Departament d'Enginyeria informàtica i Matemàtiques

Pràctica 3 Assignació de beques.

(Programació & Projectes de Sistemes Informàtics)

NOM DE L'EQUIP: Equipo_A

DIRECTOR: Francesc Recasens

EQUIP: Mihai Copil Servant

Jordi Borrell Vives

Daniel Zarco Lopez

Cristian Rodríguez Aranega

ASSIGNATURA: Programació

DATA: 11 / 12 / 2015

Índex

1.1 1.Introducció

En aquest treball, volem desenvolupar una aplicació per tal de poder resoldre el problema que ens han plantejat en classe, l'assignació de beques.

Per a realitzar aquesta tasca ens han assignat un tutor de 4rt, Francesc Recasens, ell ens guiarà en el disseny i desenvolupament de l'aplicació.

Per realitzar aquesta tasca ens hem organitzat en un equip de 4 persones: el Mihai, Jordi, Daniel i jo, Cristian. Ens hem repartit de la següent manera:

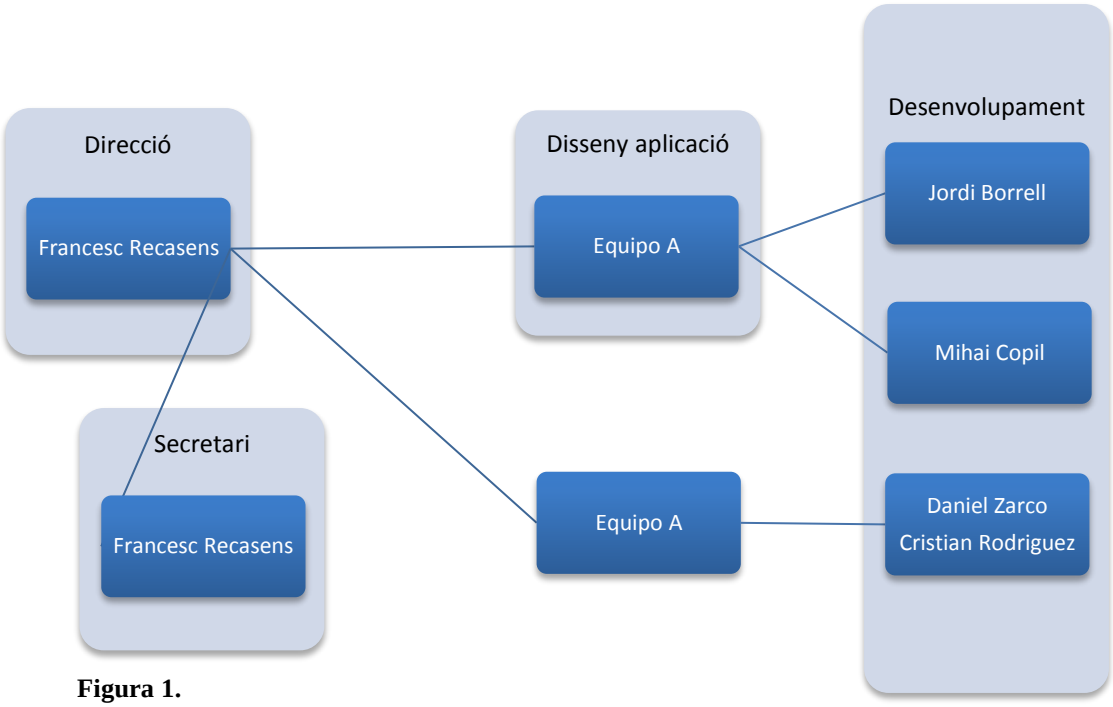

Figura 1.

Figura 1: Organigrama de l'equip de treball.

1.2 Especificacions del Projecte

En les especificacions del projecte podem diferenciar dues parts:

- Descripció de la practica:
- L'objectiu principal d'aquesta pràctica es aconseguir una aplicació funcional que sigui capaç de crear i assignar beques a partir d'unes dades que s'introdueixen per teclat.
- Per aceptar aquetes beques es faran a partir de les solicituts de beques que es tractaran de forma independent en el nostre programa.
- Además s'han de poder guardar les beques i les solicituts de beques en fitxers de text, per tal de poder carregar dades quan l'usuari ho desitgi.
- Per últim aquestes dades no es mostraran només per la consola de java, sinó que es realitzaran utilitzant la interfície gràfica proporcionada per java.
- Analisis del projecte:
- Per realitzar el projecte hem creat una classe pare, a partir de la cual tots els nostres mètodes heredaran d'ell (tots els tipus de beques hereden de beca)
- Hem creat un llista de solicituts i de beques, aquí es on hem creat tots el mètodes per tal d'afegir beca/solicitut, modificar beca...Es a dir totes les funcionalitats que demana l'enunciat.
- Además hem creat una excepció pròpia en la cual comprovem si la nostra beca existeix o no.
- Per a facilitar el projecte hem pres aquestes decisions:
- -Per assignar una beca consultem quin alumne té la nota més alta, si la solicitut es aceptada aquesta s'envia a la llista de solicituts.
- -Per tal de facilitar l'escritura de fitxers demanem al usuari el nombre de graus a la que estan destinades les beques. Així evitem problemes relacionats amb els strings en el fitxer.

1.3 Disseny

Aquest programa s'encarrega de administrar les beques i solicituts que crea un usuari i les asigna automàticament. Aquestes beques i solicituts es guarden en un fitxer de text per a que

l'usuari les pugui consultar quan ell vulgui. Per arribar aquest objectiu hem creat dues llistes on en una guardem les beques i en l'altre guardaré les solicituts.

Per dur a terme aquest problema l'hem desglossat en diversos apartats:

Apartat 1: Creació de les clases bàsiques, aquí trobem la classe pare Beca, d'aquesta heredaran totes les nostres beques secundaries (Erasmus+, Isep, ErasmusMundus i MOU). En aquestes clases bàsiques definim les propietats de cadascun dels objectes utilitzats en la nostra practica.

Apartat 2: Creació de les llistes de beques i solicituts: En aquest mètodes hem creat totes les funcionalitats que es demanen a l'enunciat de la practica.

```
public int consultarDotacioErasmusPlus(int codi) throws NoTrobat {
        int pos;

        pos = getBeca(codi);
        if(pos==-1) throw new NoTrobat();
        if (llista[pos] instanceof ErasmusPlus)
                return llista[pos].getDotacioMensual();
        else
                return 0;
}
```

Figura 2: Exemple de una de les funcionalitats del programa.

Hem implementat al main les funcionalitats per tal d'evitar errors, com ara la funció try and catch per tal d'estalviar problemes a l'hora de que l'usuari introdueix dades.

Apartat 3: Creació dels mètodes de lectura i escriptura de fitxers. En aquest mètodes hem modificat lleugerament les instruccions de com guardar les beques en l'arxiu, guardant també el nombre d'ensenyaments en el document. No obstant s'ha pogut dur a terme totes les funcionalitats opcionals d'aquest apartat.

```
public void exportarBeques(){
```

```
            try{
BufferedWriter escriptura=new BufferedWriter(new FileWriter("BequesGuardar.txt"));
                        for(int i=0; i<numBeques; i++){
                                Beca b = llista[i];
                                escriptura.write(b.toStringFitxer());
                                escriptura.newLine();
                        }
                        escriptura.close();
                }
catch(IOException e){
                System.out.println("No es pot escriure");
                }

        }
```

Figura 3: Exemple de una de les funcionalitats per tal de exportar beques.

Apartat 4: Interficie gràfica: En aquest apartat hem desenvolupat tota la GUI del programa, ha sigut realitzada a partir de la interfície que proporciona java. Hem fet una implementació sencilla, a partir de 4 botons tenim accés a totes les funcions que demana l'enunciat. Les dades que proporcionem a l'usuari es mostren a la consola i no en una finestra independent. Com exemple us mostro la capçalera del nostre mètode que mostra el menú per la interfície gràfica.

```
public class Finestra extends JFrame {

        private static final long serialVersionUID = 1L;
        private JLabel etiqueta=new JLabel("Menu Beques");
        private JButton[] botons=new JButton[4];

        private JPanel panell=new JPanel();
```

```
private JPanel entrada=new JPanel();
```

Figura 4: Exemple del codi per generar la interfície gràfica principal.

Per ultim especificare totes les decisions que hem pres com equip per desenvolupar la practica:

1-L'assignacio de beques es fa a partir de la nota més alta, no es te en compte cap factor més a l'hora d'assignarla.

2-Guardem el nombre d'ensenyaments com una variable de la beca, per facilitar la lectura i escriptura dels nostres fitxers. Si l'usuari indica que la beca esta destinada a dos ensenyaments aquest haurà de ficar els dos ensenyaments separats amb comes.

1.3.1 Disseny de Classes i Diagrames de Classes

Els diagrames de classes descriuen l'estructura del sistema mostrant les seves classes, atributs i relacions entre elles.

Hi ha diagrames estàndard per representar les classes, com els del llenguatge UML1 (Unified Modeling Language), que es el que utilitzarem nosaltres per tal de donar una representació simplificada del mètodes que inclou el programa.

1 L'UML o Llenguatge Unificat de Modelat (Unified Modeling Language, Llenguatge de Modelat Unificat) és un llenguatge de modelat de sistemes de software, és el més conegut i utilitzat en l'actualitat. Viquipèdia, 2013.

A continuació adjuntarem una imatge general del nostre diagrama de classes, en la qual podem observar la nostra distribució de classes i el contingut de cadascuna amb els seus mètodes i variables i com es relacionen entre elles.

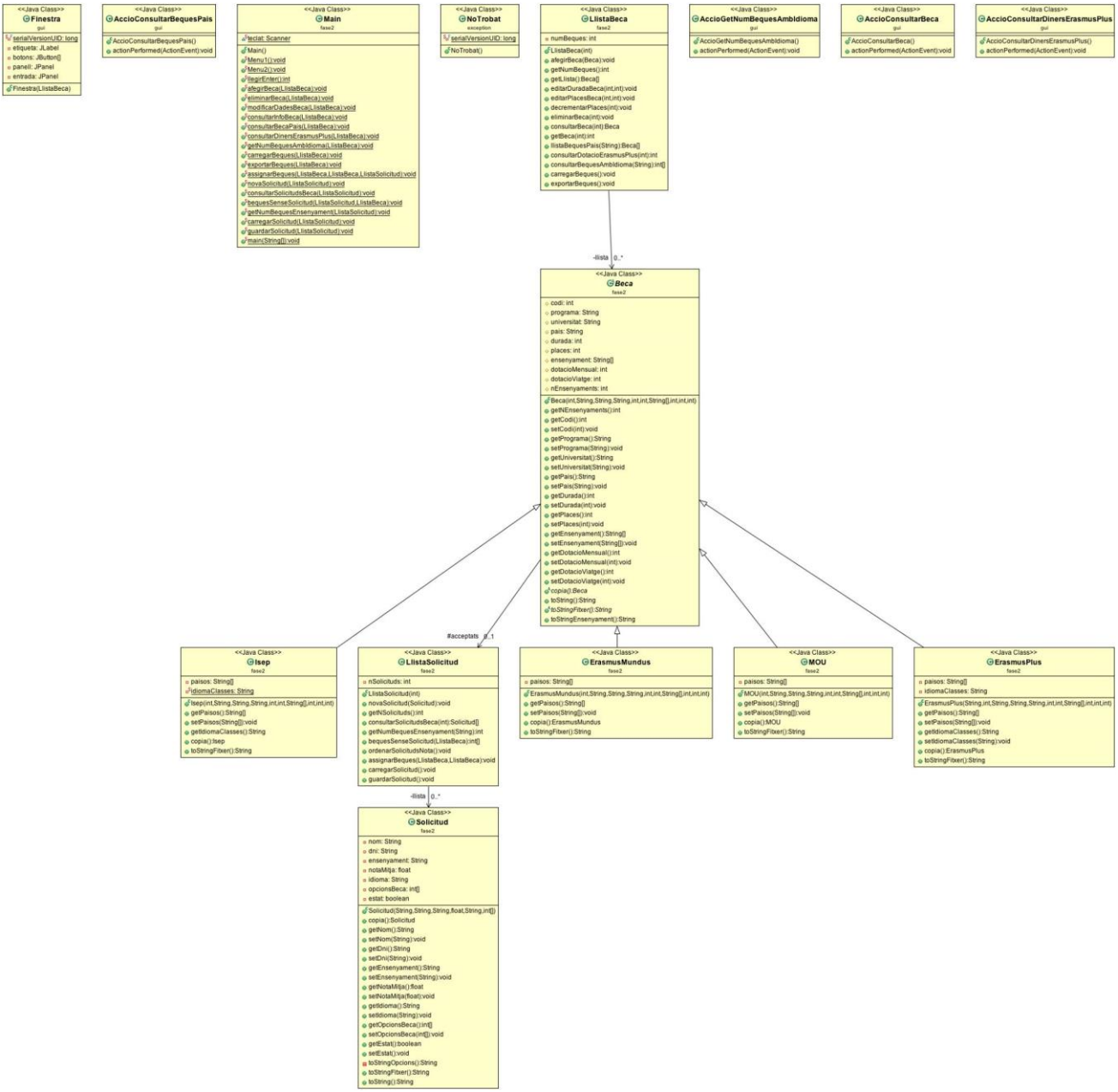

Figura 5. Diagrama de classes del programa.

1.3.2 .Disseny de les funcionalitats

-Funcionalitat per a les consultes: Un cop l'usuari ha introduït el codi de la beca o solicitut a consultar l'algoritme farà una cerca en les llistes i intentarà retornar la beca o solicitut demanada. Si aquesta no existeix s'enviara un missatge d'error per pantalla.

-Funcionalitat per comprovar que el numero que demanem per teclat es un numero i que aquest no sigui una dada incorrecte. A partir dels try and catch hem dissenyat una funció que s'encarrega de demanar a l'usuari un numero i el valida.

-Funcionalitat d'assignació de beques. Un cop introduïdes les beques el programa les asigna a partir de les notes mes altes. No es té en compte cap informació més.

-Funcionalitat d'eliminar beca. Se li demanarà a l'usuari el codi de la beca que desitja eliminar, si aquesta beca o solicitut existeix s'eliminara del sistema. Si no existeix es mostrara un misatje d'error.

-Funcionalitat per escriure i carregar en fitxer: En aquesta funcionalitat hem fet una petita modificació on hem guardat el nombre d'ensenyaments en la classe Beca, per tal d'evitar errors a l'hora de guardar en fitxer. Un cop introduïda una nova beca, si es vol guardar, es crida al mètode d'escriptura de fitxer i la imprimeix en el document txt. Com a toquen per separar cada element hem utilitzat les " , ".

1.3.3 Disseny de la Interfície

L'aplicació dissenyada per nosaltres inclou una interfície gràfica per facilitar l'ús del programa i fer possible que l'aplicació pugui ser utilitzada sense el requeriment de tenir algun coneixement d'us del eclipse. Per aconseguir això hem implementat una interfície gràfica composta per una finestra i uns botons que accionen les diferents opcions(mètodes) que pot realitzar el programa; aquestes son les 4 rutines de consulta de beques segons una informació que introduirà l'usuari. L'usuari un cop iniciada l'aplicació l'únic que haurà de fer, a part de seleccionar el menú al que vol tenir accés, és fer clic sobre l'opció que vol seleccionar i el programa iniciarà automàticament la rutina, li demanarà la informació que es necessita per fer les comprovacions i ensenyarà per pantalla el resultat.

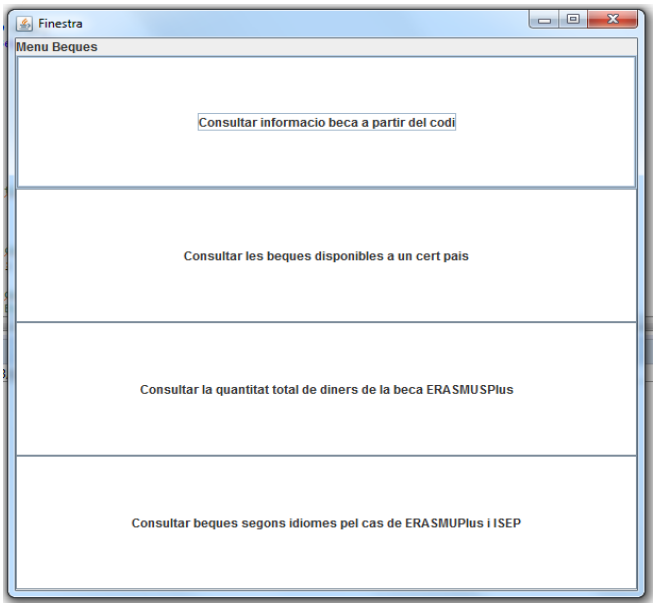

Figura 6. Interfície gràfica del menú de beques

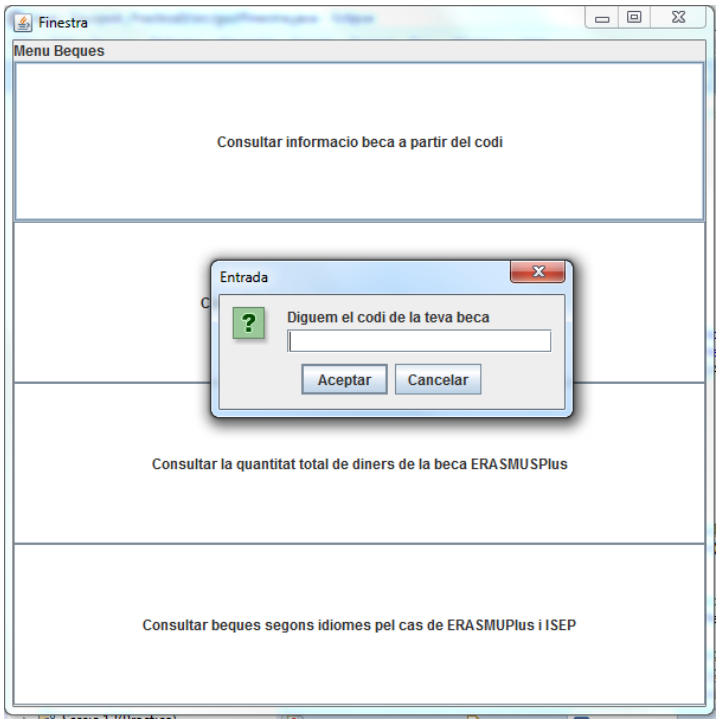

Figura 7. Interfície gràfica per interactuar amb l'usuari

```
A quin menu vols tenir acces?:

        1. Menu Beca

        2. Menu Solicituds
1
Beca [codi=5, programa=MOU, universitat=MiraboUniversity, pais=Spain, durada=5, places=5, ensenyament=[GEI], dotacioMensual=200, dotacioViatge=300]
```

Figura 8. Resultat de la rutina ensenyat per pantalla

Tal com podem observar en les fotos adjuntades anteriorment, la nostra interfície gràfica consta d'un menú amb 4 botons que estan enllaçats amb les diferents rutines per realitzar els mètodes que ens demanen i ensenyar la informació per pantalla; tal com podem observar, nomes hem decidit fer la part obligatòria de l'apartat gràfic, ja que no teníem temps suficient per implementar la resta de codi i hem decidit arreglar tots els errors i assegurar un bon funcionament del programa enlloc de fer tot lo opcional i que el programa contingui errors greus de funcionament.

1.4 Desenvolupament

En aquest apartat indicarem tot el referent a detalls específics de com s'ha implementat la practica pas a pas; paquets, classes, objectes, estructures de dades, algorismes, mètodes, funcions, etc. No inclourem el codi complet, sinó que inclourem petits trossos de codi que ens facilitaran entendre el funcionament del mètode.

Primerament començarem explicant com hem decidit assignar les beques. Per assignar les beques primerament hem generar una altra llista en la qual guardem les beques acceptades i posteriorment hem implementat un mètode que decrementi el numero de places de les beques pendents quan s'accepta una de les beques; seguidament hem fet 2 mètodes, un per ordenar totes les sol·licituds tenint en compte la nota mitja de cada sol·licitant i l'altre mètode per assignar les beques als sol·licitants tenint en compte la seva nota mitja i les seves prioritats(opcions).

```
public void decrementarPlaces(int codi) throws NoTrobat {
        int pos, places;

        pos = getBeca(codi);
        if (pos == -1)
                throw new NoTrobat();
        places = llista[pos].getPlaces();
```

Codi 1. Codi de decrementar Places.

```
public void ordenarSolicitudsNota() {
        int j = 0;
        int posNota = 0;
        float notaMax;
        Solicitud aux;
        for (int i = 0; i < nSolicituds; i++) {
            j = i;
            notaMax = 0;
            while (j < nSolicituds) {
                if (llista[j].getNotaMitja() > notaMax) {
                    notaMax = llista[j].getNotaMitja();
```

Codi 2. Codi per ordenar les sol·licituds.

```
public void assignarBeques(LlistaBeca bAcceptades, LlistaBeca bPendents) {
    int i = 0, j = 0, pos, pos2, codi = 0;
    Beca beca;
    int[] codis;

    ordenarSolicitudsNota();
    while (i < nSolicituds) {
        while (j < 3 && !llista[i].getEstat()) {
            codis = llista[i].getOpcionsBeca();
            codi = codis[j];
            pos = bPendents.getBeca(codi);
            if (pos != -1) {
                int[] opcions = new int[3];
                opcions[0] = llista[i].getOpcionsBeca()[j];
                opcions[1] = opcions[0];
                opcions[2] = opcions[0];
                llista[i].setOpcionsBeca(opcions);
                try {
                    beca = bPendents.consultarBeca(opcions[0]);
                    pos2 = bAcceptades.getBeca(opcions[0]);
                    if (pos2 == -1) {
                        beca.setPlaces(1);
                        bAcceptades.afegirBeca(beca);
                    } else
                        bAcceptades.consultarBeca(opcions[0]).setPlaces(
                                beca.getPlaces() + 1);
                    bPendents.decrementarPlaces(opcions[0]);
                    System.out
                            .println("S'ha acceptat la solicitud a nom de: "
                                    + llista[i].getNom()
                                    + " amb nota mitja: "
                                    + llista[i].getNotaMitja()
                                    + "\nSe li ha assignat la beca: "
                                    + bAcceptades.consultarBeca(opcions[0])
                                            .toString());
                    llista[i].setEstat();
                } catch (NoTrobat e) {
                }
            }
            j++;
        }
        j = 0;
        i++;
    }

}
```

Codi 3. Codi per assignar les beques tenint en compte certs factors.

Tot seguit analitzarem la manera en la que hem decidit separar els menús, nosaltres hem decidit fer dos menús separats, un per les beques i l'altre per a les solicituds; a mes a mes hem implementat un tercer menu que es el que s'encarregar de la interfície gràfica. L'usuari, al iniciar l'aplicació haurà de seleccionar al menu al que voldrà tenir accés(valor 1 o 2) i hi accedirà directament tenint accés a les diverses opcions que inclou cada menú per separat.

A continuació afegirem el codi d'una de les excepcions que hem decidit crear nosaltres per tal de evitar l'error que salta en cas de no trobar una beca.

```
package exception;


public class NoTrobat extends Exception {
        private static final long serialVersionUID =
1L; // Para que no aparezca



                // warning


        public NoTrobat() {
```

Codi 4. Codi de l'excepció.

Aquí podem observar el codi que hem creat per fer referencia a l'excepció de beca no trobada.

Tot seguit explicarem una mica com funciona el mètode d'afergir una beca; primerament demanem tota la informació que es necessita per generar una beca en general, i despres, tenint en compte el tipus de programa al qual la beca fa referencia demanarem una informació o una altra tot comprovant que la informació introduïda entra dintre d'uns paràmetres acceptats per la beca, si tot això es compleix, un cop acabat d'introduir totes les dades, es generara una beca del tipus establert.

```
        else
            System.out.println("No s'ha afegit correctament");
    }

    if (programa.equalsIgnoreCase("ERASMUSMundus")) {
        System.out.println("Introdueix la durada");
        durada = llegirEnter();
        System.out.println("Introdueix el numero de places");
        places = llegirEnter();
        System.out
                .println("A quants ensenyaments s'ofereix la beca en questio");
        nEnsenyaments = llegirEnter();
        String[] llistaensenyaments = new String[nEnsenyaments];
        for (int i = 0; i < nEnsenyaments; i++) {
            System.out
                    .println("Introdueix l'ensenyament on s'ofereix un per un");
            String ensenyament = teclat.next();
            llistaensenyaments[i] = ensenyament;
        }
        dotacioMensual = 500;
        if ((pais.equalsIgnoreCase("Georgia"))
                || (pais.equalsIgnoreCase("Azerbaitzan"))
                || (pais.equalsIgnoreCase("Armenia"))
                || (pais.equalsIgnoreCase("Ucrania"))
                || (pais.equalsIgnoreCase("Moldavia"))
                || (pais.equalsIgnoreCase("Bielorusia"))) {
            System.out.println("S'ha pogut introduir");
            ErasmusMundus beca = new ErasmusMundus(codi, programa,
                    universitat, pais, durada, places, llistaensenyaments,
                    dotacioMensual, 0, nEnsenyaments);
            llistaBeca.afegirBeca(beca);

        } else
            System.out.println("No s'ha pogut introduir");

    }
}
```

Codi 5. Part del codi d'afegir Beca.

Seguidament explicarem el mètode carregar Beques:

El fitxer des del cual carreguem les beques té una beca guardada en cada línia de text, amb els atributs de cada beca(depenen del programa) separats per comes. Per tal de facilitar la lectura del fitxer, s'ha d'introduïr el nombre d'ensenyaments abans d'especificar a quins ensenyaments va dirigida la beca.

Quan es carreguen beques el mètode llegeix línia a línia i afegeix a la llista de beques cada beca corresponent a cada una de les línies del fitxer.

```
public void carregarBeques() {
    try {
        Scanner lectura = new Scanner(new File("BequesCarregar.txt"));
        String idiomaClasses = null;
        int dotacioMensual = 0;
        int dotacioViatge = 0;
        lectura.useDelimiter(",");
        int nEnsenyaments = 0;
        while (lectura.hasNext()) {
            int codi = Integer.parseInt(lectura.next());
            String programa = lectura.next();
            String universitat = lectura.next();
            String pais = lectura.next();
            if (programa.equalsIgnoreCase("ErasmusPlus")) {
                idiomaClasses = lectura.next();
            }
            dotacioMensual = Integer.parseInt(lectura.next());
            dotacioViatge = Integer.parseInt(lectura.next());
            int durada = Integer.parseInt(lectura.next());
            int places = Integer.parseInt(lectura.next());
            nEnsenyaments = Integer.parseInt(lectura.next());
            String[] ensenyament = new String[nEnsenyaments];
            for (int i = 0; i < nEnsenyaments; i++) {
                ensenyament[i] = lectura.next();
            }
            if (programa.equalsIgnoreCase("ErasmusPlus")) {
                ErasmusPlus beca = new ErasmusPlus(idiomaClasses, codi,
                        programa, universitat, pais, durada, places,
                        ensenyament, dotacioMensual, dotacioViatge,
                        nEnsenyaments);
                afegirBeca(beca);
            }
            if (programa.equalsIgnoreCase("MOU")) {
                MOU beca = new MOU(codi, programa, universitat, pais,
                        durada, places, ensenyament, dotacioMensual,
                        dotacioViatge, nEnsenyaments);
                afegirBeca(beca);
            }
            if (programa.equalsIgnoreCase("Isep")) {
                Isep beca = new Isep(codi, programa, universitat, pais,
                        durada, places, ensenyament, dotacioMensual,
                        dotacioViatge, nEnsenyaments);
                afegirBeca(beca);
            }
            if (programa.equalsIgnoreCase("ErasmusMundus")) {
                ErasmusMundus beca = new ErasmusMundus(codi, programa,
                        universitat, pais, durada, places, ensenyament,
                        dotacioMensual, dotacioViatge, nEnsenyaments);
                afegirBeca(beca);
            }
```

Codi 6. Part del codi de carregar beques.

Finalment descriurem un dels mètodes de consulta referent a les beques:

En el mètode per a consultar les beques segons l'idioma, simplement demanem l'idioma del qual volem saber les beques que el tenen. I per obtenir el que volem recorrem la llista de beques i mirem de les beques Isep i ErasmusPlus, que són les úniques que tenen idioma, si tenen l'idioma que hem demanat, si aquest és el cas guardem el codi de la beca. Un cop recorreguda tota la llista de beques es retorna un array de int en el què hi ha els codis de les beques que tenen aquest idioma que posteriorment es printejarà per pantalla.

```
*/
public int[] consultarBequesAmbIdioma(String idioma) throws NoTrobat {
    int llistaIdiomes[] = new int[numBeques];
    int numTrobats = 0;

    for (int i = 0; i < numBeques; i++) {
        if (llista[i] instanceof ErasmusPlus) {
            if (((ErasmusPlus) llista[i]).getIdiomaClasses()
                    .equalsIgnoreCase(idioma))
                llistaIdiomes[numTrobats++] = llista[i].getCodi();
        } else if (llista[i] instanceof Isep) {
            if (((Isep) llista[i]).getIdiomaClasses().equalsIgnoreCase(
                    idioma))
                llistaIdiomes[numTrobats++] = llista[i].getCodi();
        }
    }
    if (numTrobats == 0)
        throw new NoTrobat();
    int[] retornar = new int[numTrobats];
    for (int i = 0; i < numTrobats; i++)
        retornar[i] = llistaIdiomes[i];
    return retornar;
}
```

Codi 7. Part del codi de consultar beques segons l'idioma de la beca.

1.5 Avaluació

Comentari del director:

El director de l'equip ha complit la seva tasca d'anar executant el seu joc de proves durant tot el procés de desenvolupament. Durant les reunions, revisàvem el codi fet durant la setmana i es feien unes proves bàsiques per veure si calia una revisió o no. D'altra banda, també es tenia el control sobre qui feia cada part, i fins a quin punt era la seva complexitat per tal que tots els programadors tinguessin coneixement de tot el projecte. Les proves eren senzilles: Control d'I/O, i control sobre les assignacions i consultes sobre les beques. Més endavant es controlen els accesos als fitxers i si les excepcions es controlen bé o no. Per acabar, la GUI es comprova en temps d'execució. S'ha revisat també la optimització i estructuració de codi (packages i encapsulament) per tal que pugui ser analitzat per a altres persones sense que sigui dificultós. Reutilitzament de funcions, implementació en funcions noves el codi que pot repetir-se... Es considera que anar avaluant de manera setmanal el codi és molt util per tal de trobar els errors i solventar-los al mateix moment, que no fer-les al final de tot i tenir un llistat enorme d'errors.

Hem afegit un arxiu amb les beques i sol·licituds ja carregades per tal de estalviar el temps que es trigaríem en afegir les beques i sol·licituds manualment. A partir d'aquests arxius hem realitzat totes les nostres proves. Sempre que es respectin l'ordre de les dades que es vagin demanant per pantalla el programa funcionarà correctament.

A demés per garantir el funcionament del programa hem afegit algunes excepcions com ara:

1- Menú inicial → Si l'usuari introdueix no introdueix un número, sortirà un missatge d'error en el qual l'obligarà a introduir un altre cop la informació demanada.
2- En qualsevol menú, gracies a la funció de comprovarEnter si l'usuari no introdueix una dada correcte es tornarà a demanar una dada correcte.
3- Comprovar si una beca o una sol·licitud que es busca existeix o no. Aquest joc de proves el fem per tal de que el nostre programa no busqui en una llista de 0 elements i deixi de funcionar.
4- Per tal d'estalviar problemes a l'usuari cada cop que aquest busca una de les beques utilitzem *'equals.ignore.case'* per tal de que si l'usuari no introdueix una beca amb majúscules no la trobi per aquest motiu.

1.6 Conclusions

Durant la realització d'aquest treball hem assolit un bon nivell de programació en java, a utilitzar les eines de treball en grup com el trello i a treballar en compenetració amb els diversos programadors del grup. Gracies a tots aquests factors, hem aconseguit treure sense cap error i força ràpid la fase 1 i 2; no obstant no hem aconseguit fer totes les parts opcionals de la practica referent a la fase 3(no hem pogut realitzar la part opcional de gràfics). Com a equip hem treballat força bé, no tots teníem el mateix nivell a l'hora de programar, però ens hem ajudat entre tots per tal de poder aconseguir finalitzar la practica.

El nostre principal objectiu, a part de aprovar la practica, per suposat, ha sigut que un cop finalitzada la practica poguéssim dir que hem après a programar en java a un nivell força acceptable i que hem assolit un dels principals objectius de la practica, aconseguir treballar en un bon ambient amb la resta de companys del grup i juntament amb els consells del director anar resolent els problemes que sorgien a la practica amb la millor eficàcia possible, tal com haurem de fer un cop acabéssim la carrera, en el món laboral.

1.7 Recursos Utilitzats

API de java → On hem consultat informació de les funcionalitats de java.

Moodle URV→ On hem après tots els coneixements de POO

Stackoverflow→Hem trobat la solució a alguns problemes puntual

1.8 Annexes

1.8.1 Annex. Implementació

Codi Beca:

```
package fase2;
import java.util.Arrays;

public abstract class Beca {
	protected int codi;
	protected String programa;
	protected String universitat;
	protected String pais;
	protected int durada;
	protected int places;
	protected String[]ensenyament;
	protected int dotacioMensual;
	protected int dotacioViatge;
	protected int nEnsenyaments;
	protected LlistaSolicitud acceptats;
```

```
	/**
	 * Constructor base de beca
	 * @param codi codi de la beca
	 * @param programa programa de la beca
	 * @param universitat universitat que ofereix la beca
	 * @param pais pais on esta situada la universitat
	 * @param durada durada de la beca
	 * @param places places que admet la beca
	 * @param ensenyament ensenyaments que poden demanar la beca
	 * @param dotacioMensual ajuda mensual que ofereix la beca
	 * @param dotacioViatge ajuda unica que ofereix la beca
	 * @param nEnsenyaments nombre d'ensenyaments que poden demanar la
beca
	 */
	public Beca(int codi, String programa, String universitat, String
pais, int durada, int places, String[]ensenyament, int dotacioMensual,int
dotacioViatge,int nEnsenyaments){
		this.codi = codi;
		this.programa=programa;
		this.universitat=universitat;
		this.pais=pais;
		this.durada=durada;
		this.places=places;
		this.ensenyament=ensenyament;
		this.dotacioMensual=dotacioMensual;
		this.dotacioViatge=dotacioViatge;
		this.nEnsenyaments=nEnsenyaments;
	}

	public int getNEnsenyaments(){
		return nEnsenyaments;
	}
	public int getCodi() {
		return codi;
	}

	public void setCodi(int codi) {
		this.codi = codi;
	}

	public String getPrograma() {
		return programa;
	}

	public void setPrograma(String programa) {
		this.programa = programa;
	}

	public String getUniversitat() {
		return universitat;
	}

	public void setUniversitat(String universitat) {
		this.universitat = universitat;
	}

	public String getPais() {
		return pais;
	}

	public void setPais(String pais) {
```

```
		this.pais = pais;
	}

	public int getDurada() {
		return durada;
	}

	public void setDurada(int durada) {
		this.durada = durada;
	}

	public int getPlaces() {
		return places;
	}

	public void setPlaces(int places) {
		this.places = places;
	}

	public String[] getEnsenyament() {
		return ensenyament;
	}

	public void setEnsenyament(String[] ensenyament) {
		this.ensenyament = ensenyament;
	}

	public int getDotacioMensual() {
		return dotacioMensual;
	}

	public void setDotacioMensual(int dotacioMensual) {
		this.dotacioMensual = dotacioMensual;
	}

	public int getDotacioViatge() {
		return dotacioViatge;
	}

	public void setDotacioViatge(int dotacioViatge) {
		this.dotacioViatge = dotacioViatge;
	}

/**
 *
 * @return retorna una copia de la beca
 */
	public abstract Beca copia();

	@Override
	public String toString() {
		return "Beca [codi=" + codi + ", programa=" + programa
				+ ", universitat=" + universitat + ", pais=" +
pais
				+ ", durada=" + durada + ", places=" + places
				+ ", ensenyament=" + Arrays.toString(ensenyament)
				+ ", dotacioMensual=" + dotacioMensual + ",
dotacioViatge="
				+ dotacioViatge + "]";
	}
	/**
```

```
	 *
	 * @return string que es guarda al fitxer
	 */
	public abstract String toStringFitxer();
	/**
	 *
	 * @return ensenyaments separats per ","
	 */
	public String toStringEnsenyament(){
		String string= new String();
		for(int i=0;i<ensenyament.length;i++){
			string=string+","+ensenyament[i];
		}
		return(string);
	}

}
```

Codi ErasmusMundus:

```
package fase2;

	public class ErasmusMundus extends Beca {
		private String[] paisos;

		public ErasmusMundus(int codi, String programa, String
universitat,
				String pais, int durada, int places, String[]
ensenyament,
				int dotacioMensual, int dotacioViatge,int
nEnsenyaments) {
			super(codi, programa, universitat, pais, durada,
places, ensenyament,
					dotacioMensual,
dotacioViatge,nEnsenyaments);
		}

		public String[] getPaisos() {
			return paisos;
		}

		public void setPaisos(String[] paisos) {
			this.paisos = paisos;
		}

		/**
		 * Copia de la beca, per poder ficarla a la llista
		 */
		public ErasmusMundus copia() {
			ErasmusMundus novaErasmusMundus = new
ErasmusMundus(codi, programa,
					universitat, pais, durada, places,
ensenyament, dotacioMensual,
					dotacioViatge,nEnsenyaments);
			return (novaErasmusMundus);
		}

		public String toStringFitxer() {
```

```
                  return (codi + "," + programa + "," + universitat + ","
+ pais + ","
                              + dotacioMensual + "," + dotacioViatge +
"," + durada + ","
                              + places +nEnsenyaments+
toStringEnsenyament()+",");
            }
      }
```

Codi ErasmusPlus:

```
package fase2;


public class ErasmusPlus extends Beca {

      private String[] paisos;
      private String idiomaClasses;

      public ErasmusPlus(String idiomaClasses, int codi, String programa,
                  String universitat, String pais, int durada, int
places,
                  String[] ensenyament, int dotacioMensual, int
dotacioViatge,int nEnsenyaments) {
            super(codi, programa, universitat, pais, durada, places,
ensenyament,
                        dotacioMensual, dotacioViatge,nEnsenyaments);
            this.idiomaClasses = idiomaClasses;
      }

      public String[] getPaisos() {
            return paisos;
      }

      public void setPaisos(String[] paisos) {
            this.paisos = paisos;
      }

      public String getIdiomaClasses() {
            return idiomaClasses;
      }

      public void setIdiomaClasses(String idiomaClasses) {
            this.idiomaClasses = idiomaClasses;
      }

      /**
       * Copia de la beca, per poder ficarla a la llista
       */
      public ErasmusPlus copia() {
            ErasmusPlus novaErasmusPlus = new ErasmusPlus(idiomaClasses,
codi,
                        programa, universitat, pais, durada, places,
ensenyament,
                        dotacioMensual, dotacioViatge,nEnsenyaments);
            return (novaErasmusPlus);
      }

      public String toStringFitxer() {
```

```
            return (codi + "," + programa + "," + universitat + "," +
pais + ","
                        + idiomaClasses + "," + dotacioMensual + "," +
dotacioViatge
                        + "," + durada + "," + places +nEnsenyaments+
toStringEnsenyament()+",");
      }

}
```

Codi Isep:

```
package fase2;


public class Isep extends Beca {

      private String[] paisos;
      private static String idiomaClasses="angles";

      public Isep(int codi, String programa, String universitat, String
pais,
                  int durada, int places, String[] ensenyament, int
dotacioMensual,
                  int dotacioViatge,int nEnsenyaments) {
            super(codi, programa, universitat, pais, durada, places,
ensenyament,
                        dotacioMensual, dotacioViatge,nEnsenyaments);
      }

      public String[] getPaisos() {
            return paisos;
      }

      public void setPaisos(String[] paisos) {
            this.paisos = paisos;
      }

      public String getIdiomaClasses() {
            return idiomaClasses;
      }

      /**
       * Copia de la beca, per poder ficarla a la llista
       */
      public Isep copia() {
            Isep novaIsep = new Isep(codi, programa, universitat, pais,
durada,
                        places, ensenyament, dotacioMensual,
dotacioViatge,nEnsenyaments);
            return (novaIsep);
      }

      public String toStringFitxer() {
            return (codi + "," + programa + "," + universitat + "," +
pais + ","
                        + dotacioMensual + "," + dotacioViatge + "," +
durada + ","
```

```
                        + places +nEnsenyaments +
toStringEnsenyament()+",");
      }
}
```

Codi LlistaBeca:

```
package fase2;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

import exception.NoTrobat;

public class LlistaBeca {

      private int numBeques = 0;
      private Beca[] llista;

      public LlistaBeca(int numBeques) {
            llista = new Beca[numBeques];
            numBeques = 0;
      }

      /**
       * Metode per afegir beca
       *
       * @param p
       *            beca copiada
       */
      public void afegirBeca(Beca p) {
            if (numBeques < llista.length) {
                  llista[numBeques] = p.copia();
                  numBeques++;
            }
      }

      public int getNumBeques() {
            return (numBeques);
      }

      public Beca[] getLlista() {
            return llista;
      }

      /**
       * Metode per editar la durada de la beca
       *
       * @param codi
       *            de la beca a editar
       * @param durada
       *            duracio de la beca
       * @throws NoTrobat
       *             quan la beca no existeix o no es troba
       */
```

```
	public void editarDuradaBeca(int codi, int durada) throws NoTrobat
{
		int pos;

		pos = getBeca(codi);
		if (pos == -1)
			throw new NoTrobat();
		llista[pos].setDurada(durada);
	}

	/**
	 * Metode per editar les places de la beca
	 *
	 * @param codi
	 *            de la beca
	 * @param places
	 *            numero de places disponibles
	 * @throws NoTrobat
	 *             quan la beca no existeix o no es troba
	 */
	public void editarPlacesBeca(int codi, int places) throws NoTrobat
{
		int pos;

		pos = getBeca(codi);
		if (pos == -1)
			throw new NoTrobat();
		llista[pos].setPlaces(places);
	}

	/**
	 * Metode per decrementar el nombre de places d'una beca pendent, i
si nomes
	 * quedava una plaça s'elimina la beca, significa que s'ha assignat
la beca
	 *
	 * @param codi
	 *            de la beca que se li decrementen les places
	 * @throws NoTrobat
	 *             quan la beca no existeix o no es troba
	 */
	public void decrementarPlaces(int codi) throws NoTrobat {
		int pos, places;

		pos = getBeca(codi);
		if (pos == -1)
			throw new NoTrobat();
		places = llista[pos].getPlaces();
		if (places > 1)
			llista[pos].setPlaces(places - 1);
		else
			eliminarBeca(codi);
	}

	/**
	 * Metode per eliminar una beca a partir del seu codi
	 *
	 * @param codi
	 *            id de la beca
	 * @throws NoTrobat
	 *             quan la beca no existeix o no es troba
```

```
 */
public void eliminarBeca(int codi) throws NoTrobat {
      int pos;
      pos = getBeca(codi);
      if (pos == -1)
            throw new NoTrobat();
      for (int i = pos; i < numBeques - 1; i++)
            llista[i] = llista[i + 1];

      numBeques--;
      llista[numBeques] = null;
}

/**
 * Metode per consultar beca
 *
 * @param codi
 *            ID de la beca
 * @throws NoTrobat
 *             quan la beca no existeix o no es troba
 * @return la beca demanada
 */
public Beca consultarBeca(int codi) throws NoTrobat {
      int pos;

      pos = getBeca(codi);
      if (pos == -1)
            throw new NoTrobat();
      else
            return (llista[pos]);
}

/**
 * Metode per trobar beca
 *
 * @param codi
 *            ID de la beca
 * @return trobada o no trobada
 */
public int getBeca(int codi) {
      int i, pos;
      pos = 0;
      i = 0;
      boolean trobat;
      trobat = false;
      while ((i < numBeques) && (!trobat)) {
            if (codi == (llista[i].getCodi())) {
                  trobat = true;
                  pos = i;
            }
            i++;
      }

      if (trobat) {
            return pos;
      } else {
            return -1;
      }

}
```

```
/**
 * Metode per buscar una beca segons el pais destí
 *
 * @param pais
 *            pais de desti
 * @throws NoTrobat
 *             quan la beca no existeix o no es troba
 * @return beques amb el mateix destí
 */

public Beca[] llistaBequesPais(String pais) throws NoTrobat {
      int i = 0;
      int j = 0;
      Beca[] beques = new Beca[numBeques];

      while (i < numBeques) {
            if (llista[i].getPais().equalsIgnoreCase(pais)) {
                  beques[j] = llista[i].copia();
                  j++;
            }
            i++;
      }
      if (j != 0) {
            Beca[] retornar = new Beca[j];
            for (i = 0; i < j; i++) {
                  retornar[i] = beques[i];
            }
            return retornar;
      } else
            throw new NoTrobat();
}

/**
 * Consultar dotacio de la beca erasmus plus
 *
 * @param codi
 *            ID de la beca
 * @throws NoTrobat
 *             quan la beca no existeix o no es troba
 * @return dotacio de la beca
 */
public int consultarDotacioErasmusPlus(int codi) throws NoTrobat {
      int pos;

      pos = getBeca(codi);
      if (pos == -1)
            throw new NoTrobat();
      if (llista[pos] instanceof ErasmusPlus)
            return llista[pos].getDotacioMensual();
      else
            return 0;
}

/**
 *
 * @param idioma de les beques(erasmusplus o isep)
 * @return codis de les beques amb l'idioma introduït
 * @throws NoTrobat
 *             quan la beca no existeix o no es troba
 */
```

```
	public int[] consultarBequesAmbIdioma(String idioma) throws
NoTrobat {
		int llistaIdiomes[] = new int[numBeques];
		int numTrobats = 0;

		for (int i = 0; i < numBeques; i++) {
			if (llista[i] instanceof ErasmusPlus) {
				if (((ErasmusPlus) llista[i]).getIdiomaClasses()
						.equalsIgnoreCase(idioma))
					llistaIdiomes[numTrobats++] =
llista[i].getCodi();
			} else if (llista[i] instanceof Isep) {
				if (((Isep)
llista[i]).getIdiomaClasses().equalsIgnoreCase(
						idioma))
					llistaIdiomes[numTrobats++] =
llista[i].getCodi();
			}
		}
		if (numTrobats == 0)
			throw new NoTrobat();
		int[] retornar = new int[numTrobats];
		for (int i = 0; i < numTrobats; i++)
			retornar[i] = llistaIdiomes[i];
		return retornar;
	}

	/**
	 * Mètode per carregar les beques des d'un fitxer
	 */
	public void carregarBeques() {
		try {
			Scanner lectura = new Scanner(new
File("BequesCarregar.txt"));
			String idiomaClasses = null;
			int dotacioMensual = 0;
			int dotacioViatge = 0;
			lectura.useDelimiter(",");
			int nEnsenyaments = 0;
			while (lectura.hasNext()) {
				int codi = Integer.parseInt(lectura.next());
				String programa = lectura.next();
				String universitat = lectura.next();
				String pais = lectura.next();
				if (programa.equalsIgnoreCase("ErasmusPlus")) {
					idiomaClasses = lectura.next();
				}
				dotacioMensual =
Integer.parseInt(lectura.next());
				dotacioViatge = Integer.parseInt(lectura.next());
				int durada = Integer.parseInt(lectura.next());
				int places = Integer.parseInt(lectura.next());
				nEnsenyaments = Integer.parseInt(lectura.next());
				String[] ensenyament = new String[nEnsenyaments];
				for (int i = 0; i < nEnsenyaments; i++) {
					ensenyament[i] = lectura.next();
				}
				if (programa.equalsIgnoreCase("ErasmusPlus")) {
					ErasmusPlus beca = new
ErasmusPlus(idiomaClasses, codi,
```

```
                                          programa, universitat, pais,
durada, places,
                                          ensenyament, dotacioMensual,
dotacioViatge,
                                          nEnsenyaments);
                              afegirBeca(beca);
                        }
                        if (programa.equalsIgnoreCase("MOU")) {
                              MOU beca = new MOU(codi, programa,
universitat, pais,
                                          durada, places, ensenyament,
dotacioMensual,
                                          dotacioViatge, nEnsenyaments);
                              afegirBeca(beca);
                        }
                        if (programa.equalsIgnoreCase("Isep")) {
                              Isep beca = new Isep(codi, programa,
universitat, pais,
                                          durada, places, ensenyament,
dotacioMensual,
                                          dotacioViatge, nEnsenyaments);
                              afegirBeca(beca);
                        }
                        if (programa.equalsIgnoreCase("ErasmusMundus")) {
                              ErasmusMundus beca = new
ErasmusMundus(codi, programa,
                                          universitat, pais, durada,
places, ensenyament,
                                          dotacioMensual, dotacioViatge,
nEnsenyaments);
                              afegirBeca(beca);
                        }
                        lectura.nextLine();
                  }
                  lectura.close();
            } catch (FileNotFoundException e) {

                  System.out.println("El fitxer no existeix");
            }
      }

      /**
       * Mètode per guardar les beques en un fitxer
       */
      public void exportarBeques() {
            try {
                  BufferedWriter escriptura = new BufferedWriter(new
FileWriter(
                              "BequesGuardar.txt"));
                  for (int i = 0; i < numBeques; i++) {
                        Beca b = llista[i];
                        escriptura.write(b.toStringFitxer());
                        escriptura.newLine();
                  }
                  escriptura.close();
            } catch (IOException e) {
                  System.out.println("No es pot escriure");
            }

      }
}
```

Codi LlistaSolicitud:

```
package fase2;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

import exception.NoTrobat;

public class LlistaSolicitud {
	private int nSolicituds = 0;
	private Solicitud[] llista;

	public LlistaSolicitud(int capacitat) {
		llista = new Solicitud[capacitat];
		nSolicituds = 0;
	}

	/**
	 * Metode per afegir una nova solicitut
	 *
	 * @param s
	 *            nom de la beca a copiar
	 */
	public void novaSolicitud(Solicitud s) {
		if (nSolicituds < llista.length) {
			llista[nSolicituds] = s.copia();
			nSolicituds++;
		}
	}

	public int getNSolicituds() {
		return (nSolicituds);
	}

	/**
	 * Metode per consultar les solicituts de beca a una certa
destinació donant
	 * el codi de la beca
	 *
	 * @param codi
	 *            ID de la beca
	 * @return Solicituds
	 */
	public Solicitud[] consultarSolicitudsBeca(int codi) {
		int i = 0;
		int j = 0;
		Solicitud[] solicituds = new Solicitud[nSolicituds];

		while (i < nSolicituds) {
			int[] opcions = llista[i].getOpcionsBeca();
			if ((opcions[0] == codi) || (opcions[1] == codi)
					|| (opcions[2] == codi)) {
				solicituds[j] = llista[i];
				j++;
			}
			i++;
```

```
            }
            Solicitud[] retorna = new Solicitud[j];
            retorna = solicituds;
            return (retorna);
      }

      /**
       * Metode per consultar el numero de solicituds a un cert
ensenyament
       *
       * @param ensenyament
       *            tipus de ensenyament
       * @return num de solicituds
       */
      public int getNumBequesEnsenyament(String ensenyament) {
            int i = 0, res = 0;

            while (i < nSolicituds) {
                  if
(llista[i].getEnsenyament().equalsIgnoreCase(ensenyament))
                        res++;
                  i++;
            }
            return res;
      }

      /**
       * Metode per a consultar els codis de les beques que no tenen cap
solicitud
       *
       * @param llista2
       *            és la llista de beques
       * @return els codis de les beques sense solicitud
       */
      public int[] bequesSenseSolicitud(LlistaBeca llista2) {
            int i = 0, j = 0, z = 0;
            boolean trobat = false;
            int[] codis = new int[llista2.getNumBeques()];
            Beca[] beques = llista2.getLlista();

            while (i < llista2.getNumBeques()) {
                  while ((j < nSolicituds) && (!trobat)) {
                        if ((llista[i].getOpcionsBeca()[0] ==
beques[j].getCodi())
                                    || (llista[i].getOpcionsBeca()[1] ==
beques[j]
                                                .getCodi())
                                    || (llista[i].getOpcionsBeca()[2] ==
beques[j]
                                                .getCodi()))
                              trobat = true;
                        j++;
                  }
                  if (!trobat) {
                        codis[z] = beques[j - 1].getCodi();
                        z++;
                  } else
                        trobat = false;
                  i++;
            }
            int[] retorna = new int[z];
```

```
            retorna = codis;
            return (retorna);

      }

      /**
       * Metode per a ordenar la llista de solicituds de major nota mitja
a menor
       */
      public void ordenarSolicitudsNota() {
            int j = 0;
            int posNota = 0;
            float notaMax;
            Solicitud aux;
            for (int i = 0; i < nSolicituds; i++) {
                  j = i;
                  notaMax = 0;
                  while (j < nSolicituds) {
                        if (llista[j].getNotaMitja() > notaMax) {
                              notaMax = llista[j].getNotaMitja();
                              posNota = j;
                        }
                        j++;
                  }
                  aux = llista[posNota];
                  llista[posNota] = llista[i];
                  llista[i] = aux;
            }
      }

      /**
       * metode per assignar totes les beques possibles a les solicituds
segons la
       * seva nota mitja i les seves prioritats
       *
       * @param bAcceptades
       *            es la llista de beques que ja estan assignades a una
solicitud
       * @param bPendents
       *            es la llista de beques que estem mirant d'assignar
       */
      public void assignarBeques(LlistaBeca bAcceptades, LlistaBeca
bPendents) {
            int i = 0, j = 0, pos, pos2, codi = 0;
            Beca beca;
            int[] codis;

            ordenarSolicitudsNota();
            while (i < nSolicituds) {
                  while (j < 3 && !llista[i].getEstat()) {
                        codis = llista[i].getOpcionsBeca();
                        codi = codis[j];
                        pos = bPendents.getBeca(codi);
                        if (pos != -1) {
                              int[] opcions = new int[3];
                              opcions[0] = llista[i].getOpcionsBeca()[j];
                              opcions[1] = opcions[0];
                              opcions[2] = opcions[0];
                              llista[i].setOpcionsBeca(opcions);
                              try {
```

```
                                    beca =
bPendents.consultarBeca(opcions[0]);
                                    pos2 =
bAcceptades.getBeca(opcions[0]);
                                    if (pos2 == -1) {
                                          beca.setPlaces(1);
                                          bAcceptades.afegirBeca(beca);
                                    } else

      bAcceptades.consultarBeca(opcions[0]).setPlaces(
                                                      beca.getPlaces() +
1);

      bPendents.decrementarPlaces(opcions[0]);
                                    System.out
                                                .println("S'ha acceptat
la solicitud a nom de: "
                                                      +
llista[i].getNom()
                                                      + " amb nota
mitja: "
                                                      +
llista[i].getNotaMitja()
                                                      + "\nSe li ha
assignat la beca: "
                                                      +
bAcceptades.consultarBeca(opcions[0])

      .toString());
                                          llista[i].setEstat();
                                    } catch (NoTrobat e) {
                                    }
                              }
                              j++;
                        }
                        j = 0;
                        i++;
                  }

      }

      /**
       * Mètode per carregar les solicituds des d'un fitxer
       */
      public void carregarSolicitud() {
            try {
                  Scanner lectura = new Scanner(new
File("SolicitudsCarregar.txt"));
                  lectura.useDelimiter(",");
                  while (lectura.hasNext()) {
                        String nom = lectura.next();
                        String dni = lectura.next();
                        String ensenyament = lectura.next();
                        float notaMitja =
Float.parseFloat(lectura.next());
                        String idioma = lectura.next();
                        int[] opcionsBeca = new int[3];
                        opcionsBeca[0] =
Integer.parseInt(lectura.next());
                        opcionsBeca[1] =
Integer.parseInt(lectura.next());
```

```
                        opcionsBeca[2] =
Integer.parseInt(lectura.next());

                        Solicitud solicitud = new Solicitud(nom, dni,
ensenyament,
                                    notaMitja, idioma, opcionsBeca);
                        novaSolicitud(solicitud);
                  }
                  lectura.close();
            } catch (FileNotFoundException e) {

                  System.out.println("El fitxer no existeix");
            }
      }

      /**
       * Mètode per guardar les solicituds en un fitxer
       */
      public void guardarSolicitud() {
            try {
                  BufferedWriter escriptura = new BufferedWriter(new
FileWriter(
                              "SolicitudsGuardar.txt"));

                  for (int i = 0; i < nSolicituds; i++) {
                        Solicitud s = llista[i];
                        escriptura.write(s.toStringFitxer());
                  }
                  escriptura.close();
            } catch (IOException e) {
                  System.out.println("No es pot escriure");
            }
      }

}
```

Codi Main:

```
package fase2;
import java.util.*;

import exception.NoTrobat;
import gui.Finestra;

public class Main {

      static Scanner teclat = new Scanner(System.in);

      /**
       * Menu que es mostrara per pantalla referent a les Beques
       */

      public static void Menu1() {
            System.out.println("\n\nMENU BECA:");
            System.out.println("\n\t1. Afegir Beca");
            System.out.println("\t2. Eliminar Beca");
            System.out
                        .println("\t3. Editar dades beca(durada i nombre
de places) ");
```

```
            System.out.println("\t4. Consultar informacio beca a partir
del codi");
            System.out
                        .println("\t5. Consultar les beques disponibles a
un cert pais");
            System.out
                        .println("\t6. Consultar la quantitat total de
diners de la beca ERASMUSPlus");
            System.out
                        .println("\t7. Consultar beques segons idiomes
pel cas de ERASMUPlus i ISEP");
            System.out
                        .println("\t8. Carregar la informacio de les
noves beques des d'un fixer de text");
            System.out.println("\t9. Guardar la llista de beques en un
fixer");
            System.out.println("\t10. Assignar beques");
            System.out.println("\t11. Sortir");
            System.out.print("\n\t\t\tQuina opcio vols escollir?:\n");
      }

      /**
       * Menu que es mostrara per pantalla referent a les4 Solicituds
       */

      public static void Menu2() {
            System.out.println("\n\nMENU SOLICITUD:");
            System.out.println("\n\t1. Afegir Solicitut");
            System.out
                        .println("\t2. Consultar una solicitud a partir
del codi de beca");
            System.out
                        .println("\t3. Consultar quantes solicituds hi ha
d'alumnes d'un cert ensenyament");
            System.out
                        .println("\t4. Consultar quines beques no tenen
cap solicitud");
            System.out.println("\t5. Carregar des d'un fixer les
solicituds");
            System.out.println("\t6. Guardar solicituds en un fixer de
text");
            System.out.println("\t7. Sortir");
            System.out.print("\n\t\t\tQuina opcio vols escollir?:\n");
      }

      // *Metodes Beca
      public static int llegirEnter() {
            boolean comprobat = false;
            int valor = 0;
            while (!comprobat) {
                  try {
                        String s = teclat.next();
                        valor = Integer.parseInt(s);
                        comprobat = true;
                  } catch (NumberFormatException e) {
                        System.out.print("Error en el format del
codi\n");
                  }
            }
            return valor;
      }
```

```
      public static void afegirBeca(LlistaBeca llistaBeca) {
            int codi, durada, places, nEnsenyaments, dotacioMensual;
            System.out.println("Introdueix el codi de la beca");
            codi = llegirEnter();
            System.out
                        .println("Introdueix el programa per al qual vols
solicitar la beca");
            String programa = teclat.next();
            teclat.nextLine();
            System.out.println("Introdueix el nom de la universitat
desti");
            String universitat = teclat.nextLine();
            System.out.println("Introdueix el nom del pais \n  Països
permesos: \n *ISEP: EUA i Puerto Rico \n *ERASMUS MUNDUS: Geòrgia,
Azerbaitjan, Armènia, Ucraïna, Moldàvia, Bielorússia");
            String pais = teclat.nextLine();

            if (programa.equalsIgnoreCase("ERASMUSPlus")) {

                  System.out.println("Introdueix la durada");
                  durada = llegirEnter();
                  System.out.println("Introdueix el numero de places");
                  places = llegirEnter();
                  System.out
                              .println("A quants ensenyaments s'ofereix
la beca en questio");
                  nEnsenyaments = llegirEnter();
                  String[] llistaensenyaments = new
String[nEnsenyaments];
                  for (int i = 0; i < nEnsenyaments; i++) {
                        System.out
                                    .println("Introdueix l'ensenyament on
s'ofereix un per un");
                        String ensenyament = teclat.next();
                        llistaensenyaments[i] = ensenyament;
                  }
                  System.out.println("Introdueix una dotacio mensual");
                  dotacioMensual = llegirEnter();

                  while ((dotacioMensual < 200) && (dotacioMensual >
500)) {
                        System.out.println("Valor fora de l'interval ");
                        System.out.println("Introdueix una dotacio
mensual");
                        dotacioMensual = llegirEnter();

                  }
                  System.out.println("Valor dintre de l'interval");
                  System.out.println("Introdueix l'idioma");
                  String idioma = teclat.next();
                  ErasmusPlus beca = new ErasmusPlus(idioma, codi,
programa,
                              universitat, pais, durada, places,
llistaensenyaments,
                              dotacioMensual, 0, nEnsenyaments);
                  llistaBeca.afegirBeca(beca);
            }
            if (programa.equalsIgnoreCase("MOU")) {
                  System.out.println("Introdueix la durada");
                  durada = llegirEnter();
```

```
                System.out.println("Introdueix el numero de places");
                places = llegirEnter();
                System.out
                            .println("A quants ensenyaments s'ofereix
la beca en questio");
                nEnsenyaments = llegirEnter();
                String[] llistaensenyaments = new
String[nEnsenyaments];
                for (int i = 0; i < nEnsenyaments; i++) {
                      System.out
                                  .println("Introdueix l'ensenyament on
s'ofereix un per un");
                      String ensenyament = teclat.next();
                      llistaensenyaments[i] = ensenyament;
                }
                int dotacioViatge = 600;

                MOU beca = new MOU(codi, programa, universitat, pais,
durada,
                            places, llistaensenyaments, 0,
dotacioViatge, nEnsenyaments);
                llistaBeca.afegirBeca(beca);

          }

          if (programa.equalsIgnoreCase("ISEP")) {
                System.out.println("Introdueix la durada");
                durada = llegirEnter();
                System.out.println("Introdueix el numero de places");
                places = llegirEnter();
                System.out
                            .println("A quants ensenyaments s'ofereix
la beca en questio");
                nEnsenyaments = llegirEnter();
                String[] llistaensenyaments = new
String[nEnsenyaments];
                for (int i = 0; i < nEnsenyaments; i++) {
                      System.out
                                  .println("Introdueix l'ensenyament on
s'ofereix un per un");
                      String ensenyament = teclat.next();
                      llistaensenyaments[i] = ensenyament;
                }
                int dotacioViatge = 600;
                dotacioMensual = 500;
                if ((pais.equalsIgnoreCase("EUA"))
                            || (pais.equalsIgnoreCase("PuertoRico"))) {
                      System.out.println("S'ha afegit correctament");
                      Isep beca = new Isep(codi, programa, universitat,
pais, durada,
                                  places, llistaensenyaments,
dotacioMensual,
                                  dotacioViatge, nEnsenyaments);
                      llistaBeca.afegirBeca(beca);

                }

                else
                      System.out.println("No s'ha afegit
correctament");
          }
```

```
            if (programa.equalsIgnoreCase("ERASMUSMundus")) {
                  System.out.println("Introdueix la durada");
                  durada = llegirEnter();
                  System.out.println("Introdueix el numero de places");
                  places = llegirEnter();
                  System.out
                              .println("A quants ensenyaments s'ofereix
la beca en questio");
                  nEnsenyaments = llegirEnter();
                  String[] llistaensenyaments = new
String[nEnsenyaments];
                  for (int i = 0; i < nEnsenyaments; i++) {
                        System.out
                                    .println("Introdueix l'ensenyament on
s'ofereix un per un");
                        String ensenyament = teclat.next();
                        llistaensenyaments[i] = ensenyament;
                  }
                  dotacioMensual = 500;
                  if ((pais.equalsIgnoreCase("Georgia"))
                              || (pais.equalsIgnoreCase("Azerbaitzan"))
                              || (pais.equalsIgnoreCase("Armenia"))
                              || (pais.equalsIgnoreCase("Ucrania"))
                              || (pais.equalsIgnoreCase("Moldavia"))
                              || (pais.equalsIgnoreCase("Bielorusia"))) {
                        System.out.println("S'ha pogut introduir");
                        ErasmusMundus beca = new ErasmusMundus(codi,
programa,
                                    universitat, pais, durada, places,
llistaensenyaments,
                                    dotacioMensual, 0, nEnsenyaments);
                        llistaBeca.afegirBeca(beca);

                  } else
                        System.out.println("No s'ha pogut introduir");

            }

      }

      public static void eliminarBeca(LlistaBeca llistaBeca) {
            int codi;
            System.out
                        .println("Introdueix el identificador de la beca
que vols consultar");
            codi = llegirEnter();
            try {
                  llistaBeca.eliminarBeca(codi);
                  System.out.println("Beca borrada correctamente");
            } catch (NoTrobat e) {
                  System.out.println("Beca no trobada");
            }
      }

      public static void modificarDadesBeca(LlistaBeca llistaBeca) {
            int codi, durada, places;
            System.out
                        .println("Introdueix el identificador de la beca
que vols consultar");
            codi = llegirEnter();
```

```
            int retorna = llistaBeca.getBeca(codi);

            if (retorna != -1) {
                  System.out.println("Introdueix la nova durada de la
beca");
                  durada = llegirEnter();
                  System.out.println("Introdueix el nou nombre de
places");
                  places = llegirEnter();
                  try {
                        llistaBeca.editarDuradaBeca(codi, durada);
                        llistaBeca.editarPlacesBeca(codi, places);
                        System.out.println("S'ha modificat la beca");
                  } catch (NoTrobat e) {
                        System.out.println("Beca no trobada");
                  }
            }

            else

                  System.out
                              .println("La beca amb aquest codi
identificador no existeix");

      }

      public static void consultarInfoBeca(LlistaBeca llistaBeca) {
            int codi;
            System.out.println("Diguem el codi de la teva beca");
            codi = llegirEnter();
            try {
                  Beca beca2 = llistaBeca.consultarBeca(codi);
                  System.out.println(beca2);
            } catch (NoTrobat e) {
                  System.out.println("No s'ha trobat una beca");
            }
      }

      public static void consultarBecaPais(LlistaBeca llistaBeca) {
            System.out.println("Introdueix el pais que vols consultar");
            String pais = teclat.nextLine();
            try {
                  Beca[] beques = llistaBeca.llistaBequesPais(pais);
                  for (int i = 0; i < beques.length; i++) {
                        System.out.println(beques[i].toString());
                  }

            } catch (NoTrobat e) {
                  System.out.println("No s'ha trobat una beca");
            }

      }

      public static void consultarDinersErasmusPlus(LlistaBeca
llistaBeca) {
            int codi;
            System.out
                        .println("Consultar la quantitat total de diners
d'una beca ERASMUS+, introdueix el codi de la beca");
            codi = llegirEnter();
```

```
            try {

      System.out.println(llistaBeca.consultarDotacioErasmusPlus(codi));
            } catch (NoTrobat e) {
                  System.out.println("Beca no trobada");
            }

      }

      public static void getNumBequesAmbIdioma(LlistaBeca llistabeca) {

            System.out.println("Introdueix l'idioma que vols
consultar:");
            String idioma = teclat.nextLine();
            idioma = teclat.nextLine();
            try {
                  int[] beques =
llistabeca.consultarBequesAmbIdioma(idioma);
                  System.out
                              .println("Els codis de les beques amb
aquest idioma són: ");
                  for (int i = 0; i < beques.length; i++)
                        System.out.println(beques[i]);
            } catch (NoTrobat e) {
                  System.out.println("Beca no trobada");
            }

      }

      public static void carregarBeques(LlistaBeca llistaBeca) {
            llistaBeca.carregarBeques();
      }

      public static void exportarBeques(LlistaBeca llistaBeca) {
            llistaBeca.exportarBeques();
      }

      public static void assignarBeques(LlistaBeca bAcceptades,
                  LlistaBeca llistaBeca, LlistaSolicitud llistaSolicitud)
{
            llistaSolicitud.assignarBeques(bAcceptades, llistaBeca);
      }

      // *Metodes Solicitud

      public static void novaSolicitud(LlistaSolicitud llistaSolicitud) {
            String nom, dni, ensenyament, idioma;
            int opcions;
            float notaMitja;
            int[] opcionsBeca = new int[3];

            System.out.println("Introdueix el nom:");
            nom = teclat.nextLine();
            teclat.nextLine();
            System.out.println("Introdueix el dni:");
            dni = teclat.nextLine();
            System.out.println("Introduix l'ensenyament:");
            ensenyament = teclat.nextLine();
            System.out.println("Introduix la nota mitja:");
            notaMitja = teclat.nextFloat();
            System.out.println("Introduix l'idioma:");
```

```
            idioma = teclat.nextLine();
            teclat.nextLine();
            System.out.println("Introdueix el nivell del idioma");
            String nivell = teclat.nextLine();
            if ((nivell.equalsIgnoreCase("B2")) ||
(nivell.equalsIgnoreCase("C1"))
                        || (nivell.equalsIgnoreCase("C2"))) {
                  System.out.println("El teu nivell d'angles compleix els
requisits");

                  System.out
                              .println("Introdueix el numero de opcions
de beca que vols introduir");
                  opcions = llegirEnter();
                  int i = 0;
                  while ((i < opcions) && (opcions <= 3)) {
                        System.out.println("Introdueix opció de beca:");
                        opcionsBeca[i] = teclat.nextInt();
                        i++;

                  }

                  Solicitud solicitud = new Solicitud(nom, dni,
ensenyament,
                              notaMitja, idioma, opcionsBeca);
                  llistaSolicitud.novaSolicitud(solicitud);
            } else
                  System.out
                              .println("El teu nivell d'angles no
compleix els requisits");

      }

      public static void consultarSolicitudsBeca(LlistaSolicitud
llistaSolicitud) {
            int codi;
            System.out.println("Introdueix el codi de la beca que vols
consultar:");
            codi = llegirEnter();
            Solicitud[] retorna =
llistaSolicitud.consultarSolicitudsBeca(codi);
            for (int i = 0; i < retorna.length; i++) {
                  System.out.println(retorna[i].toString());
            }
      }

      public static void bequesSenseSolicitud(LlistaSolicitud
llistaSolicitud,
                  LlistaBeca llistaBeca) {

            int[] codis =
llistaSolicitud.bequesSenseSolicitud(llistaBeca);
            int i = 0;
            if (codis[i] != 0) {
                  System.out
                              .println("Els codis de les beques sense cap
solicitud són:\n");
                  while (i < codis.length) {
                        System.out.println(codis[i]);
                        i++;
                  }
```

```
            } else
                  System.out.println("No hi ha cap beca sense
solicitud");
      }

      public static void getNumBequesEnsenyament(LlistaSolicitud
llistaSolicitud) {

            System.out
                        .println("Introdueix les sigles de l'ensenyament
de la beca que vols consultar:");
            String ensenyament = teclat.next();
            int resultat =
llistaSolicitud.getNumBequesEnsenyament(ensenyament);
            System.out.println("El número de beques a aquest ensenyament
és: "
                        + resultat);
      }

      public static void carregarSolicitud(LlistaSolicitud
llistaSolicitud) {
            llistaSolicitud.carregarSolicitud();
      }

      // 6.Guardar solicituds en fitxer
      public static void guardarSolicitud(LlistaSolicitud
llistaSolicitud) {
            llistaSolicitud.guardarSolicitud();
      }

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            int opcio;
            int opcio2;
            int numBeques = 100;
            int capacitat = 100;
            LlistaBeca llistaBeca = new LlistaBeca(numBeques);
            LlistaSolicitud llistaSolicitud = new
LlistaSolicitud(capacitat);
            LlistaBeca bAcceptades = new LlistaBeca(numBeques);

            System.out.println("\n A quin menu vols tenir acces?:");
            System.out.println("\n\t1. Menu Beca");
            System.out.println("\n\t2. Menu Solicituds");
            System.out.println("\n\t3. GUI beca");
            System.out.println("\n\t4. Sortir");
            int n = teclat.nextInt();

            while (n != 4) {
                  while ((n < 0) || (n > 4)) {
                        System.out.println("\n Opcio incorrecta");
                        System.out.println("\n A quin menu vols tenir
acces?:");
                        System.out.println("\n\t1. Menu Beca");
                        System.out.println("\n\t2. Menu Solicituds");
                        System.out.println("\n\t3. GUI beca");
                        System.out.println("\n\t4. Sortir");
                        n = teclat.nextInt();

                  }
```

```
                  if (n == 1) {
                        Menu1();
                        opcio = teclat.nextInt();

                        while (opcio != 11) {
                              switch (opcio) {
                              case 1:
                                    afegirBeca(llistaBeca);
                                    break;
                              case 2:
                                    eliminarBeca(llistaBeca);
                                    break;
                              case 3:
                                    modificarDadesBeca(llistaBeca);
                                    break;
                              case 4:
                                    consultarInfoBeca(llistaBeca);
                                    break;
                              case 5:
                                    consultarBecaPais(llistaBeca);
                                    break;
                              case 6:

      consultarDinersErasmusPlus(llistaBeca);
                                    break;
                              case 7:
                                    getNumBequesAmbIdioma(llistaBeca);
                                    break;
                              case 8:
                                    carregarBeques(llistaBeca);
                                    break;
                              case 9:
                                    exportarBeques(llistaBeca);
                                    break;
                              case 10:
                                    assignarBeques(bAcceptades,
llistaBeca, llistaSolicitud);

                              }

                              Menu1();
                              opcio = teclat.nextInt();
                        }
                  }

                  if (n == 2) {
                        Menu2();
                        opcio2 = teclat.nextInt();

                        while (opcio2 != 7) {
                              switch (opcio2) {
                              case 1:
                                    novaSolicitud(llistaSolicitud);
                                    break;
                              case 2:

      consultarSolicitudsBeca(llistaSolicitud);
                                    break;
                              case 3:

      getNumBequesEnsenyament(llistaSolicitud);
```

```
                                    break;
                              case 4:
                                    bequesSenseSolicitud(llistaSolicitud,
llistaBeca);
                                    break;
                              case 5:
                                    carregarSolicitud(llistaSolicitud);
                                    break;
                              case 6:
                                    guardarSolicitud(llistaSolicitud);
                                    break;

                              }

                              Menu2();
                              opcio2 = teclat.nextInt();
                        }

                  }
                  if (n == 3) {
                        new Finestra(llistaBeca);
                  }
                  if (n != 4) {
                        System.out.println("\n A quin menu vols tenir
acces?:");
                        System.out.println("\n\t1. Menu Beca");
                        System.out.println("\n\t2. Menu Solicituds");
                        System.out.println("\n\t3. GUI beca");
                        System.out.println("\n\t4. Sortir");
                        n = teclat.nextInt();
                  }
            }
      }
}
```

Codi MOU:

```
package fase2;

public class MOU extends Beca {

      private String[] paisos;

      public MOU(int codi, String programa, String universitat, String
pais,
                  int durada, int places, String[] ensenyament, int
dotacioMensual,
                  int dotacioViatge,int nEnsenyaments) {
            super(codi, programa, universitat, pais, durada, places,
ensenyament,
                        dotacioMensual, dotacioViatge,nEnsenyaments);

      }

      public String[] getPaisos() {
            return paisos;
      }

      public void setPaisos(String[] paisos) {
```

```
            this.paisos = paisos;
      }

      public MOU copia() {
            MOU novaMou = new MOU(codi, programa, universitat, pais,
durada,
                        places, ensenyament, dotacioMensual,
dotacioViatge,nEnsenyaments);
            return (novaMou);
      }

      public String toStringFitxer() {
            return (codi + "," + programa + "," + universitat + "," +
pais + ","
                        + dotacioMensual + "," + dotacioViatge + "," +
durada + ","
                        + places +nEnsenyaments+
toStringEnsenyament()+",");
      }
}
```

Codi Solicitud:

```
package fase2;
import java.util.Arrays;


public class Solicitud {
      private String nom;
      private String dni;
      private String ensenyament;
      private float notaMitja;
      private String idioma;
      private int[] opcionsBeca;
      private boolean estat;

      /**
       * constructor de la solicitud de l'alumne
       * @param nom és el nom de l'alumne
       * @param dni és el dni de l'alumne
       * @param ensenyament és l'ensenyament que cursa l'alumne
       * @param notaMitja és la nota mitja de l'alumme
       * @param idioma és l'idioma que parla l'alumne
       * @param opcionsBeca és un vector amb les beques que vol solicitar
l'alumne
       */
      public Solicitud(String nom, String dni, String ensenyament,
                  float notaMitja, String idioma, int[] opcionsBeca) {
            this.nom = nom;
            this.dni = dni;
            this.ensenyament = ensenyament;
            this.notaMitja = notaMitja;
            this.idioma = idioma;
            this.opcionsBeca = opcionsBeca;
            estat=false;
      }
      /**
       *
       * @return retorna un copia de la solicitud
```

```
        */
    public Solicitud copia(){
      Solicitud novaSolicitud=new
Solicitud(nom,dni,ensenyament,notaMitja,idioma,opcionsBeca);
      return(novaSolicitud);
    }

      public String getNom() {
            return nom;
      }


      public void setNom(String nom) {
            this.nom = nom;
      }


      public String getDni() {
            return dni;
      }


      public void setDni(String dni) {
            this.dni = dni;
      }


      public String getEnsenyament() {
            return ensenyament;
      }


      public void setEnsenyament(String ensenyament) {
            this.ensenyament = ensenyament;
      }


      public float getNotaMitja() {
            return notaMitja;
      }


      public void setNotaMitja(float notaMitja) {
            this.notaMitja = notaMitja;
      }


      public String getIdioma() {
            return idioma;
      }


      public void setIdioma(String idioma) {
            this.idioma = idioma;
      }


      public int[] getOpcionsBeca() {
            return opcionsBeca;
      }
```

```
	public void setOpcionsBeca(int[] opcionsBeca) {
		this.opcionsBeca = opcionsBeca;
	}

	public boolean getEstat() {
		return estat;
	}


	public void setEstat() {
		estat=true;
	}
	private String toStringOpcions(){
		String s="";
		for(int i=0; i<opcionsBeca.length;i++) s+=opcionsBeca[i]+",";
		return (s.substring(0,s.length()-1));
	}
	public String toStringFitxer(){

	return(nom+","+dni+","+ensenyament+","+notaMitja+","+idioma+","+toS
tringOpcions()+",");
	}
	@Override
	public String toString() {
		return "Solicitud [nom=" + nom + ", dni=" + dni + ",
ensenyament="
				+ ensenyament + ", notaMitja=" + notaMitja + ",
idioma="
				+ idioma + ", opcionsBeca=" +
Arrays.toString(opcionsBeca)
				+ ", estat=" + estat + "]";
	}


}
```

Codi excepció No Trobat:

```
package exception;

public class NoTrobat extends Exception {
		private static final long serialVersionUID=1L;  //Para que no
aparezca warning
		public NoTrobat(){
			super("No s'ha trobat cap resultat");
		}
}
```

Accions dels Botons:

package gui;

import java.awt.event.*;

import javax.swing.*;

```
import exception.NoTrobat;
import fase2.Beca;
import fase2.LlistaBeca;


public class AccioConsultarBeca implements ActionListener{

	public void actionPerformed(ActionEvent evt) {
		// TODO Auto-generated method stub
		LlistaBeca llistaBeca=new LlistaBeca(100);
		llistaBeca.carregarBeques();
		boolean comprovat=true;
		int codi=0;
			try {
				String s = JOptionPane.showInputDialog("Diguem el codi de
la teva beca");
				codi = Integer.parseInt(s);
			} catch (NumberFormatException e) {
				System.out.print("Error en el format del codi\n");
				comprovat=false;
			}

		if(comprovat){
			try {
				Beca beca = llistaBeca.consultarBeca(codi);
			System.out.println(beca);
		} catch (NoTrobat e) {
			System.out.println("No s'ha trobat una beca");
		}
```

```
		}

	}

}
package gui;
import java.awt.event.*;
import javax.swing.*;
import exception.NoTrobat;
import fase2.Beca;
import fase2.LlistaBeca;

public class AccioConsultarBequesPais implements ActionListener{

	@Override
	public void actionPerformed(ActionEvent evt) {
		// TODO Auto-generated method stub

		LlistaBeca llistaBeca=new LlistaBeca(100);
		llistaBeca.carregarBeques();

		String pais = JOptionPane.showInputDialog("Diguem el pais de la beca");

		try{
			Beca[] beques = llistaBeca.llistaBequesPais(pais);
			for (int i = 0; i < beques.length; i++) {
				System.out.println(beques[i].toString());
			}
```

```
		} catch (NoTrobat e) {
			System.out.println("No s'ha trobat una beca");
		}
	}



}

package gui;
import java.awt.event.*;
import javax.swing.*;
import exception.NoTrobat;
import fase2.LlistaBeca;

public class AccioConsultarDinersErasmusPlus implements ActionListener{

	@Override
	public void actionPerformed(ActionEvent evt) {
		// TODO Auto-generated method stub
		LlistaBeca llistaBeca=new LlistaBeca(100);
		llistaBeca.carregarBeques();

		boolean comprovat = true;
		int codi=0;
			try {
				String s = JOptionPane.showInputDialog("Diguem el codi de
la teva beca");
				codi = Integer.parseInt(s);
```

```
                        } catch (NumberFormatException e) {
                                System.out.print("Error en el format del codi\n");
                                comprovat=false;
                        }
                if(comprovat){
                try {
                        System.out.println("La dotació de la beca és de: "+llistaBeca.consultarDotacioErasmusPlus(codi)+"€");
                } catch (NoTrobat e) {
                        System.out.println("No s'ha trobat una beca");
                }
                }
        }

}

package gui;
import java.awt.event.*;
import javax.swing.*;
import exception.NoTrobat;
import fase2.LlistaBeca;

public class AccioGetNumBequesAmbIdioma implements ActionListener{

        @Override
        public void actionPerformed(ActionEvent evt) {
                // TODO Auto-generated method stub
                LlistaBeca llistaBeca=new LlistaBeca(100);
                llistaBeca.carregarBeques();
```

```
            String idioma = JOptionPane.showInputDialog("Diguem l'idioma de la beca");

            try{
                int[] beques = llistaBeca.consultarBequesAmbIdioma(idioma);
                System.out.println("Els codis de les beques amb aquest idioma són: ");
                for (int i = 0; i < beques.length; i++)
                    System.out.println(beques[i]);
            }catch (NoTrobat e){
                System.out.println("Beca no trobada");
            }
        }

}
```

Codi de la Finestra:

```
package gui;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import fase2.LlistaBeca;
```

```
public class Finestra extends JFrame {

	private static final long serialVersionUID = 1L;
	private JLabel etiqueta=new JLabel("Menu Beques");
	private JButton[] botons=new JButton[4];

	private JPanel panell=new JPanel();
	private JPanel entrada=new JPanel();
	// Constructor sense paràmetres,
	// crida a un altre constructor on cal el títol de la finestra com a paràmetre.
	public Finestra(LlistaBeca llistaBeca) {
		super("Finestra");

		// Obtenim la referència a l'objecte "contenidor" de la finestra.
		Container container = getContentPane();

		// Forcem la disposició dels objectes que contindrà el contenidor
		// de la finestra principal al tipus "BorderLayout"
		container.setLayout(new BorderLayout());

		//AccioDelTextField2 accio2=new AccioDelTextField2(this);

		entrada.setLayout(new FlowLayout());
		//dades.addActionListener(accio2);
		container.add(entrada, BorderLayout.CENTER);


		panell.setLayout(new GridLayout(4,1));
		//AccioDelBoto accio= new AccioDelBoto();
```

```
String[] text={"Consultar informacio beca a partir del codi","Consultar les beques disponibles a un cert pais","Consultar la quantitat total de diners de la beca ERASMUSPlus","Consultar beques segons idiomes pel cas de ERASMUPlus i ISEP"};
for (int i=0; i<4;i++){
    botons[i]=new JButton(text[i]);
    botons[i].setBackground(Color.WHITE);
    panell.add(botons[i]);
}



AccioConsultarBeca accio=new AccioConsultarBeca();
botons[0].addActionListener(accio);

AccioConsultarBequesPais accio2 = new AccioConsultarBequesPais();
botons[1].addActionListener(accio2);

AccioConsultarDinersErasmusPlus accio3=new AccioConsultarDinersErasmusPlus();
botons[2].addActionListener(accio3);

AccioGetNumBequesAmbIdioma accio4=new AccioGetNumBequesAmbIdioma();
botons[3].addActionListener(accio4);

container.add(panell, BorderLayout.CENTER);
container.add(etiqueta, BorderLayout.NORTH);
```

```
        // Necessari per alliberar la memòria quan tanquem la finestra.
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        // Forcem les mides de l'objecte contenidor, es a dir, la finestra.
        setSize(600,600);
        // Fem la finestra visible.
        setVisible(true);

        }

}
```

1.8.2 Annex. Manual d'ús

Menú inicial: L'usuari ha de seleccionar entre el menú de de sol·licituds o el menú de beques. Si l'usuari no inserta un numero sortirà un missatge d'error en el qual demanarà una dada correcte.

1-Menu beques:

1.1-Afegir beques: L'usuari haurà d'introduir les dades com demana l'enunciat per tal de crear una nova beca.

1.2-Eliminar beca: A l'usuari li demanar el codi de la beca que desitja esborrar, si aquesta beca no existeix es mostrarà per pantalla un missatge d'error.

1.3-Modificar dades de la beca: A l'usuari li demanarà el codi de la beca que desitja modificar, si aquesta beca no existeix es mostrarà per pantalla un missatge d'error.

1.4-Consultar informació d'una beca: A l'usuari se li demanarà el codi de la beca que es vol modificar, a continuació es mostrarà per pantalla la informació de la beca a consultar.

1.5-Consultar una beca segons el país : A l'usuari se li demanarà el país a consultar, si es troba una beca amb aquest idioma es mostrarà per pantalla.

1.6-Consultar diners de les beques Erasmuns+: A l'usuari se li demanarà la beca a consultat, si es troba una beca que es Erasmus+ es mostrarà la dotació per pantalla.

1.7-Consultar una beca segons l'idioma: A l'usuari se li demanarà el país a consultar, si es troba una beca amb aquest idioma es mostrarà per pantalla.

1.8-Carregar beques: L'usuari podrà carregar beques guardades a partir d'un fitxer si selecciona aquesta opció.

1.9-Exportar beques: L'usuari te la opció a guardar les beques que ha creat durant l'execució del programa.

2- Menú sol·licituds:

2.1- Nova sol·licitud: L'usuari te la opció de guardar una nova sol·licitud, haurà de seguir les instruccions donades al programa.

2.2- Consultar sol·licitud: L'usuari ha de donar el codi de la sol·licitud que vol consultar, si aquesta sol·licitud existeix es mostrarà per pantalla la informació de la sol·licitud.

2.3- Consultar quantes beques hi ha d'un cert ensenyament: L'aplicació demanarà a l'usuari l'ensenyament que vol consultar, si troba aquest ensenyament el mostrarà per pantalla.

2.4-Consultar quines beques no tenen sol·licitud: L'aplicació mostrarà per pantalla quines beques no tenen sol·licitud, si no troba cap sense, es mostrarà un missatge per pantalla.

2.5- Carregar sol·licituds des de un fitxer: L'aplicació carregarà sol·licituds des de un fitxer extern

2.6-Guardar sol·licituds des de un fitxer: L'aplicació guardarà les sol·licituds creades durant l'execució del programa en un fitxer de text.

3-Menù amb interfície gràfica

-En aquest menú es mostraran les mateixes opcions que en els dos anteriors menús, però amb la diferencia de que l'usuari es mourà a partir de la interfície gràfica.

Recursos utilitzats Software

[1] Microsoft Word 2010 ™. Editor de texts.

[2] Eclipse Luna

[3] WhatsApp

[4] Facebook

[5] Trello

[6]

Hardware

[1] Portàtils dels integrants del grup