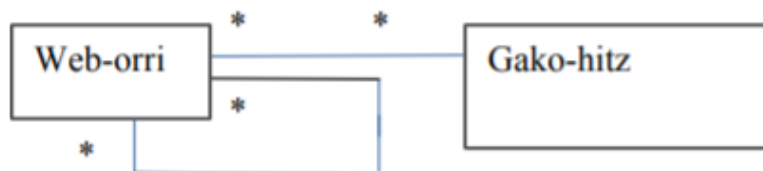


Web-a kudeatzeko aplikazioa



Egileak:

Aitor San José, Martin Amezola, Leire Garcia

Irakasgia:

Datu-Egiturak eta Algoritmoak

Irakasleak:

Iñigo Mendialdua eta Koldobika Gojenola

2. maila

46. taldea

2020.eko urriaren 12

Aurkibidea

1	Sarrera eta arazoaren aurkezpena	2
2	Diseinua eta bere eboluzioa	3
3	Datu egituren diseinua	6
4	Metodo nagusien diseinu eta inplementazioa	7
4.1	Fitxeroen karga	7
4.1.1	Lehen bertsioa	7
4.1.2	Bigarren bertsioa	7
4.2	Web-orri baten bilaketa url bidez	10
4.3	Web-orri berri baten txertaketa	11
4.4	Web-orri bat ezabatu	11
4.5	Web batek estekatzen dituen web-orrien zerrenda bueltatu	12
4.6	Gako-hitz bat emanda, gako-hitz hau duten web-orrien zerranda bueltatu	13
4.7	Web-orrien zerrenda fitxategitan gorde	13
4.8	Web-orrien zerrenda ordenatua lortu	14
5	Kodea	15
5.1	WebZerrenda	15
5.2	Web	19
5.3	GakoHitzZerrenda	20
5.4	Hitza	21
5.5	Teklatua	22
5.6	HasieratuPraktika	23
6	Emaitza enpirikoak	27
6.1	GakoHitzZerrenda	27
6.2	WebZerrenda	27
6.3	Hitza	27
7	Ondorioak	28
8	Erreferentziak	29

1 Sarrera eta arazoaren aurkezpena

Datu-Egiturak eta Algoritmoak ikasgaiako lehenengo eginkizuna Web-orri kopuru handia kudeatuko dituen aplikazioa sortzea da.

Ikasgai honetan, asko azpimarratzen da programaren kostua, beraz inplementatzerako orduan datuen maneiatzearen arabera datu egitura bat edo bestea erabili dugu, inplementazioa gero eta eraginkorragoa izatearren.

Web orri kopuru handia daukagu eta web orri bakoitzak beste hainbat web orritara nabigatzeko estekak ditu, gainera, web-orri bakoitzak zenbait gako-hitzekin lotuta dago. Lehen eginkizun honetan, web orri horiek kudeatuko dituen aplikazioa sortu beharko dugu. Honetarako, ezagunak ditugun *ArrayList*-ak eta berria den *HashMap* egiturak erabiliko ditugu.

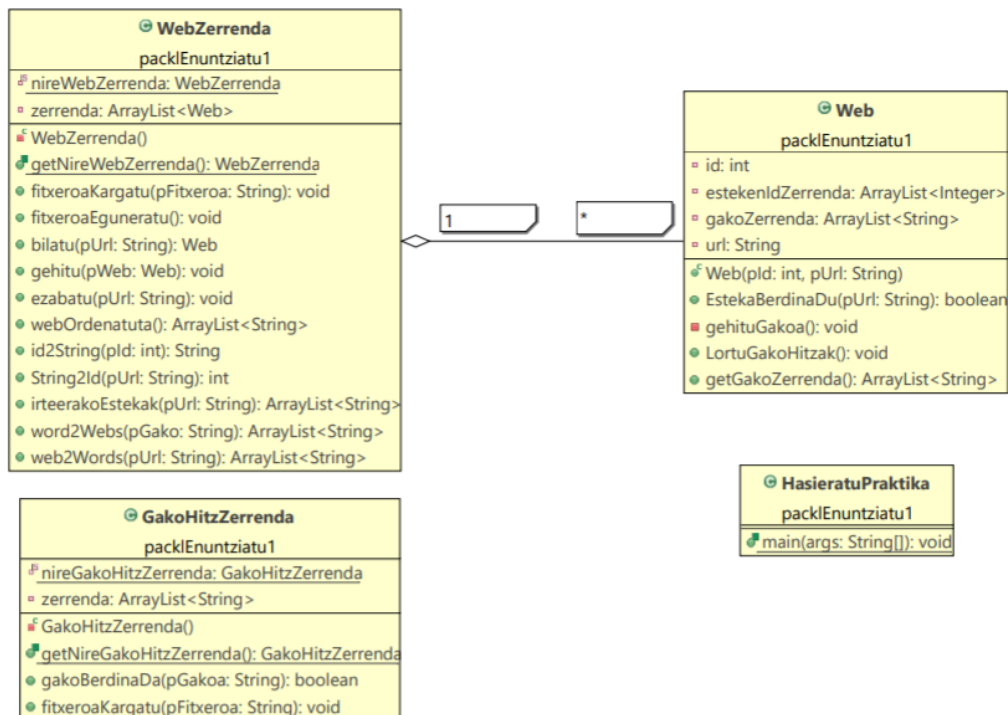
Aplikazio honek honako eragiketak egingo ditu modu eraginkorrean:

- Datuak kargatu fitxategietatik.
- Web-orri baten bilaketa.
- Web-orri berri baten txertaketa.
- Web-orri bat ezabatu.
- Web-orri bat emanda, honek estekatzen dituen web-orrien zerrenda bueltatu.
- Gako-hitz bat emanda, gako-hitz hau duten web-orrien zerranda bueltatu.
- Web-orrien zerrenda fitxategitan gorde.
- Web-orrien zerrenda ordenatua lortu (alfabetikoki).

2 Diseinua eta bere eboluzioa

Hasieratik, diseinua nahiko bideratuta zegoen.

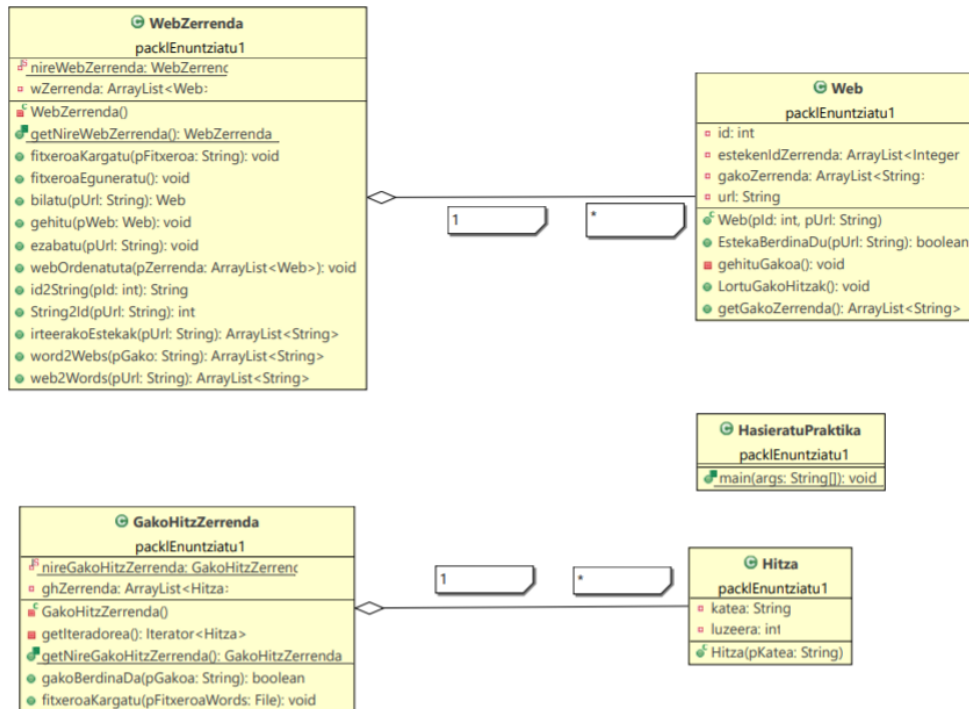
Lehen klase diagraman (1. irudia), lau klase sartu genituen, 2 EMArekin eta main metodoa izango zuen klasearekin: WebZerrenda, GakoHitzZerrenda, Web eta HasieratuPraktika (lehenengo biak EMAk eta azkenengoa metodo nagusia duen klasea izanik).



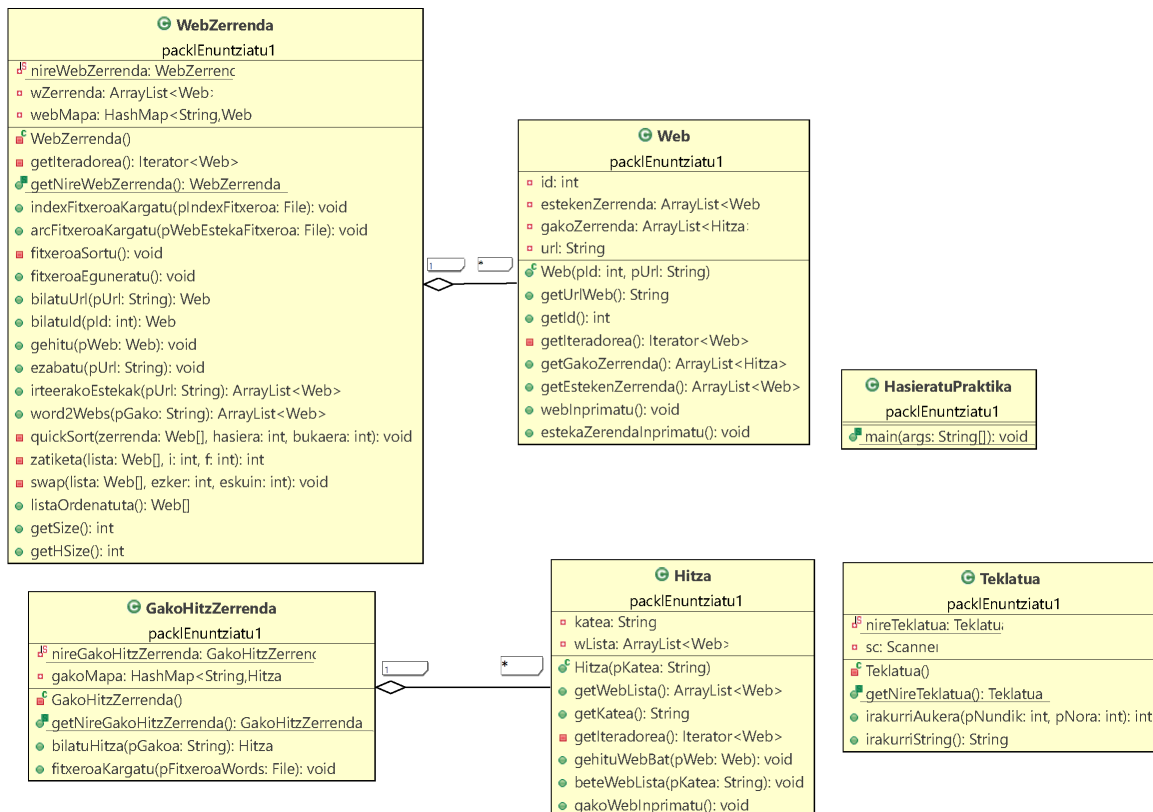
1. irudia: Hasierako diseinua.

Diseinuak arazoak zituen, hainbat elementu falta baitziren, haien artean klase berri bat eta zenbait metodo. Hasierako diseinuak GakoHitzZerrenda String-eko ArrayList bat zuen baina hori oso eraginkorra ez zela ohartu ginen, beraz, Hitza klasea sortu genuen, atribututzat String-az gain hitzaren luzera gorde ahal izateko (2. irudia).

Momentu honetan, proiektua ez zegoen guztiz bukatuta. Hitza klasean luzera atributua soberan zegoela, web *ArrayList*-a atributua eta zenbait metodo faltan genituela konturatu ginen, eta azken hauek implementatu genituen. Eraginkortasunaren harira, irakasleak klasean *HashMap* datu egitura aipatu zuen, eta horren inguruan gure diseinua berreraiki genuen (3. irudia).



2. irudia: Bigarren diseinua.



3. irudia: Hirugarren diseinua.

Azkenengo diseinu honetan WebZerrenada eta GakoHitzZerrenda klaseetan *HashMap*-ak implementatu genituen, baita Teklatua EMA klasea ere, menuko aukerak hautatzeko eta teklatutik informazioa sartu ahal izateko..

3 Datu egituren diseinua

Proiektu honetan hiru datu egitura desberdin erabili ditugu; *Array*-ak, *ArrayList*-ak eta *HashMap*-ak.

Datu asko gorde behar dituzten bi klase ditugu: *WebZerrenda* eta *GakoHitzZerrenda*. Bi klase hauetan *HashMap* bat erabili dugu, objektuak sartzeko, bilatzeko edo ezabatzeko kostu konstantea duelako; *ArrayList*-ak adibidez, elementu bat bilatzeko kostu lineala du, eta hain datu kopuru handiarekin denbora asko behar izango luke. *HashMap*-arekin exekuzio denbora asko jaisten da.

Horretaz aparte, lau *ArrayList* sortu ditugu: *Web* bakoitzak estekatzen dituen *Web*-en zerrenda, *Web* batek dituen *GakoHitz*-en zerrenda, *GakoHitz* bakoitzak estekatzen dituen *Web*-en zerrenda, eta, *WebZerrenda* ordenatzeko erabiliko dugun zerrenda. *WebZerrenda*-ren zerrendan izan ezik, beste zerrendetan elementu gehiegirik ez daudenez, ez da arazorik egongo hauekin lan egitean. *WebZerrendaren* kasuan, *ArrayList*-aren bitartez alfabetikoki ordenatzea askoz errazagoa denez erabili dugu.

Bukatzeko, *Array*-ak fitxeroak kargatzean erabili ditugu. Datuak kargatzean, lerro bakoitza irakurri eta *split* metodoa erabiltzean, bi *Array* edo gehiagotan (fitxeroaren arabera) banatu egiten da lerroa, *HashMap*- eta *ArrayList*-etan sartzeko.

4 Metodo nagusien diseinu eta implementazioa

4.1 Fitxeroen karga

Fitxeroak kargatzeko, bi bertsio izan ditugu proiektuan aurreratu ahala:

4.1.1 Lehen bertsioa

Lehen bertsioan gure hiru fitxeroak, "Words.txt", "index.txt" eta "pId-arc-1-N.txt" kargatu genituen, esandako ordenean. Horretarako `FileReader` eta `BufferedReader` klaseak erabili genituen, fitxeroaren irakurketa errazten dituztelako.

Lehen pausuan, "words.txt" fitxeroa kargatzean, 'Hitza' motatako objektua sortzen genuen hitz bakoitzeko eta gero sartzen genuen ere `GakoHitzZerrendak` duen `HashMap`-ean. `HashMap` horretan, 'Key' bezala hitzaren `String`-a edo 'katea' zegoen eta 'Value' bezala `Hitza` objektua.

`HashMap`-ean sartu eta gero, metodo bat egin genuen orain sortutako hitza web guztien url-ekin konparatzen duena ikusteko zein url-tan zegoen. Noski, hau errore bat izan zen zeren eta "index.txt" fitxeroa oraindik ez genuelako kargatu eta `Web` objektuen `ArrayList`-a hutsik zegoelako. Hurrengo apartaduan 2. Bertsioan arazo hau ikusi eta konpondu genuen.

Bigarren pausuan, "index.txt" fitxeroa kargatu egiten genuen. Horretarako, lerro zerro ere `Web` objektuak sortzen genituen (`String.split()` erabiliz indizea eta url-a banatzeko) eta gero `HashMap`-ean eta `ArrayList`-ean gordetzen genituen.

Hirugarren pausuan, "pId-arc-N.txt" fitxeroa kargatzen genuen (`String.split()` erabiliz lehen indizea eta estekak banatzeko eta berriz ere estekak haien artean banatzeko), eta metodo baten bidez `Web` bakoitzaren estekak irakurtzen genituen eta sartzen genituen web horren esteken `Zerrenda ArrayList`-ean.

4.1.2 Bigarren bertsioa

4.1.2.1 words fitxeroa kargatu

public void fitxeroaKargatu(File pFitxeroaWords) throws FileNotFoundException

// aurre: Fitxeroaren absolute edo relative path sartu behar da.

// post: Fitxeroaren datuak programan kargatu dira. Gako-hitzen HashMap-a bete egin da.

- Proba kasuak
 1. Datuak programan kargatu dira
 2. Fitxategia kargatzean errore bat egon da

- Algoritmoa

```

FileReader fr = new FileReader (pFitxeroaWords);
BufferedReader b = new BufferedReader(fr);

String fila;

try
{
    while ((fila=b.readLine())!=null)
    {
        Hitza h = new Hitza(fila);
        this.gakoMapa.put(h.getKatea(), h);
    }
} catch (Exception e) {
    e.printStackTrace();
} finally {
    try {
        if ( null != fr ) {
            fr.close();
        }
    } catch (Exception e2) {
        e2.printStackTrace();
    }
}

```

- Kostua: $O(n)$
n irakurri diren lerro kopurua.

4.1.2.2 index fitxeroa kargatu

public void indexFitxeroaKargatu(File pIndexFitxeroa) throws FileNotFoundException

// aurre: Fitxeroaren absolute edo relative path sartu behar da.

// post: Fitxeroaren datuak programan kargatu dira. Web HashMap-a eta ArrayList-a bete egin dira. Gako-hitz bakoitzak estekatzen dituen web-en zerrenda bete egin da.

- Proba kasuak

1. Datuak programan kargatu dira
2. Fitxategia kargatzean errore bat egon da

- Algoritmoa

```

FileReader fr = new FileReader (pIndexFitxeroa);
BufferedReader b = new BufferedReader(fr);

String lerro;
WebZerrenda wz = WebZerrenda.getNireWebZerrenda();
GakoHitzZerrenda ghz = GakoHitzZerrenda.getNireGakoHitzZerrenda();

try {
    while ((lerro = b.readLine())!=null)

```

```

{
    String zatiak[] = lerro.split(" ");
    Web w = new Web(Integer.parseInt(zatiak[1]), zatiak[0]);
    wz.gehitu(w);
    int luz = zatiak[0].length();
    int iterazio;
    int dif = 3;
    Hitza atxe;
    String sub;
    while (dif <= luz)
    {
        iterazio = 0;
        while (iterazio <= luz - dif )
        {
            sub = zatiak[0].substring(iterazio, iterazio + dif);
            atxe = ghz.bilatuHitza(sub);
            if (atxe!=null)
            {
                atxe
                atxe.gehituWebBat(this.webMapa.get(zatiak[0]));
            }
            iterazio++;
        }
        dif++;
    }
}
} catch (Exception e)
{
    e.printStackTrace();
} finally {
    try {
        if ( null != fr ) {
            fr.close();
        }
    } catch (Exception e2) {
        e2.printStackTrace();
    }
}
}

```

- Kostua: $O(w \cdot a)$
w kargatuko diren web-wn kopurua eta a web bakoitzaren url-a zatitu ahal den azpihitz kopurua.

4.1.2.1 arc fitxeroa kargatu

public void arcFitxeroaKargatu(File pWebEstekaFitxeroa) throws FileNotFoundException

// aurre: Fitxeroaren absolute edo relative path sartu behar da.

// post: Fitxeroaren datuak programan kargatu dira. Web bakoitzak estekatzen dituen web-en ArrayList-a bete da.

- Proba kasuak
 1. Datuak programan kargatu dira
 2. Fitxategia kargatzean errore bat egon da

- Algoritmoa

```

FileReader fr = new FileReader (pWebEstekaFitxeroa);
BufferedReader b = new BufferedReader(fr);

String lerro;
Web w=null;
int luzera=0;
int i;
WebZerrenda wz = WebZerrenda.getNireWebZerrenda();

try {
    while ((lerro = b.readLine())!=null)
    {
        String zatiak1 [] = lerro.split(" —>");
        w=wz.bilatuId(Integer.parseInt(zatiak1[0]));
        if (zatiak1.length == 2)
        {
            String zatiak2 [] = zatiak1[1].split(" ");
            luzera=zatiak2.length;
            i=1;
            while(i<=luzera-1) {
                w.getEstekenZerrenda().add(wz.bilatuId(Integer.parseInt(←
                    zatiak2[i])));
                i++;
            }
        }
    }
} catch (Exception e)
{
    e.printStackTrace();
} finally{
    try{
        if( null != fr ){
            fr.close();
        }
    } catch (Exception e2){
        e2.printStackTrace();
    }
}
}

```

- Kostua: $O(N)$
n irakurri diren lerro kopurua.

4.2 Web-orri baten bilaketa url bidez

```
public Web bilatuUrl(String pUrl)
```

// aurre: Bilatu nahi den web-aren url-a parametro gisa sartu behar da.

// post: Web-a aurkitzen badu web-a bera bueltatzen du, null bestela.

- Proba kasuak
 1. Web-a badago
 - (a) Elementu batez osatutako listan
 - (b) Elementu askoz osatutako listan

2. Web-a ez badago

- (a) Lista hutsean
- (b) Elementu askoz osatutako listan

- Algoritmoa

```
return this.webMapa.get(pUrl);
```

- Kostua: $O(1)$

HashMap-ean elementu bat bilatzeak kostu konstantea du.

4.3 Web-orri berri baten txertaketa

```
public void gehitu(Web pWeb)
```

// aurre: Gehitu nahi den web-a parametro gisa sartu behar da. Metodo hau pentsatuta dago sartutako web-a zerrendan ez egoteko txertaketa egin baino lehen.

// post: Web-a zerrendaren azken posizioan txertatzen du eta web-en HashMapean ere txertatzen du.

- Proba kasuak

1. Web-a badago
2. Web-a ez badago

- Algoritmoa

```
this.wZerrenda.add(pWeb);
this.webMapa.put(pWeb.getUrlWeb(), pWeb);
```

- Kostua: $O(1)$

HashMap-ean elementu bat txertatzeak kostu konstantea du, *ArrayList*-ean ere.

4.4 Web-orri bat ezabatu

```
public void ezabatu(String pUrl)
```

// aurre: Ezabatu nahi den web-aren url-a parametro gisa sartu behar da.

// post: Web-a aurkitzen badu, listatik ezabatu egiten du.

- Proba kasuak
 1. Web-a badago
 2. Web-a ez badago
- Algoritmoa

```
Web w=bilatuUrl(pUrl);
if(w!=null) {
    webMapa.remove(pUrl);
    this.wZerrenda.remove(w);
}
```

- Kostua: $O(1)$
HashMap-ean elementu bat ezabatzeak kostu konstantea du, *ArrayList*-ean ere.

4.5 Web batek estekatzen dituen web-orrien zerrenda bueltatu

```
public ArrayList<Web> irteerakoEstekak(String pUrl)
```

```
// aurre: Web baten url-a jasoko du parametro gisa .
```

// post: Url-arenkin HashMap-ean web-a bilatuko du eta estekatzen dituen web-en lista bueltatuko du. Aurkitu ezean, zerrenda hutsa bueltatuko du.

- Proba kasuak
 1. Web-a badago
 - (a) Esteken zerrenda hutsa bada
 - (b) Esteken zerrenda hutsa ez bada
 2. Web-a ez badago
- Algoritmoa

```
Web w=bilatuUrl(pUrl);
ArrayList<Web>a=new ArrayList<Web>();
if(w!=null) {
    a=w.getEstekenZerrenda();
}
return a;
```

- Kostua: $O(1)$
HashMap-ean elementu bat bilatzeak kostu konstantea du, eta Web-ak estekatzen dituen web-en zerrenda bueltatzeak ere.

4.6 Gako-hitz bat emanda, gako-hitz hau duten web-orrien zerrenda bueltatu

```
public ArrayList<Web> word2Webs(String pGako)
```

```
// aurre: Gako-hitz bat jasoko du parametro gisa.
```

```
// post: Gako-hitza jakinda, HashMap-ean bilatu eta aurkitzean, gako-hitz hori daukaten web-orriak itzultzen ditu eta gako-hitza web-orriek duten gako-hitzen zerrendan sartzen du. Aurkitzen ez badu, zerrenda hutsa bueltatuko du.
```

- Proba kasuak
 1. GakoMapan ez dagoen hitz bat sartzea.
 2. GakoMapan dagoen hitz bat sartzea.
- Algoritmoa

```
ArrayList<Web> ema = new ArrayList<Web>();
Iterator<Web>itr = this.getIteradorea();
Web w;
while (itr.hasNext()){
    w = itr.next();
    if (w.getUrlWeb().contains(pGako)) {
        w.getGakoZerrenda().add(GakoHitzZerrenda.getNireGakoHitzZerrenda().←
            bilatuHitza(pGako));
        ema.add(w);
    }
}
return ema;
```

- Kostua: $O(n)$
n web-en kopurua izanda, While-ak web-en *ArrayList* guztia errekorritzen duelako.

4.7 Web-orrien zerrenda fitxategitan gorde

```
public void fitxeroaEguneratu() throws IOException
```

```
// aurre: -
```

```
// post: Web-en url-ak fitxero berri batean sartuko dira.
```

- Proba kasuak
 1. Fitxategia sortzea
 2. Fitxategia sortzean arazoa egotea

- Algoritmoa

```

FileWriter fitxero= null;
PrintWriter pw=null;
fitxeroaSortu();
try {
    fitxero= new FileWriter("");
    pw=new PrintWriter(fitxero);
    for(int i=0; i<wZerrenda.size();i++) {
        pw.println(bilatuId(i).getUrlWeb()+" "+i);
    }
} catch(Exception e) {
    e.printStackTrace();
}finally {
    try {
        if(null!=fitxero) {
            fitxero.close();
        }
    } catch(Exception e2) {
        e2.printStackTrace();
    }
}

```

- Kostua: $O(n)$
n web-en zerrenda izanda, While-ak web-en *ArrayList* guztia errekorritzen duelako.

4.8 Web-orrien zerrenda ordenatua lortu

```

public Web[] listaOrdenatuta()
// aurre:-
// post: HashMap-atik lortutako web-en zerrenda ordenatuta bueltatuko du.

```

- Proba kasuak
 1. Lista ordenatuta egotea
 2. Lista ordenatu gabe egotea
 3. lista hutsa egotea
- Algoritmoa

```

Collection<Web> c= this.webMapa.values();
Web[] zerrenda= c.toArray(new Web[c.size()]);
quickSort(zerrenda,0,zerrenda.length-1);
return zerrenda;
}

```

- Kostua: $O(n \log n)$
quicksort metodoak duen kostua da.

5 Kodea

5.1 WebZerrenda

```

package pack1Enuntziatu1;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.*;
public class WebZerrenda {

    // atributuak
    private static WebZerrenda nireWebZerrenda = null;
    private ArrayList<Web>wZerrenda;
    private HashMap<String, Web> webMapa; //HashMap sortu behar dugu non Key = url↔
        eta Value = Web objektua

    // eraikitzailea
    private WebZerrenda() {
        this.wZerrenda = new ArrayList<Web>();
        this.webMapa = new HashMap<String, Web>();
    }

    // gainontzeko metodoak
    private Iterator<Web> getIteradorea(){
        return this.wZerrenda.iterator();
    }

    public static WebZerrenda getNireWebZerrenda() {
        if(nireWebZerrenda == null){
            nireWebZerrenda = new WebZerrenda();
        }
        return nireWebZerrenda;
    }

    public void indexFitxeroaKargatu(File pIndexFitxeroa) throws ↵
        FileNotFoundException {
        //Metodo hau jasotzen du: Fitxeroa web orrien url-ekin eta indizeekin.
        //Metodo hau sartzen du informazio hori web objektuetan, gero web ↵
            zerrendan sartzeko
        // eta baita ere string en zerrendan (geroago alfabetikoki ordenatu ↵
            beharko duguna) eta
        // HashMapean sartuko duguna

        FileReader fr = new FileReader (pIndexFitxeroa);
        BufferedReader b = new BufferedReader(fr);

        String lerro;
        WebZerrenda wz = WebZerrenda.getNireWebZerrenda();
        GakoHitzZerrenda ghz = GakoHitzZerrenda.getNireGakoHitzZerrenda();

        try {
            while ((lerro = b.readLine())!=null)
            {
                String zatiak[] = lerro.split(" "); // Zatitzen dugu web url eta ↵
                    indizea
                // Sortzen dugu Web objektu bat eta sartzen dugu indizea eta url ↵
                    parametro bezala
                Web w = new Web(Integer.parseInt(zatiak[1]), zatiak[0]);
                //gehitzen dugu web zerrendara
            }
        }
    }
}

```



```

        wz.gehitu(w);
        int luz = zatiak[0].length();
        int iterazio;
        int dif = 3;
        Hitza atxe;
        String sub;
        while (dif <= luz){
            iterazio = 0;
            while (iterazio <= luz - dif ){
                sub = zatiak[0].substring(iterazio, iterazio + dif);
                atxe = ghz.bilatuHitza(sub);
                if (atxe!=null) { // Aurkitzen badu
                    atxe.gehituWebBat(this.webMapa.get(zatiak[0]));
                }
                iterazio++;
            }
            dif++;
        }
    }
} catch (Exception e){
    e.printStackTrace();
} finally { // Finally hay da fitxeroa ixteko zerbait txarto badoa.
    try{
        if( null != fr ){
            fr.close();
        }
    } catch (Exception e2){
        e2.printStackTrace();
    }
}
}

public void arcFitxeroaKargatu(File pWebEstekaFitxeroa) throws ←
    FileNotFoundException {
    //Pre: Metodo hau jasotzen du: Fitxeroa web orrien id-ekin eta estekatzen ←
        dituen web-en indizeekin.

    // Post: Web bakoitzaren esteka zerrendak osatzen ditu

    FileReader fr = new FileReader (pWebEstekaFitxeroa);
    BufferedReader b = new BufferedReader(fr);

    String lerro;
    Web w=null;
    int luzera=0;
    int i;
    WebZerrenda wz = WebZerrenda.getNireWebZerrenda();

    try {
        while ((lerro = b.readLine())!=null){
            String zatiak1[] = lerro.split(" —>"); // Zatitzen dugu estekak ←
                eta indizea
            w=wz.bilatuId(Integer.parseInt(zatiak1[0])); //w-n gorde behar ←
                ditugu esteka guztiak
            if (zatiak1.length == 2){ // zatiak1[1]!=null
                String zatiak2[] = zatiak1[1].split(" ");
                luzera=zatiak2.length;
                i=1;
                while(i<=luzera-1) {
                    w.getEstekenZerrenda().add(wz.bilatuId(Integer.parseInt(←
                        zatiak2[i])));
                    i++;
                }
            }
        }
    } catch (Exception e){
        e.printStackTrace();
    } finally { // Finally hay da fitxeroa ixteko zerbait txarto badoa.
        try{

```

```

        if( null != fr ){
            fr.close();
        }
    } catch (Exception e2){
        e2.printStackTrace();
    }
}

private void fitxeroaSortu() throws IOException{
    //post: Web-orrien zerrenda fitxategitan gordetzen du
    String ruta= "resources\\fitxeroBerria.txt";
    File fitxeroa= new File(ruta);
    BufferedWriter bw;
    if(fitxeroa.exists()) {
        bw = new BufferedWriter(new FileWriter(fitxeroa));
    } else {
        bw = new BufferedWriter(new FileWriter(fitxeroa));
    }
    bw.close();
}

public void fitxeroaEguneratu() throws IOException {
    FileWriter fitxero= null;
    PrintWriter pw=null;
    fitxeroaSortu();
    try {
        fitxero= new FileWriter("resources\\fitxeroBerria.txt");
        pw=new PrintWriter(fitxero);
        Collection<Web> c= this.webMapa.values();
        ArrayList<Web> list= new ArrayList<>(c);
        // Hash mapeko balioak arrayList<Web>n gorde

        for(int i=0; i<list.size();i++) {
            pw.println(list.get(i).getUrlWeb()+" " +i);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            if (null!=fitxero) {
                fitxero.close();
            }
        } catch (Exception e2) {
            e2.printStackTrace();
        }
    }
}

public Web bilatuUrl(String pUrl){
    // Pre: Url string bat sartzen da.
    // Post: Bueltatzen du Web objektua url hori duena bere url atributuan.
    return this.webMapa.get(pUrl);
}

public Web bilatuId(int pId){
    Web ema=null;
    if(pId>=0 && pId<wZerrenda.size()) {
        ema=wZerrenda.get(pId);
    }
    return ema;
}

public void gehitu (Web pWeb){
    this.wZerrenda.add(pWeb);
    //urlZerrenda.add(pWeb.getUrlWeb()); // String zerrendan gehitzeko
    this.webMapa.put(pWeb.getUrlWeb(), pWeb); // HashMapan ere gehitzeko
}

```

```

public void ezabatu (String pUrl){
    webMapa.remove(pUrl);
}

public ArrayList<Web> irteerakoEstekak(String pUrl){
    // post: web-orri baten izena emanda, estekatzen dituen web-orriak ↔
    // itzultzen ditu.
    Web w=bilatuUrl(pUrl);
    ArrayList<Web>a=new ArrayList<Web>();
    if(w!=null) {
        a=w.getEstekenZerrenda();
    }
    return a;
}

public ArrayList<Web> word2Webs(String pGako){
    // pre: parametroa gako-hitza bat da
    // post: gako-hitza daukaten web-orriak itzultzen ditu eta web-orrien ↔
    // zerrendan sartzen du
    ArrayList<Web> ema = new ArrayList<Web>();
    Iterator<Web>itr = this.getIteradorea();
    Web w;
    while (itr.hasNext()){
        w = itr.next();
        if (w.getUrlWeb().contains(pGako)) {
            w.getGakoZerrenda().add(GakoHitzZerrenda.getNireGakoHitzZerrenda())↔
            .bilatuHitza(pGako));
            ema.add(w);
        }
    }
    return ema;
}

// public ArrayList<Hitza> web2Words(String pUrl){
//     // post: web-orrian agertzen diren gako-hitza itzultzen ditu
//     Web w = this.bilatuUrl(pUrl);
//     return w.getGakoZerrenda();
// }

private void quickSort(Web[] zerrenda, int hasiera, int bukaera) {
    if(bukaera-hasiera>0) {
        int indizeaZatiketa=zatiketa(zerrenda,hasiera,bukaera);//error
        quickSort(zerrenda,hasiera,indizeaZatiketa-1); //error
        quickSort(zerrenda,indizeaZatiketa+1,bukaera);
    }
}

private int zatiketa(Web[] lista, int i, int f) {
    Web lag1=lista[i];
    lista[i]= lista[i+(f-i)/2];
    lista[i+(f-i)/2]=lag1;
    Web lag= lista[i];
    int ezker=i;
    int eskuin=f;
    while(ezker<eskuin) {
        while(lista[ezker].getUrlWeb().compareToIgnoreCase(lag.getUrlWeb())<=0↔
            && ezker<eskuin)
            ezker++;
        while(lista[eskuin].getUrlWeb().compareToIgnoreCase(lag.getUrlWeb())↔
            >0)
            eskuin--;
        if(ezker<eskuin)
            swap(lista,ezker,eskuin);
    }
    lista[i]=lista[eskuin];
    lista[eskuin]=lag;
    return eskuin;
}

```

```

private void swap(Web[] lista, int ezker, int eskuin) {
    Web temp=lista[ezker];
    lista[ezker]=lista[eskuin];
    lista[eskuin]=temp;
}

public Web[] listaOrdenatuta() {
    Collection<Web> c= this.webMapa.values();
    Web[] zerrenda= c.toArray(new Web[c.size()]);
    quickSort(zerrenda,0,zerrenda.length-1);
    return zerrenda;
}

// Metodos para probar a ver si funciona.

public int getSize(){
    return this.wZerrenda.size();
}

public int getHSize(){
    return this.webMapa.size();
}
}

```

5.2 Web

```

package pack1Enuntziatu1;

import java.util.*;

public class Web {
    // atributuak
    private int id;
    private ArrayList<Web>estekenZerrenda; // Hemen gordeko dira web bakoitzak ↔
        dituen esteken web-ak
    private ArrayList<Hitza>gakoZerrenda; // Hemen gordeko dira web bakoitzak ↔
        URL-AN DITUEN GAKOAK
    private String url;

    // eraikitzailea
    public Web (int pId, String pUrl) {
        this.id= pId;
        this.url = pUrl;
        this.estekenZerrenda= new ArrayList<Web>() ;
        this.gakoZerrenda= new ArrayList<Hitza>();
    }

    // getters
    public String getUrlWeb(){
        return this.url;
    }

    public int getId() {
        return this.id;
    }

    // gainontzeko metodoak
    private Iterator<Web> getIteradorea(){
        return this.estekenZerrenda.iterator();
    }
}

```

```

public ArrayList<Hitza> getGakoZerrenda(){
    return this.gakoZerrenda;
}

public ArrayList<Web> getEstekenZerrenda(){
    return this.estekenZerrenda;
}

public void webInprimatu() {
    System.out.println(" ");
    System.out.println("    Sartu duzun web orrialdearen informazioa honako da↵
        :");
    System.out.println(" url:      "+this.url);
    System.out.println(" indizea: "+this.id);
    System.out.println(" ");
}

public void estekaZerrendaInprimatu() {
    System.out.println(" ");
    Iterator<Web> itr= this.getIteradorea();
    Web w=null;
    while(itr.hasNext()) {
        w=itr.next();
        System.out.println( w.id + " " + w.url);
    }
}
}

```

5.3 GakoHitzZerrenda

```

package pack1Enuntziatu1;

import java.util.HashMap;
//import java.util.Iterator;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
public class GakoHitzZerrenda {

    // atributuak
    private static GakoHitzZerrenda nireGakoHitzZerrenda = null;
    //private ArrayList<Hitza>ghZerrenda; // Ez dakigu behar dugun ala ez

    private HashMap<String, Hitza> gakoMapa; // HashMap bat sortu behar dugu Key ⇔
        katea eta Value = Hitza objektua

    // eraikitzailea
    private GakoHitzZerrenda() {
        //this.ghZerrenda = new ArrayList<Hitza>(); //gako array-a sortu
        this.gakoMapa = new HashMap<String,Hitza>(); //gako mapa sortu
    }

    // gainontzeko metodoak

    public static GakoHitzZerrenda getNireGakoHitzZerrenda() {
        if(nireGakoHitzZerrenda == null){
            nireGakoHitzZerrenda = new GakoHitzZerrenda();
        }
        return nireGakoHitzZerrenda;
    }

    public Hitza bilatuHitza(String pGakoa) {
        return gakoMapa.get(pGakoa);
    }
}

```

```

public void fitxeroaKargatu(File pFitxeroaWords) throws FileNotFoundException ←
{
    // Pre: Fitxeroa jasotzen du words.txt

    // Post: GakoHitzZerrenda betetzen du, HashMapa betetzen du.
    FileReader fr = new FileReader (pFitxeroaWords);
    BufferedReader b = new BufferedReader(fr);

    String fila;

    try
    {
        while ((fila=b.readLine())!=null)
        {
            Hitza h = new Hitza(fila);
            //this.ghZerrenda.add(h);
            this.gakoMapa.put(h.getKatea(), h);
        }
    }catch(Exception e){
        e.printStackTrace();
    }finally{ // Finally hau da fitxeroa ixteko zerbait txarto badoa.
        try{
            if( null != fr ){
                fr.close();
            }
        }catch (Exception e2){
            e2.printStackTrace();
        }
    }
}
}

```

5.4 Hitza

```

package pack1Enuntziatu1;

import java.util.ArrayList;
import java.util.Iterator;

public class Hitza {
    // atributuak
    private String katea;
    private ArrayList<Web> wLista;

    // eraikitzailea
    public Hitza (String pKatea){
        this.katea = pKatea;
        this.wLista = new ArrayList<Web>();
    }

    // getters
    public ArrayList<Web> getWebLista(){
        return this.wLista;
    }

    public String getKatea(){
        return this.katea;
    }

    // metodoak

    private Iterator<Web> getIteradorea(){
        return this.wLista.iterator();
    }
}

```

```

public void gehituWebBat(Web pWeb){
    this.wLista.add(pWeb);
}

public void beteWebLista (String pKatea){
    this.wLista = WebZerrenda.getNireWebZerrenda().word2Webs(pKatea);
}

public void gakoWebInprimatu() {
    System.out.println(" ");
    Iterator<Web> itr= this.getIteradorea();
    Web w=null;
    while(itr.hasNext()) {
        w=itr.next();
        System.out.println( w.getId() + " " + w.getUrlWeb());
    }
}
}

```

5.5 Teklatua

```

package pack1Enuntziatu1;

import java.util.Scanner;

public class Teklatua {

    // atributuak
    private static Teklatua nireTeklatua = null;
    private Scanner sc;

    // eraikitzailea
    private Teklatua() {
        this.sc=new Scanner(System.in);
    }

    // gainontzeko metodoak
    public static Teklatua getNireTeklatua() {
        if(nireTeklatua == null){
            nireTeklatua = new Teklatua();
        }
        return nireTeklatua;
    }

    public int irakurriAukera(int pNundik, int pNora){
        int emaitza = -1;
        boolean denaOndo=false;
        do {
            String str = sc.nextLine();
            try{
                emaitza = Integer.parseInt(str);
                if( emaitza >pNora || emaitza<pNundik){
                    throw new NumberFormatException();//sartzen duen balioa ez du
                }
                denaOndo=true;
            }catch (NumberFormatException e) { System.out.println("Sar ezazu
                zenbaki baliogarri bat:");
            }
        }while(!denaOndo);
        return emaitza;
    }

    public String irakurriString(){

```

```

        String mezua=this.sc.nextLine();
        return mezua;
    }
}

```

5.6 HasieratuPraktika

```

package pack1Enuntziatu1;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
//import java.io.*;
import java.util.*;

public class HasieratuPraktika {

    public static void main(String[] args) {

        File wordsFitxeroa = null;
        File webIndexFitxeroa = null;
        File webEstekaFitxeroa = null;

        wordsFitxeroa = new File ("resources\\words.txt");
        webIndexFitxeroa = new File ("resources\\index.txt");
        webEstekaFitxeroa = new File ("resources\\p1d-arcs-1-N.txt");

        GakoHitzZerrenda ghz = GakoHitzZerrenda.getNireGakoHitzZerrenda();
        WebZerrenda wz = WebZerrenda.getNireWebZerrenda();
        Teklatua tk= Teklatua.getNireTeklatua();
        Web w=null;
        String s= " ";
        boolean ondo= false;
        Hitza h=null;

        try {
            // Lehenik fitxeroen karga egiten dugu.
            ghz.fitxeroaKargatu(wordsFitxeroa);
            wz.indexFitxeroaKargatu(webIndexFitxeroa);
            wz.arcFitxeroaKargatu(webEstekaFitxeroa);

            System.out.println("");
            System.out.println("");
            System.out.println("");

        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        System.out.println("↵
        *****");
        System.out.println("↵
        *****");
        System.out.println("**
        WEB KUDEAKETA APLIKAZIOA ↵
        **");
        System.out.println("↵
        *****");
        System.out.println("↵
        *****");
        System.out.println("");
        System.out.println("");
        System.out.println("");

        do {
            Scanner aukerak = null;
            //aukerak pantailan inprimatzeko:

```



```

File aukerakTestuaFile = new File("resources\\aukerak.txt");
try {
    aukerak = new Scanner(aukerakTestuaFile);

} catch (FileNotFoundException e) {
    System.out.println("Ez da aurkitu fitxategia");
}

while(aukerak.hasNextLine()) {
    System.out.println(aukerak.nextLine());
}
aukerak.close();

int kasua = tk.irakurriAukera(1 , 8);
switch(kasua) {
case 1: //Web bat bilatu.
    ondo=false;
    do{
        System.out.println("    Sar ezazu bilatu nahi duzun web orriaren ←
            url-a:");
        s= tk.irakurriString();
        w= wz.bilatuUrl(s);
        if(w!= null) {
            w.webInprimatu();
            ondo=true;
        }
        else {
            System.out.println("    Sartu duzun web orria ez da existitzen←
                .");
        }
    }while(!ondo);
    break;

case 2: //Web orri bat txertatu.
    ondo=false;
    do{
        System.out.println("    Sar ezazu txertatu nahi duzun web orriaren←
            url-a:");
        s= tk.irakurriString();
        w= wz.bilatuUrl(s);
        if(w == null) {
            w= new Web(wz.getHSize()+1,s);
            wz.gehitu(w);
            ondo=true;
        }
        else {
            System.out.println("    Sartu duzun web orria jada existitzen ←
                da.");
        }
    }while(!ondo);
    System.out.println("    Txertaketa arrakastatsua izan da.");
    break;

case 3: // Web orri bat ezabatu.
    ondo=false;
    do{
        System.out.println("    Sar ezazu ezabatu nahi duzun web orriaren ←
            url-a:");
        s= tk.irakurriString();
        w= wz.bilatuUrl(s);
        if(w!= null) {
            wz.ezabatu(s);
            ondo=true;
        }
        else {
            System.out.println("    Sartu duzun web orria ez da existitzen←
                .");
        }
    }

```

```

    }
    }while(!ondo);
    System.out.println("    Ezabaketa arrakastatsua izan da.");
    break;

case 4: //Web bat bat estekatzen dituen web orrien zerrenda ikusi.
ondo=false;
do{
    System.out.println("    Sar ezazu web orriaren url-a:");
    s= tk.irakurriString();
    w= wz.bilatuUrl(s);
    if(w!= null) {
        System.out.println("    Honako hau da "+ s +" web orrialdeak ←
        estekatzen dituen web orrien zerrenda: ");
        w.estekaZerendaInprimatu();
        ondo=true;
    }
    else {
        System.out.println("    Sartu duzun web orria ez da existitzen←
        .");
    }
}while(!ondo);
break;

case 5: //Gakohitz bat sartu bere web orrien zerrenda ikusteko.
ondo=false;
do{
    System.out.println("    Sar ezazu gako-hitza:");
    s= tk.irakurriString();
    h= ghz.bilatuHitza(s);
    if(h!= null) {
        System.out.println("    Honako hau da "+ s +" gako-hitza duten←
        web orrien zerrenda: ");
        h.gakoWebInprimatu();
        ondo=true;
    }
    else {
        System.out.println("    Sartu duzun gako-hitza ez da ←
        existitzen.");
    }
}while(!ondo);
break;

case 6: //Web orrien zerrenda fitxeroa eguneratu.
try {
    wz.fitxeroaEguneratu();
    System.out.println("    Fitxeroa eguneratu da.");
} catch (IOException e) {
    e.printStackTrace();
}
break;

case 7: //Web zerrenda ordenatua lortu.
wz.listaOrdenatuta();
System.out.println("    Web orrien zerrenda ordenatu da.");
break;

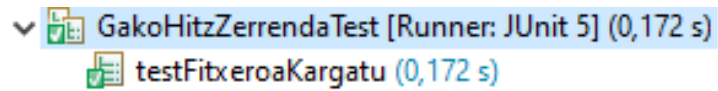
case 8: //Aplikaziotik irten.
System.out.println("←
*****");
System.out.println("←
*****");
System.out.println("*****AGUR←
*****");
System.out.println("←
*****");
System.out.println("←
*****");
System.out.println("←
*****");



```

```
        System.exit(0);
        break;
    }
} while (true);
} // -----_main_ amaiera -----
```

6 Emaiza enpirikoak

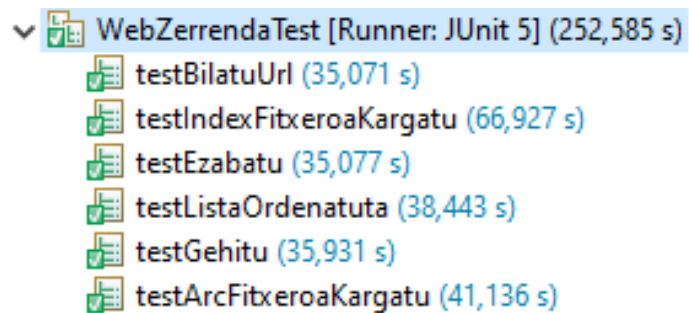
6.1 GakoHitzZerrenda










▼  GakoHitzZerrendaTest [Runner: JUnit 5] (0,172 s)
  testFitxeroaKargatu (0,172 s)

6.2 WebZerrenda

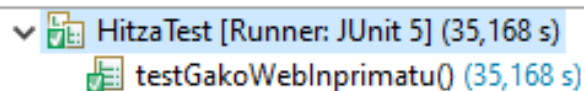
Denborak oso luzeak dira test-ak egiteko fitxeroak kargatu behar izan ditugulako setUp() atalean.





▼  WebZerrendaTest [Runner: JUnit 5] (252,585 s)
  testBilatuUrl (35,071 s)
  testIndexFitxeroaKargatu (66,927 s)
  testEzabatu (35,077 s)
  testListaOrdenatuta (38,443 s)
  testGehitu (35,931 s)
  testArcFitxeroaKargatu (41,136 s)

6.3 Hitza

Denborak oso luzeak dira test-ak egiteko fitxeroak kargatu behar izan ditugulako setUp() atalean.



▼  HitzaTest [Runner: JUnit 5] (35,168 s)
  testGakoWebInprimatu() (35,168 s)

7 Ondorioak

Hasteko, aipatzekoa da klaseen diseinuaren garrantzia. Klaseen diseinu on bat inplementazioa asko errazten du. PMOO irakasgaian zenbait datu egitura ikasi genituen, baina, klaseak jaso ditugun ahala datu egitura berriak ikasi ditugu. Gure lehenengo diseinua azkenean daukagun diseinuaren nahiko gertu zegoen arren, orain askoz eraginkorragoa da datu kopuru handiak kudeatu behar baititugu.

Aurrean aipatu dugunez, klasean asko ikasi dugu, ikasitakoaren artean *HashMap*-aren erabilera eta *split* metodoak ikasi ditugu. *HashMap*-aren erabilera gure programaren exekuzio denboran eragin handia dauka baita kostuan ere. Adibidez, *WebZerrenda* eta *GakoHitzZerrenda* hasieran inplementatzean, *ArrayList*-ak sortu egin genituen baina *HashMap*-ak ezagutzean, hauek erabiltzen errazagoak eta askoz ere eraginkorragoak zirela konturatu, eta gure datu egitura nagusiak *HashMap* bihurtu genituen.

Orain *ArrayList*-ak *HashMap*-ak erabiltzea baino kostu txikiagoa dutenean edota azken hauek erabili ezin direnean inplementatzen ditugu.

8 Erreferentziak

- **Oracle. HashMap (Java Platform SE 8)**
URL <https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html>
- **Mendialdua, Iñigo. Datu-Egiturak eta Algoritmoak: proiektua - Web-orri kopuru handia kudeatu.**
URL https://egela.ehu.eus/pluginfile.php/4148007/mod_resource/content/1/Praktika%202020%202021%20Eginkizuna1.pdf
- **Gojenola, Koldo. Algoritmoen analisisia.**
URL https://egela.ehu.eus/pluginfile.php/3532491/mod_resource/content/3/1-AlgoritmoenAnalisia.pdf
- **Lewis, John eta Chase, Joseph.**
Java software structures, third edition.
ISBN-13: 978-013-607858-6
- **Prieto, Ander eta San Juan, Kerman.**
Latex txantiloiarekin laguntza.