

public class Bazkide {
 String izena;
 int id;
 LinkedList<Liburu> zBakideLiburuak = new LinkedList(
);

}
public class Liburu {
 String titulo;
 String egilea;
 Bazkide maileguBazk;
}
```

2. Diseinua eta inplementazioa

```
public  void mailegu (String pBaz, String pLib)
```

-- Pre: “pBaz” bazkidea existitzen da liburutegian
--Pos: “pLib” liburua existitzen bada, inprimatu errorezko mezua. Maileguan badago, maileguan duen bazkidearen izena inprimatzen da. Maileguan ez badago, liburutegia eguneratzen da mailegu horrekin

Algoritmoa:

Bilatu liburua liburuen zerrendan
If liburua EZ aurkitua
Liburu ez dagoen mezua inprimatu
else If liburua maileguan
Bazkidearen izena inprimatu
else
1. Mailegatu liburua Bazkideari
endIf
endIf
	1. <u>Mailegatu liburua bazkideari</u>
	Mailegua txertatu bazkideen zerrendan
	Liburua bazkidearekin lotu

Inplementazioa:

```
public  void mailegu (String pBaz, String pLib){

  // Liburua bilatu liburuen zerrendan
  Bazkide   baz=bazkideBilatu(pBaz);//   String-etik   objektua
lortzen dugu
  // Bazkidea bilatu bazkideen zerendan
  Liburu  lib=liburuBilatu(pLib);//String-etik  objektua  lortzen
dugu

    if (lib==null) //Liburua EZ aurkitua
     System.out.println(pLib+" Ez dago liburutegian");
     else
      {
      if (baz==null) //Bazkidea EZ aurkitua
       System.out.println(pBaz" ez dago liburutgian");
       else
       {
        if (lib.maileguBazk!=null) //Liburua mailegatua

     System.out.println(pLib+" liburua ”+lib.maileguBazk.izena +
     “dauka”);
           else
            {
             baz.liburuMailegatu(lib);
             System.out.println(pLib+"   liburua   “+baz.izena+”
bazkideari mailegatzen zaio”);
            }
       }
      }}
```

```
public Liburu liburuBilatu(String pTit) {
   zLiburuak.goFirst();
   boolean aurk=false;
   Liburu lib=null;
   while ((zLiburuak.hasMoreElements()) && ! aurk) {
    lib=(Liburu) zLiburuak.getActual();
    if (lib.titulo.equals(pTit)) aurk =true;
     else
      zLiburuak.goNext();
   }
   if (!aurk) return null; else return lib;
}

public class Bazkide{

  public void liburuMailegatu(Liburu pLib) {
   zBazkideLiburuak.insert(pLib);
   pLib.maileguBazk=this;
   }
}
```

(S95)

1. Klaseen espezifikazioa

Idem J94

2. Diseinua eta inplementazioa

```
public void bikoteTxertatu(String h1, String h2)
```

-- Pre: hutsa.

-- Pos: Hiztegian sarrera berri bat txertatu da parametrozko hitzekin. Hitzaren bat existituko balitz, errorezko mezu bat inprimatuko litzateke.

Algoritmoa:

1. Bilatu Gaztelaniazko hitza eta aurrekoa
    If gaztelaniazko hitza EZ aurkitua
        2. Bilatu ingelesezko hitza eta aurrekoa
        If ingelesezko hitza EZ aurkitua
            Elementu bat sortu bi hitzekin
            3. Elementua txertatu gaztelaniazko zerrendan
            4. Elementua txertatu ingelesezko zerrendan
        else
            Errorea, hitz ingelesa existitzen da
        endIf
    else
        Errorea, hitz gaztelania existitzen da
    endIf

1. Bilatu Gaztelaniazko hitza eta aurrekoa
Hasieratu hGazt, gaztelaniazko lehen osagaira
Hasieratu hAurreGazt null baliora
While (hurrengo hGazt ez null) AND
(hGazt daukan hitza<txertatu nahi dugun hitza)
    hAurreGazt <--hGazt
    hGazt aurreratu hurrengo osagaira

2. Bilatu Ingelesezko hitza eta aurrekoa
Idem 1. Baino ingelesezko zerrendan

3. Osai bat txertatu gaztelaniazko zerrendan
if Gaztelaniazko hitza hasieran txertatu behar bada
    Hasieran txertatu
else
    txertatu bi osagaien artean

Inplementazioa:

```
public void bikoteTxertatu(String h1, String h2) {
 Bikote pAurreGazt=null;
 Bikote pAurreIng=null;
 boolean aurk=false;
 Bikote h3=null;
 // Bilatu osagaia gaztelaniazko hitzarekin eta aurrekoa
  zGazt.goFirst();

 while ( (zGazt.hasMoreElements()) && !aurk) {
     h3=zGazt.getActual();
     if (h3.pGaztelania.compareTo(h1)<0) {
      pAurreGazt=h3;
      zGazt.goNext();
      }
       else aurk =true;
   }
 if ((aurk) && (h3.pGaztelania.compareTo(h1)==0))
   System.out.println("Errorea ,"+h1+"  badago");
   else // Gaztelaniazko hitza EZ aurkitua
    {// Bilatu osagaia ingelesezko hitzarekin eta aurrekoa
     aurk =false;
     zIng.goFirst();

     while ( (zIng.hasMoreElements()) && ! aurk) {
      h3=zIng.getActual();
      if (h3.pIngelesa.compareTo(h2)<0) {
       pAurreIng=h3;
       zIng.goNext();
      }
       else aurk =true;
     }
```

```
if ((aurk) && (h3.pIngelesa.compareTo(h2)==0))
       System.out.println("Errorea, ingeleseko hitza badago”);
       else // Ez daude inongo biak;
       {
        System.out.println("Bikote berria "+h1+" "+h2);
        Hitza hb=new Bikote(h1,h2);
        if (pAurreGazt==null) //Gaztelaniazko lehen hitza
          zGazt.insertFirst(hb);
         else if (!zGazt.hasMoreElements()) { //Azken osagaia
          zGazt.find(pAurreGazt);
          zGazt.insert(hb);
          }
          else //
           zGazt.insert(hb); // Erdi osagaia

 if (pAurreIng==null) zIng.insertFirst(hb);
         else if (!zIng.hasMoreElements()) {
          zIng.find(pAurreIng);
          zIng.insert(hb);
          }
          else zIng.insert(hb);
        }
    }}
```

(J96)

1. Klaseen espezifikazioa

```
public class PlanBerria {
  LinkedList<Ikasle> zIkasleak = new LinkedList( );
}

public class Ikasle {
 int zEsp;
 int kredituak=0;
 LinkedList<Irkasgai> zIrakasgaiak= new LinkedList();
}

public class Irakasgai {
  String izena;
}
```

2. Diseinu eta inplementazioa

Espezifikazioa:

`public void irakasgaiGainditu(int pzEsp, String pTit, int pKred)`

--Pre: hutsa

--Pos: "pTit" irakasgaia aurrez gainditua bazegoen, errore bat inprimatuko da. "pKred" kreditoak gehi ikasleak zeuzkan gainditutako kreditoak 300 kredito gainditzen badute, errore bat inprimatuko da. EZ errorearen kasuan, irakasgai hori txertatzen zaio ikasleari, eta ikaslea ordenatzen da behar den tokian.

1. Bilatu ikaslea
If espediente EZ aurkitua
  Objektu ikasle berri bat sortu
  Kokatu ikaslea zerrrendan
else If kreditu maximoa gainditzen du
  Errorea
else
  Bilatu irakasgaia
  If irakasgai aurkitua
   Errorea
   else
    Ikasle elementua aldatu
      2. Ezabatu ikaslea zerredatik
      Txertatu ikaslea zerendan

1. Bilatu ikaslea
Hasieratu aurk=false;
While (elems zerrendan) y (EZ aurk)
Si EgungoElem=BilatutakoElem
 aurk=true;
 else hurrengoElementuraJoan;

2. Ezabatu ikaslea zerrendatik
Bilatu ikaslea zerrendan
If lehenengo ikaslea
  Ezabatu lehenengo ikaslea
else
  Ezabatu ikaslea bi aldagai erabiliz: "ikAurre" eta"ik"
endIf

Inplementazioa:

```
public   void irakasgaiGainditu(int pzEsp, String pTit, int
pKred) {
      //Bilatu ikaslea
     boolean aurk=false;
     zIkasleak.goFirst();
     Ikasle ik=null;
    while ( (zIkasleak.hasMoreElements()) && !aurk ) {
      ik=zIkasleak.getActual();
      if (ik.zEsp==pzEsp) aurk=true;
       else
       zIkasleak.goNext();
    }
    if (aurk) //Ikaslea badago
    {
      if (ik.krediuak+pKred>300) // Kredituak gainditzen ditu
     System.out.println("Errorea, 300 kred. baino gehiago");
      else
       if (ik.aurrezGainditua(pTit)) //Irakasgai aurrez gainditua
        System.out.println("Error, irakasgai gainditua");
         else //Ikaslea badago eta irakasgaia aurrez gainditu gabe
          {
            ik.irakasgaiaGehitu(pTit,pKred);
            ikasleIzabatu(ik);
            ikasleKokatu(ik);
      } }else  //Ikaslea ez dago
        {
         Ikasle nIk=new Ikasle(pzEsp);
         nIk. irakasgaiaGehitu(pTit,pKred);
           ikasleKokatu(nIk);
        }
    }
}
public  void ikasleEzabatu(Ikasle pIk) {
      boolean aurk= zIkasleak.find(pIk);
      if (aurk) zIkasleak.remove(pIk); }
```

(S96)

1. **Klaseen espezifikazioa**

   Idem (J96)

2. **Diseinua eta implementazioa**

```
public void ikasleKokatu (Ikasle ikBerria, Ikasle
ikZaharra)
```

-- Pre: "ikBerria" ez da agertzen "ikasleak" zerrendan.
-- "ikZaharra" null desberdina bada, ikasle zerrendaren elementu bat izango da eta ikZaharra.zEsp =ikBerria.zEsp.
--Pos: ikZaharra=null denean "ikBerria" objektua zerrendan kokatuko da goranzko ordenean kreditu zenbakiagatik eta kreditu berdin dutenen artean, gorazko ordenenan espediente zenbakiagatik.
-- ikZaharra!=null denean eta "ikZaharra" y "ikBerria" objetuek erreferenziatzen duten informazio berdina denean, ez da ezer egiten. Berriz, erreferenziatzen duten informazioa desberdina bada, errore bat inprimatuko da.

```
If (ikZaharra)==null
   1. ikaslea kokatu bere ordenean
else
   2. Konparatu ikZaharra eta ikBerria
      errorea desberdinak badira
EndIf
```

1. Ikaslea kokatu bere ordenean
While ikasle irakasgaiekin "irakasg"
Ikasle.irakasgaiGainditu("irakasg")*

1. Konparatu ikZaharra eta ikBerria errorea desberdinak badira

```
If Kreditu desberdin
   Errorea
else
   Bilatu lehen irakasg. desberdin
   If aurkitu errorea
EndIf
```

(*) J96 azterketa

Implementazioa

```
public void ikasleKokatu (Ikasle ikBerria, Ikasle ikZaharra)
{
  if (ikZaharra==null) {
   ikBerria.zIrakasgaiak.goFirst();
   boolean lehena=true; //kredituak eramaten duena
   while (ikBerria.zIrakasgaiak.hasMoreElements()) {
     Irakasgaia i=ikBerria.zIrakasgaiak.getActual();
       if (lehena) {
        lehena=false;
        irakasgaiGainditu(ikBerria.zEsp,i.izena,
                                          ikBerria.kredituak);
        } else
         irakasgaiGainditu(ikBerria.zEsp,i.izena,0);
        ikBerria.zIrakasgaiak.goNext();
    }}}
```

(J97)

1. **Klaseen espezifikazioa**

   Idem (J95)

2. **Diseinua eta inplementazioa**

```
public void bazkideaEzabatu(Bazkide pBaz)
```

-- Pre: hutsa.

-- Pos: Liburutegia pBaz gabe. Bazkideak zeuzkan liburuak, liburutegian daudela agertuko dira (mailegu aske). Bazkidea agertzen ez bada, errore bat inprimatuko da.

Algoritmoa:

1. Bazkidearen liburuak itzuli **2. Bazkidea ezabatu**

1. Bazkidearen liburuak itzuli
While liburuak mailegu-zerrendan Liburua mailegu-aske ipini mailegua ezabatu endWhile

Inplementazioa:

```
public void bazkideaEzabatu(Bazkide pBaz) {
   //Bazkidea bilatu
   boolean aurk=zBazkideak.find(pBaz);
   if (aurk) //Bazkidea badago
  // Liburuak itzultzen ditu liburutegira
   System.out.println("Hurrengo liburuak itzultzen dira:");
   pBaz.zBazkideLiburuak.goFirst();
   while (pBaz.zBazkideLiburuak.hasMoreElements()) {
    Liburu lib=pBaz.zBazkideLiburuak.getActual();
    lib.maileguBazk=null;
    System.out.println(lib.titulo);
    pBaz.zBazkideLiburuak.goNext();
   }
    pBaz.zBazkideLiburuak.empty();
   // Bazkidea zerrendatik ezabatzen du
      zBazkideak.remove(pBaz);}
```

(S97)

1. **Klaseen espezifikazioa**

   LinkedList klasea erabiltzen da soilik.

2. **Diseinua eta inplementazioa**

```
public void errepikatuakEzabatu(Object pObj)
```

-- Pre: hutsa
-- Pos: Elementua agertzen bada eta aldi bat baino gehiagotan agertzen bada, lehenengo agerketa utziko da, beste guztiak ezabatuz. Elementua existitzen bada, eta aldi bat agertzen bada, ez da ezer egiten. Elementua agertzen ez bada, errore mezu bat inprimatzen da.

Algoritmoa:

While elemetuak zerrendan
  If bilatutako elementua
    If Lehenengo agerpena
      Ezer ez egin
    else
      Zerrendatik ezabatu
endWhile
if elementua ez aurkitua ERROREA

Inplementazioa:

```
public void errepikatuakEzabatu(T pObj) {
  goFirst();
  boolean goFirst=true;

  while (hasMoreElements()) { //If elementuak zerrendan
   if (getActual().equals(pObj))//if bilatutako elementua
    if (goFirst) //If lehen agerpena
     {
      goFirst=false;
      goNext();
     }
      else //Ez da aurrenekoa, orduan ezabatu
         removeActual();
    else
       goNext();
   }
   if (goFirst){// Zerrendan agertzen ez bada, ERROREA
     System.out.println();
     System.out.println("ERROREA  "+pObj+"  ez  da  agertzen
zerrendan");
    }
  }
```

(J98)

1. **Klaseen espezifikazioa**

```
public class Klinika {
  LinkedList<Gaixoa>zGaixoak = new LinkedList();
  LinkedList<Medikamendu> zMedikamentu =
                                  new LinkedList();
}

public class Medikamentu {
   String izena;
   int zGaixoak;
}
public class Gaixoa {
 String izena;
 int ohea;
 LinkedList<Medikamendu> zMedGaixoa =
                                   new LinkedList();
}
```

2. **Diseinua eta inplementazioa**

```
public void medikamentuGehitu(Gaixoa pGai, String
pMedikamendu){
```

-- Pre: "pGai" gaixoa, gaixoen zerrendako elementu bat da.
"pMedikamentu" medikamentu berri bat da gaixoarentzat (ez dago bere medikuntza zerrendan).

-- Pos: pMedikamendu, medikamentu zerrendan txertazen da agertzen ez bada.
Medikamentu badago, gaixoen zenbatzailea gehitu da. Gaixoaren medikamentu zerrendan elementu berri bat txertatzen da, aurreko edozein kasu guztietan.

Algoritmoa:

Bilatu medikamentua
If ez aurkitua
  Txertatu medikamentu berria
Gaixo zenbatzailea gehitu
Gaixoa sortu eta txertatu medikamentu berria

Inplementazioa:

```
public void medikamentuGehitu(Gaixoa pGai, String pMedik){
   // medikamendu zerrendan bilatu
zMedikamentu.goFirst();
boolean aurk=false;
Medikamentu pMed;
while ((zMedikamentu.hasMoreElements()) &&
!((zMedikamentu.getActual()).izena.equals(pMedik)))
     zMedikamentu.goNext();

if (!zMedikamentu.hasMoreElements()){//medikamentua ez da existitzen
      pMed=new Medikamentu(pMedik);
      zMedikamentu.insertFirst(pMed);
     } else pMed=(Medikamentu) zMedikamentu.getActual();

    pMed.zGaixoak++;
    pGai.zMedGaixoa.insert(pMed); //Gaixoan medikamentu berria txertatu
  }
```

(S98)

1. **Klaseen espezifikazioa**
   LinkedList klasea erabiltzen da soilik.

2. **Diseinu eta inplementazioa**

```
public static int ebakidura(LinkedList pL1,LinkedList pL2,LinkedList pL3)
```

--Pre: “pL1” y “pL2” ordenatuta daude goranzko ordenean. Osoak errepikatu daitezke zerrenda batean, eta elementu bat agertu daiteke bi zerrendetan.

--Pos: “pL3” beste zerrenda ordenatu bat da beste bi zerrenden ebakidurarekin. Sarrerako bi zerrenden elementu berdin bakoitzari, Irteerako elementu berri bat dagokio. Elementua sarrerako zerrendetan errepikatzen bada, bi zerrenden errepikapen kopuru txikienen agerpenak txertatuko dira.
Metodoak, irteerako elementu desberdinen kopurua itzultzen du.

Algoritmoa:

Hasieratu zerrenda indizeak lehen posizioan eta zenbatzailea
Hasieratu emaitza zerrendaren indizea
1. **2 zerrenda korritu** ebakidura lortuz eta zenbatuz
Zenbatzailea itzuli

1. **<u>2 zerrenda korritu ebakidura lortuz eta zenbatuz</u>**
While 2 zerrendak elementuekin
If berdinak
    Elementu bat txertatu emaitza zerrendan
    Bi zerrendetan aurreratu
    If elementua!=azkenekoa zenbatzailea gehitu
else if (egungoa<lehenengo zerrendako elementua)
    aurreratu lehenengo zerrendan
else if (egungoa< bigarren zerrendako elementua)
    aurreratu bigarren zerrendan

Inplementazioa:

```
public static int ebakidura(LinkedList pL1,LinkedList
pL2,LinkedList pL3) {
// Hasieratu zerrendak eta zenbatzailea
  pL1.goFirst();
  pL2.goFirst();
  int zDesb=0;
  boolean zBerria=true;
 //2 zerrendak korritu ebakidura lortuz eta gehituz
  while ( pL1.hasMoreElements() && pL2.hasMoreElements() ) {
   Integer i1=pL1.getActual();
   Integer i2=pL2.getActual();
   if (i1.equals(i2)) {
    pL3.insert(i1);
    pL1.goNext();
    pL2.goNext();
  // Zenbatzailea gehitu (elementua!=azkeneko elementua)
    if (zBerria) {zBerria=false; zDesb++;}
    } else
     if (i1.intValue()<i2.intValue()) {
      zBerria=true;
      pL1.goNext();
      } else
       { zBerria=true;
       pL2.goNext();
       }
   }
   return zDesb;
  }
```