Marina Castellano
Mar Moreno

INFORME LAB 2: IMPLEMENTING CLASS RELATIONS

En aquest laboratori hem implementat part de l'aplicació Universitat que vam haver de dissenyar durant el seminari 2. L'objectiu és crear les classes que formen part del projecte Universitat, i les relacions que hi han entre elles.
Les classes estan organitzades per la classe Universitat, la qual és la principal, i dintre de cada classe hem implementat els atributs i mètodes necessaris. Una vegada hem implementat la estructura principal hem implementat les queries.
La pràctica conté una classe anomenada Utility i un directori xml que conté els fitxers dels quals treurem la informació que utilitzarem en les altres classes.

CLASSES
Les classes implementades durant aquest laboratori han estat: University, Teacher, Student, Course, Classroom , Lecture, Assignment, Enrollment i TestUniversity.

- **Student:** Els atributs de la classe student son: name, NIA i enrollmentList.
  Cada estudiant té un nom i s'identifica pel NIA, i la classe student està associada amb la classe enrollment.
  Els mètodes implementants han estat: Student, on li hem passat el nom i el NIA; toString i addEnrollment.

- **Teacher:** Els atributs implementats en aquesta classe han estat: name i assignmentList, perquè els professors s'identifiquen amb el nom, i la classe teacher està associada amb la classe assignment.
  Els mètodes son: la funció Teacher, a la qual li passem el nom, addAssignment i toString.

- **Course:** Els atributs implementats han estat: el name, per a identificar els diferents tipus de courses; assignmentList; enrollmentList; lectureList, perquè les classes assignment, enrollment i lecture estan associades a course.
  Els mètodes implementats han estat: Course, que li hem passat el nom; toString, addLecture, addEnrollment, addAssignment.

- **Classroom:** Els atributs implementats han estat: code, per identificar la classe i lectureList, perquè la classe lecture està associada a classroom.
  Els mètodes implementats han estat: Classroom, a que li hem passat el code; addLecture i toString.

En les quatre classes principals hem implementat la funció toString, que retorna els objectes com un string per després poder afegir-lo a les llistes d'strings.

```
public String toString() {
        return this.name;
}
```

També hem implementat linked lists de: lecture, assignment, enrollment, en els constructors de les quatre classes principals, depenent de amb quines classes estan associades.

```
lectureList = new LinkedList<Lecture>();
assignmentList = new LinkedList<Assignment>();
enrollmentList = new LinkedList<Enrollment>();
```

Lecture, Assignment i Enrollment son classes d'associació.

- **Lecture:** En aquesta classe em implementat els següents atributs: group (string), quel numero de grup, el time slot (int), el type (int), i classroom i course, que son atributs de tipo classe. Els mètodes aplicats són: Lecture, que li passem el grup, el timeSlot i el type; addCourse i addClassroom.

- **Assignment:** Els atributs són: teachers, courses, que son atributs de tipus classe, i groupList. Els mètodes són Assignment, que li passem Teacher, initCourses i initGroups ; addTeacher i addCourse.

```
public Assignment( Teacher initTeacher, Course initCourses,
LinkedList<Integer> initGroups){
        groupList = new LinkedList<Integer>();
        this.groupList = initGroups;
        this.teachers =  initTeacher;
        this.courses =  initCourses;

    }
```

- **Enrollment:** Els atributs són: seminarGroup, un string del grup de seminari; student i course, que son atributs de tipus classe.
  Els mètodes són: Enrollment, que li passem seminar group; addStudent i addCourse.

- **University:** És la classe principal, ja que organitza a totes les demés classes.
  Implementem el mètode principal, University, el qual ens permet dur a terme la pràctica, ja que será la que llegeixi els fitxers xml per a poder afegir a les llistes de les classes que hem creat anteriorment.
  Hem creat set atributs amb forma linkedlist de cada una de les classes que ja hem que mencionat (teacher, student, course…), per després implementar-les a University per a poder emmagatzemar la informació en cada una d'elles.

  A continuació, hem creat una linkedlist per cada classe d'un array d'strings, que tindrà la informació del fitxer al que pertany. En les quatre primeres classes: student, teacher, course i classroom; recorrerá l'array i per a cada classe en la seva funció creara un nou element cridant a la funció principal de cada classe, amb la informació proporcionada, per després afegir-la a la llista corresponent.

  En el cas d'student li passarem el primer índex que és el nom de l'estudiant i el següent índex de l'array que és el número NIA de l'estudiant. El NIA l'hem transformat en un número enter ja que estava com a string. Després l'hem afegit a la llista studentList.

```
Student student = new Student(array[0], Integer.parseInt(array[1]));
studentList.add(student);

Al teacher només li passem un paràmetre, el nom del professor.
Teacher teacher = new Teacher(array[0]);
teacherList.add(teacher);

A el course li passem el nom del curs.
Course course = new Course(array[0]);
courseList.add(course);

Al classroom li passem el número de classe en forma d'string.
Classroom classroom = new Classroom(array[0]);
classroomList.add(classroom);
```

Per les altres tres classes: enrollment, assignment i lecture; hem creat una linkedlist d'un array d'strings, la qual tindrà la informació del fitxer al que pertany.
En el cas de la classe enrollment i de lecture, al recorrer l'array per cada classe amb la que està relacionada (en enrollment seria student i course; i en lecture, classroom i course), en lloc de cridar a la classe a la qual pertanyen, cridem a la funció getObject que està en la classe Utility, aquesta funció a través d'un string i la llista corresponent ens retorna l'objecte, que afegirem a la classe enrollment i lecture.
Dintre de la classe Universitat també hem creat una nova linkedlist d'enrollment i lecture, que cridaran a la seva classe amb la informació necessària. En el cas d'assignment, a més, també haurem de crear una linkedlist dels grups i recorrer l'array començant per l'índex 2, ja que és on comencen a estar els grups en el fitxer, i a continuació afegir-los a la llista.

- **TestUniversity:** Hem creat aquesta classe per poder imprimir les funcions dissenyades durant la pràctica, que cridarem des de la classe universitat (la classe principal).

Per comprovar si les llistes principals (student, teacher, course i classroom) les hem cridat bé en la classe University, erm fet que ens surtin per pantalla cridant-les en el TestUniversity implementant-les en la classe University.

```
public LinkedList< String> getStudents(){
      return Utility.toString(studentList);
   }
public LinkedList<String> getTeachers(){
      return Utility.toString(teacherList);
   }
public LinkedList<String> getClassrooms(){
      return Utility.toString(classroomList);
   }
public LinkedList<String> getCourses(){
      return Utility.toString(courseList);
   }
```

Hem creat una funció tipo linkedlist d'strings, que retorna la llista d'strings corresponent a cada funció depenen de cada classe, i hem utilitzat un mètode de la classe Utility, toString, perquè ens retorni la llista d'strings.

Durant la implementació de la pràctica, al crear les classes, ens va sortir un error:

Cannot invoke "java.util.LinkedList.iterator()" because "objectList" is null

No enteníem on estava el error, ja que creiem haver fet bé la implementació de les classes. Tornant a repassar la pràctica vam veure que hi havia un error al crear public university, i que al utility en la funció readXML, per a que funciones vam haver de canviar xml/ per:

```
File input = new File("code/xml/" + type + "s.xml");
```

Després de fer els següents canvis, el programa va compilar i va imprimir el nom dels estudiants.

A continuació vem procedir a implementar las queries.

- **CoursesOfStudent (obligatòria):** El que fa aquesta funció és agafar de la classe enrollment els cursos d'un estudiant, i retornar una llista d'strings de quins cursos fa aquest estudiant.

```
LinkedList<String[]> enroll = Utility.readXML("enrollment");
public LinkedList<String> coursesOfStudent (String student) {
      LinkedList<String>        coursesOfStudent        =        new
      LinkedList<String>();
       for(String[] array: enroll){
            if(student.equals(array[0])){
                 coursesOfStudent.add(array[1]);

            }
       }
       return Utility.toString(coursesOfStudent);
 }
```

- **TecahersOfCourse (obligatòria):** Aquesta funció retorna una llista d'strings dels cursos d'un professor, agafant dels cursos de classe assignment de cada professor.

```
LinkedList<String[]> assig = Utility.readXML("assignment");
public LinkedList<String> teachersOfCourse (String teacher) {
      LinkedList<String>        teachersOfCourse        =        new
      LinkedList<String>();
       for(String[] array: assig){
            if(teacher.equals(array[0])){
                 teachersOfCourse.add(array[1]);
            }
        }
        return Utility.toString(teachersOfCourse);
    }
```

- **CoursesOfClassroom (extra):** La funció ha de retornar una llista d'strings dels cursos a els quals es fa a la classroom cridada.

El plantejament que hem utilitzat per fer les tres queries és el mateix. Primer hem llegit el fitxer corresponent on es troba la informació necessària, depenent de cada funció i les hem posat en un array. Després hem recorregut l'array i hem comparat les dades que entren a la funció quan la cridem en el TestUniverity, amb les que hi ha en el fitxer; si les dades concideixen , les afegim en la llista que hem creat dins de la funció.

Les llistes de les funcions de cada query:

```
LinkedList<String> coursesOfStudent = new LinkedList<String>();
LinkedList<String> teacherOfCourses = new LinkedList<String>();
LinkedList<String> coursesOfClassroom = new LinkedList<String>();
```

Per a que ens retorni la llista d'strings cridem a la funció toString de la classe Utility, amb la llista que conté la informació que em afegit abans.
En la última query, la que és extra, hem fet el mateix però hem afegit que si es repeteix la informació que afegim la llista, l'elimini per a que només hi aparegui un cop, ja que sinó la llista ens retornava més d'una vegada el mateix curs. Per tant, hem pensat que només s'afegeixi un cop el curs amb el número de la classroom, si coincideix amb el que cridem a la funció.

CONCLUSIÓ

Una vegada implementat tot el codi, hem observat que el resultat ha estat l'esperat, i que per tant les classes han estat ben implementades. Hem comprovat que els resultats de les queries son correctes provant-les per diferents estudiants, professors, cursos… en el TestUniversity.

Creiem que aquest laboratori ens ha ajudat molt ha entendre com funcionen les relacions entre les classes.