Softwarearen Ingeniaritza

2014-ko Uztailak 8

1. PATROIAK (1 puntu)

Ondorengo kodean:

a) Erabili diren patroiak identifikatu.
b) Zein da patroi bakoitzaren helburua?
c) Zein abantaila ematen digute?
d) Zein desabantaila izan dezake sistemak eboluzionatzen badu?
e) UML klase diagrama irudikatu, patroiak non dauden azalduz.

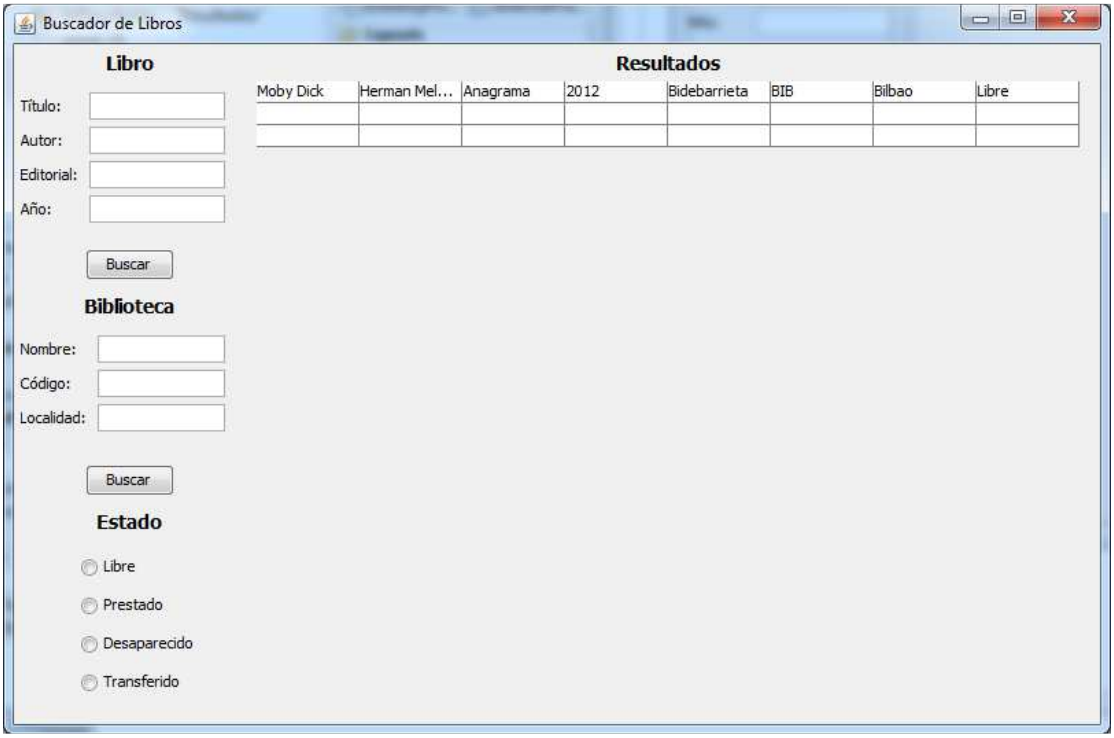
```
import java.util.ArrayList;
import java.util.Iterator;

public class Liburutegia {
	private static Liburutegia nLiburutegia = null;
	private static ArrayList<Liburua> liburuak;
	private Liburutegia() {};

	public static Liburutegia getInstance(){
		if (nLiburutegia == null) {
			nLiburutegia = new Liburutegia ();
		}
		return nLiburutegia;
	}

	public static void liburuakErakutsi (){
		Iterator<Liburua> itLiburuak = liburuak.iterator();
		while (itLiburuak.hasNext()){
			itLiburuak.next().liburuaErakutsi();
		}
	}
}

public class Liburua {

	private String izenburua;
	private String egilea;

	public Liburua (String pIzenburua, String pEgilea){
		izenburua = pIzenburua;
		egilea = pEgilea;
	}

	public void liburuaErakutsi(){
		System.out.println("Izenburua: " + izenburua);
		System.out.println("Egilea: " + egilea);
	}
}
```

2. INTERFAZE GRAFIKOAK (1 puntu)

Ondoren agertzen den leihoa ikusita:

a) Leihoa inplementatzeko erabili diren osagai eta edukiontziak identifikatu, azken hauentzat zein *layout* erabiliko zenukeen azalduz.

b) Leihoak funtzionalitate egokia izan dezan inplementatu beharko liratekeen gertaera kudeatzaileak identifikatu.



Interfaze Kudeatzailea	Adapterra	Metodo kudeatzailea
ActionListener		void actionPerformed (ActionEvent e)
ChangeListener		void stateChanged (ChangeEvent e)
ItemListener		void itemStateChanged (ItemEvent evt)
ListSelectionListener		void valueChanged (ListSelectionEvent evt)
MouseListener	MouseInputAdapter MouseAdapter	void mouseClicked(MouseEvent evt) void mouseEntered(MouseEvent evt) void mousePressed(MouseEvent evt) void mouseRelised(MouseEvent evt)
WindowListener	WindowAdapter	void windowClosed(WindowEvent evt) void windowClosing(WindowEvent evt) void windowActivated(WindowEvent evt) void windowOpened(WindowEvent evt)

3. PATROIAK (1 puntu)

Zainketa Intentsiboetako Unitatean (ZIU) ospitaleratutako gaixoen bihotz taupaden maiztasuna kontrolatzeko aplikazio bat inplementatu nahi da. Honetarako **MaiztasunKontrolatzailea** izeneko klase bat diseinatzen da, gaixoaren bihotz taupaden maiztasuna kontrolatuko dueña eskumuturrean daraman eskumuturrekoa erabiliz. Gainera, aplikazioak **MaiztasunPantaila** klasea erabiliz gaixoaren bihotz taupaden maiztasuna erakutsiko du gaixoaren ohe alboan dagoen pantailan. Balio hau maiztasuna aldatzen den bakoitzean eguneratuko da. Bukatzeko, momenturen batean maiztasuna minutuko 120 taupadatik igotzen bada, **MaiztasunAltuAbisua** klaseak soinu bat egingo du osasun langileei abisatzeko. Eskatzen da:

a) Zein patroi erabiliko zenituzke aplikazio hau inplementatzeko? Zergatik? Bere funtzionamendua azaldu zure aukera justifikatuz.

b) Aplikazio honen klase diagrama irudikatu, metodo garrantzitsuenak adieraziz.

c) Zure hitzekin eta laburki azaldu klaseen funtzionamendua eta noiz deitzen zaion metodo bakoitzari.

4. DISEINUA (2 puntu)

Bizkaiko Liburutegi Elkarteak bezeroei eskaintzen dizkien liburu aleen kudeaketa optimizatzeko aplikazio bat garatu nahi du. Honetarako klasean ikusitako Liburutegia ariketaren diseinua zabaltzea proposatu digute, probintziako liburutegi bakoitzeko bazkide eta liburuak dituen katalogo komun bat sortuz. Honela, **liburutegi** bakoitzak izena, kodea eta herria izango ditu**, liburu zerrenda** eta **bazkide** guztien informazioaz gain. **Liburu** bakoitzak bere izenburua, liburutegiak dituen ale kopurua eta horietatik zenbait dauden eskuragarri une bakoitzean izango ditu. Zabalkuntza honekin liburuak mailegatzeko prozesua aldatuko da, bazkide batek probintziako edozein liburutegitan dagoen liburu bat eskatu ahal izateko. Liburua eskatzen ari den liburutegian eskuragarri ez dagoenean, aplikazioak besteetan bilatuko du baduen liburutegiren bat aurkitu eta liburutegi horri utzi diezaion eskatzeko. Prozesu honek egun bat eramango du, jatorrizko liburutegian zeuden ale kopurua eta egoera eguneratuz eta data horrekin mailegua sortuz. Beste liburutegiek ez badute liburu horren alerik, aplikazioak `EzAlerikEskuragarri` salbuespena sortuko du.

Liburutegien arteko liburu transferentziak kontrolatzeko aplikazioak **transferentzia katalogo** bat izango du, bakoitzarentzat liburuaren kodea, transferentzia eskatu zuen liburutegiaren kodea, liburua daukan liburutegiaren kodea eta eskaeraren data dituelarik.

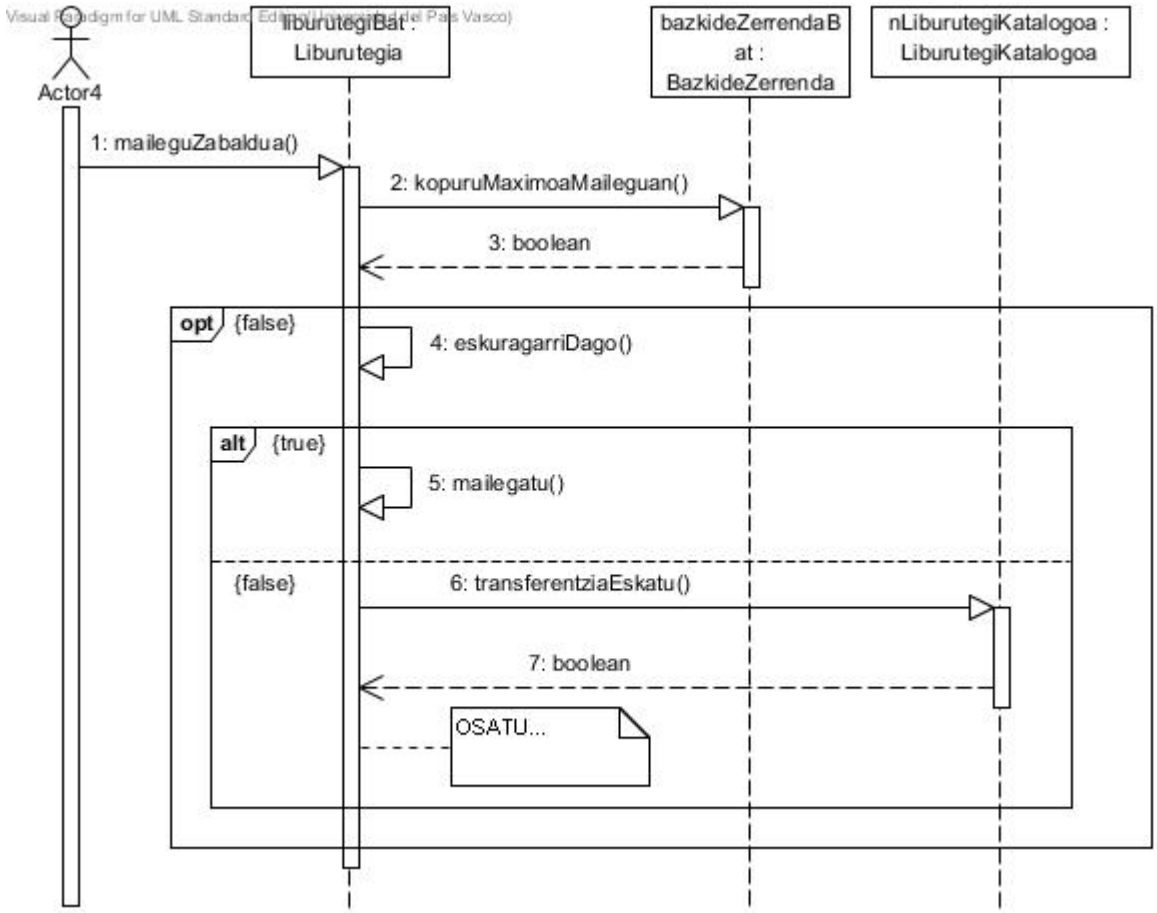
Informazio honetan oinarrituz, diseinatzea eskatzen den liburutegien arteko mailegu prozesua klasean ikusitako Liburutegia ariketaren aldaketa bat da, ondorengo pausoekin:

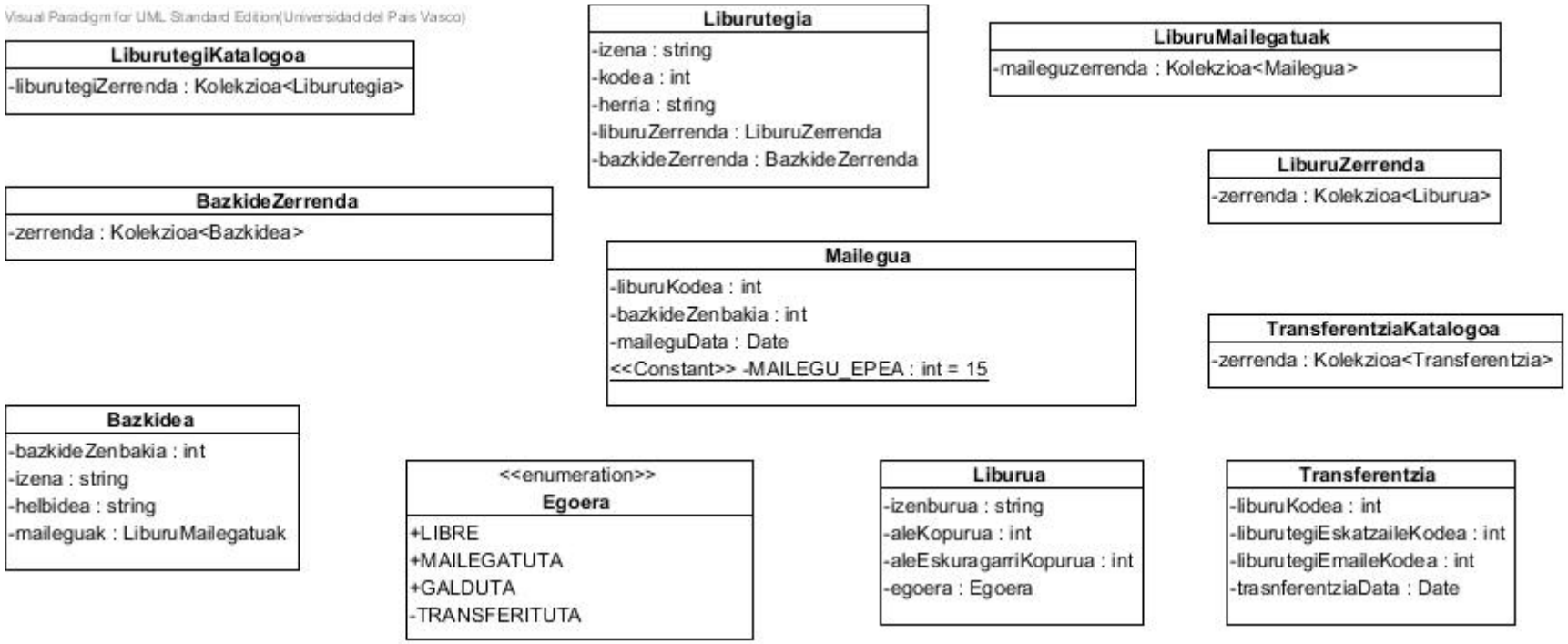
- Bazkideak mailegu kopuru maximoa bete ez duela egiaztatu.
- Eskatutako liburua liburutegian dagoen egiaztatu, hala izango ez balitz, transferitu diezaiokeen beste liburutegi bat bilatu.
- Liburuaren egoera, bazkidearen mailegu zerrenda eta liburutegien artean transferitutako liburu zerrendak eguneratu.
- Mailegua sortu.

Ondoren prozesuan parte hartzen duten klaseak agertzen dira, bere atributuekin.

Eskatzen da:

a) Irudikatutako klaseen konstruktoreak eta prozesua egin ahal izateko metodoak zehaztu, bakoitzaren ikusgarritasuna eta sarrera eta irteera parametroak bere motekin adieraziz.
b) Aurretik azaldutako prozesua egiten duen `maileguZabaldua` metodoaren sekuentzia diagrama osatu eta `LiburutegiKatalogoa` klaseko `transferentziaEskatu` eta `Liburutegia` klaseko `mailegatu` metodoen diagramak egin.
c) Klaseen arteko dependentzia harremanak irudikatu.


Visual Paradigm for UML Standard Edition(Universidad del País Vasco)
Actor4
liburutegiBat : Liburutegia
bazkideZerrendaBat : BazkideZerrenda
nLiburutegiKatalogoa : LiburutegiKatalogoa
1: maileguZabaldua()
2: kopuruMaximoaMaileguan()
3: boolean
opt
{false}
4: eskuragarriDago()
alt
{true}
5: mailegatu()
{false}
6: transferentziaEskatu()
7: boolean
OSATU...



5. GENERIZITATEA (1 puntu)

Liburutegi katalogoko liburutegiak, beste liburutegietara transferituta dituzten ale kopuruaren arabera ordenatzen dituen metodoa inplementatu. Horretarako, klase egokian liburutegi bakoitzak egindako transferentzia kopurua kalkulatzen duen metodoa inplementatu beharko duzu. Bai metodo hau eta bai ordenatzeko balio duen metodoa JGA erabiliz inplementatu.

Sort - Class in net.sf.jga.algorithms
Adapts standard java Sort capabilities to the interfaces common to jga.
Sort() - Constructor for class net.sf.jga.algorithms.Sort
sort(T[]) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterable object over the sorted contents of the array.
sort(T[], Comparator<? super T>) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterable object over the sorted contents of the array, using the given comparator to determine ordering.
sort(Iterable<T>) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterable object over the sorted contents of the input.
sort(Iterable<T>, Comparator<? super T>) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterable object over the sorted contents of the input, using the given comparator to determine ordering.
sort(Iterator<T>) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterator object over the sorted contents of the input.
sort(Iterator<T>, Comparator<? super T>) - Static method in class net.sf.jga.algorithms.Sort
Returns an iterator object over the sorted contents of the input, using the given comparator to determine ordering.
sort(Iterable<? extends T>, TCollection) - Static method in class net.sf.jga.algorithms.Sort
Appends the sorted contents of the input to the output.
sort(Iterable<T>, Comparator<? super T>, TCollection) - Static method in class net.sf.jga.algorithms.Sort
Appends the sorted contents of the input to the output, using the given comparator to determine ordering.

Summarize - Class in net.sf.jga.algorithms
Algorithms that consume input and produce a single value.
Summarize() - Constructor for class net.sf.jga.algorithms.Summarize
average(Class<T>, Iterator<T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the average of the values.
count(T[], UnaryFunctor<T, Boolean>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of elements in the array for which the predicate is true
count(Iterable<? extends T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of elements in the input.
count(Iterable<? extends T>, T) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of times that the given value appears in the input.
count(Iterable<? extends T>, T, Comparator<? super T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of times that the given value appears in the input, using the given Comparator.
count(Iterable<? extends T>, UnaryFunctor<T, Boolean>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of elements in the input for which the predicate is true.
count(Iterator<? extends T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the number of elements in the iterator.
max(T[], Comparator<? super T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the largest T value in the input, as determined by the given comparator.
max(T[]) - Static method in class net.sf.jga.algorithms.Summarize
Returns the largest T value in the input.
max(T[], BinaryFunctor<T, T, T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the largest T value in the input, as determined by the given functor.
min(Iterator<? extends T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the smallest T value in the input.
min(Iterator<? extends T>, BinaryFunctor<T, T, T>) - Static method in class net.sf.jga.algorithms.Summarize
Returns the smallest T value in the input, as determined by the given functor.