

Johann Amaya Lopez  
Sebatian Zamora Urrego  
PROGRAMACIÓN ORIENTADA A OBJETOS  
Excepciones  
2023-1  
Laboratorio 4/6

## OBJETIVOS

1. Perfeccionar el diseño y código de un proyecto considerando casos especiales y errores.
2. Construir clases de excepción encapsulando mensajes.
3. Manejar excepciones considerando los diferentes tipos.
4. Registrar la información de errores que debe conocer el equipo de desarrollo de una aplicación en producción.
5. Vivenciar las prácticas *Designing* - *Simplicity*.



Refactor whenever and wherever possible .

## ENTREGA

- ☑ Incluyan en un archivo **.zip** los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.
- ☑ Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada, en los espacios preparados para tal fin.

## Compos

### EN BLUEJ

## PRACTICANDO MDD y BDD con EXCEPCIONES

[En lab04.doc, IEMOIS.asta y BlueJ program]

En este punto vamos a aprender a diseñar, codificar y probar usando excepciones. Para esto se van a trabajar algunos métodos de la clase [Nanodegree](#)

1. En su directorio descarguen los archivos contenidos en [programs.zip](#) revisen el contenido y estudien el diseño estructural de la aplicación (únicamente la zona en azul).
2. Expliquen por qué el proyecto no compila. Realicen las adiciones necesarias para lograrlo.
  - El proyecto no compila ya que falta la clase excepciones .

```

package domain;

/**
 * The exception class of IEMOIS
 *
 * @author Johann Amaya Lopez
 * @author Sebastian Zamora Urrego
 * @version 1.0
 */
public class IEMOISException extends Exception
{
    public static final String WEEKS_EMPTY="";
    public static final String WEEKS_ERROR="";

    /**
     * Constructor for objects of class IEMOISException
     */
    public IEMOISException(String msm)
    {
        super(msm);
    }
}

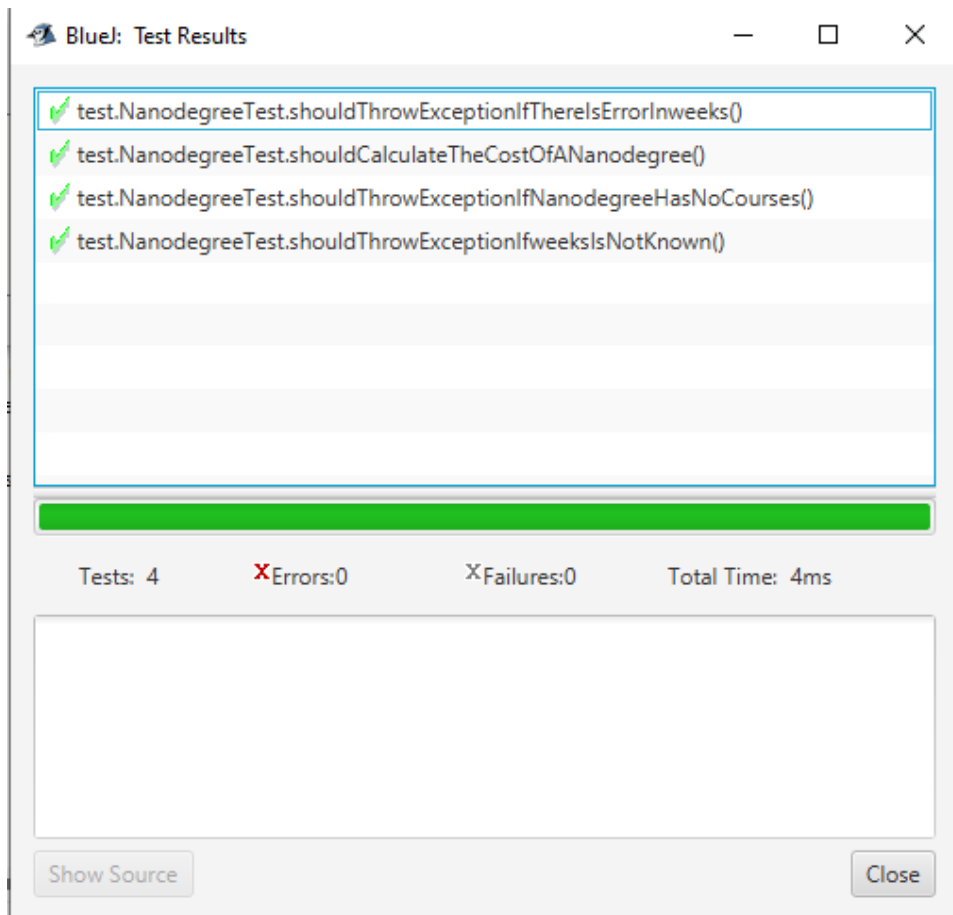
```

- 
3. Dado el diseño y las pruebas, documenten y codifiquen el método `weeks()`.

```

@Override
public int weeks() throws IEMOISException{
    int weeks = 0;
    if(courses.size() == 0)throw new IEMOISException(IEMOISException.NANO_EMPTY);
    for(Course i: courses){
        weeks += i.weeks();
    }
    return projectWeeks + weeks ;
}

```

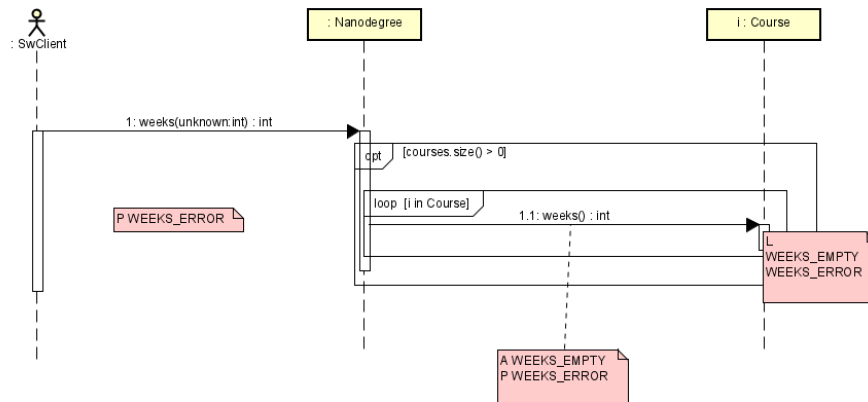


4. Dada la documentación y el diseño, codifiquen y prueben el método `weeks(type)`.

```
/**
 * Calculates an estimate of weeks
 * For courses where the weeks cannot be known or has error, the max, min or avg of the known courses is assumed
 * @param type (max,min,avg)
 * @return
 * @throws IEMOISException NANO_EMPTY, if it don't have courses. IMPOSSIBLE, if it can't be calculated
 */
public int weeks(String type) throws IEMOISException{
    int weeks = projectWeeks;
    int error = 0;
    int cambio = cambio(type);
    if(courses.size() == 0) throw new IEMOISException(IEMOISException.NANO_EMPTY);
    for(Course i: courses){
        try{
            weeks += i.weeks();
        }catch(IEMOISException e){
            error ++;
            weeks += cambio;
        }
    }
    if(error == courses.size()) throw new IEMOISException(IEMOISException.IMPOSSIBLE);
    return weeks;
}
```

- ✓ test.NanodegreeTest.shouldThrowExceptionIfThereIsErrorInWeeks()
- ✓ test.NanodegreeTest.shouldCalculateTheCostOfANanodegree()
- ✓ test.NanodegreeTest.shouldCalculateTheMaxOfTheWeeks()
- ✓ test.NanodegreeTest.shouldCalculateTheAverageOfTheWeeks()
- ✓ test.NanodegreeTest.shouldThrowExceptionIfNanodegreeHasNoCourses()
- ✓ test.NanodegreeTest.shouldThrowExceptionIfWeeksIsNotKnown()
- ✓ test.NanodegreeTest.shouldCalculateTheMinOfTheWeeks()

5. Documenten, diseñen, codifiquen y prueben el método `weeks(unknown)`.



```

/**
 * Calculates an estimate of weeks
 * For courses where the weeks cannot be known, unknown is assumed
 *
 * @param unknown the number of weeks that assume if the week has a error
 * @return The total course weeks.
 * @throws IEMOISException NANO_EMPTY, if it don't have courses. WEEKS_ERROR, if
 * some course has error
 */
public int weeks(int unknown) throws IEMOISException {
    int valor = projectWeeks;
    if (courses.size() > 0) {
        for (Course i : courses) {
            try {
                valor += i.weeks();
            } catch (IEMOISException e) {
                if (e.getMessage().equals(IEMOISException.WEEKS_EMPTY)) {
                    valor += unknown;
                } else {
                    throw new IEMOISException(IEMOISException.WEEKS_ERROR);
                }
            }
        }
    }
    return valor;
}
  
```

```

✓ test.NanodegreeTest.shouldNotCalculateTotalWeeksError()
✓ test.NanodegreeTest.shouldCalculateTotalWeeksWithEmptyWeeks()
✓ test.NanodegreeTest.shouldThrowExceptionIfThereIsErrorInWeeks()
✓ test.NanodegreeTest.shouldCalculateTheCostOfANanodegree()
✓ test.NanodegreeTest.shouldCalculateTheMaxOfTheWeeks()
✓ test.NanodegreeTest.shouldCalculateTheAverageOfTheWeeks()
✓ test.NanodegreeTest.shouldThrowExceptionIfNanodegreeHasNoCourses()
✓ test.NanodegreeTest.shouldThrowExceptionIfWeeksIsNotKnown()
✓ test.NanodegreeTest.shouldCalculateTheMinOfTheWeeks()
✓ test.NanodegreeTest.shouldCalculateANormalWeeks()

```

## IEMOIS

### EN CONSOLA

El objetivo de esta aplicación es mantener un catálogo de los MOOC ofrecidos por la decanatura en el período intermedio a sus estudiantes en el proyecto IEMOIS. En este proyecto se ofrecen diferentes programas: cursos y *nanodegrees*.

### Conociendo el proyecto IEMOIS [En lab04.doc]

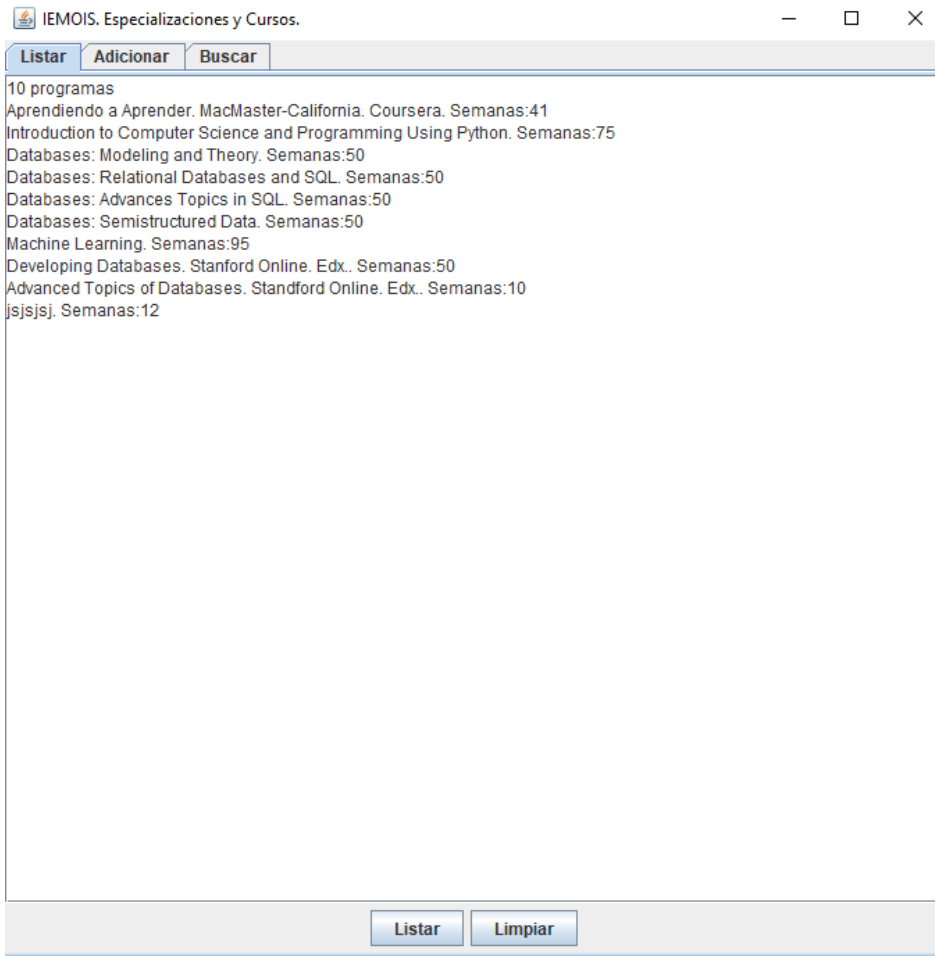
#### No olviden respetar los directorios bin docs src

- En su directorio descarguen los archivos contenidos en [IEMOIS.zip](#), revisen el contenido.  
¿Cuántos archivos se tienen? ¿Cómo están organizados? ¿Cómo deberían estar organizados?
  - El directorio tiene 3 archivos: IEMOISGUI, IEMOIS y Log, todos son .java.
  - Están todos en el mismo directorio.
  - Deberían estar separados en los directorios domain y presentation.
- Estudien el diseño del programa: diagramas de paquetes y de clases. ¿cuántos paquetes tenemos? ¿cuántas clases tiene el sistema? ¿cómo están organizadas? ¿cuál es la clase ejecutiva?
  - Tenemos dos paquetes: Presentation y Domain.
  - Tiene 6 clases: IEMOIS, IEMOISGUI, Program, Nanodegree, Course y Log.
  - Esta dividido en los dos paquetes que mencionamos anteriormente, en el paquete de Domain tenemos las clases IEMOIS, Log, Program, Nanodegree y Course y en Presentation está IEMOISGUI.
  - Se ejecuta la clase de IEMOISGUI dado que es la que tiene el método main().
- Prepare los directorios necesarios para ejecutar el proyecto. ¿qué estructura debe tener? ¿qué clases deben tener? ¿dónde están esas clases? ¿qué instrucciones debe dar para ejecutarlo?
  - Debe tener tres directorios principales, Docs que va a ir la documentación, bin que van los ejecutables de las clases y src donde estarán las clases organizadas en los dos paquetes de Presentation y Domain.
  - En el paquete de domain que estaba en un zip aparte están Program, Nanodegree y Course y en el de IEMOIS están IEMOISGUI, IEMOIS y Log.

```

C:\Repositorio\POOB\LAB_4>javac -d bin src/presentation/IEMOISGUI.java src/domain/*.java
C:\Repositorio\POOB\LAB_4>java -cp bin presentation/IEMOISGUI

```
- Ejecute el proyecto, ¿qué funcionalidades ofrece? ¿cuáles funcionan?



- Nos ofrece las funciones que nos ofrece Listar los programas, limpiar el lugar donde se ven los programas, adicionar y buscar programas.
- En este caso la única que no funciona es buscar un curso ya que lanza el siguiente error:

```
C:\Repositorio\POOB\LAB_4>java -cp bin presentation/IEMOISGUI
Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException: Cannot invoke "java.util.LinkedList.add(Object)"
because "<local2>" is null
```

5. Revisen el código y la documentación del proyecto. ¿De dónde salen los cursos iniciales? ¿Qué clase pide que se adicionen? ¿Qué clase los adiciona?

- Los cursos iniciales están en la clase addSome de la clase IEMOIS.java.
- El constructor y addCourse.
- Los adiciona addCourse.

**Adicionar y listar. Todo OK.** [En lab04.doc, IEMOIS.astay \*.java]

(NO OLVIDEN BDD - MDD)

El objetivo es realizar ingeniería reversa a las funciones de adicionar y listar.

1. Adicionen un nuevo curso y un nuevo *nanodegree* Curso

Natural Computing

5

*Nanodegree*

Artificial Intelligence

¿Qué ocurre? ¿Cómo lo comprueban? Capturen la pantalla. ¿Es adecuado este comportamiento?

- No se pueden listar dado que hay un problema con el método data de Nanodegree.

```
C:\Repositorio\POOB\LAB_4>java -cp bin presentation\IEMOISGUI
Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException: Cannot invoke "domain.Course.data()" because "<local3>" is null
    at domain.Nanodegree.data(Nanodegree.java:130)
    at domain.IEMOIS.data(IEMOIS.java:129)
    at domain.IEMOIS.toString(IEMOIS.java:154)
    at presentation.IEMOISGUI.actionList(IEMOISGUI.java:236)
    at presentation.IEMOISGUI$actionPerformed(IEMOISGUI.java:191)
    at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1972)
    at java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2314)
    at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:407)
    at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
    at java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:279)
    at java.desktop/java.awt.Component.processMouseEvent(Component.java:6620)
    at java.desktop/javax.swing.JComponent.processMouseEvent(JComponent.java:3398)
    at java.desktop/java.awt.Component.processEvent(Component.java:6385)
    at java.desktop/java.awt.Container.processEvent(Container.java:2266)
    at java.desktop/java.awt.Component.dispatchEventImpl(Component.java:4995)
    at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2324)
    at java.desktop/java.awt.Component.dispatchEvent(Component.java:4827)
    at java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:4948)
    at java.desktop/java.awt.LightweightDispatcher.processMouseEvent(Container.java:4575)
    at java.desktop/java.awt.LightweightDispatcher.dispatchEvent(Container.java:4516)
    at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2310)
    at java.desktop/java.awt.Window.dispatchEventImpl(Window.java:2780)
    at java.desktop/java.awt.Component.dispatchEvent(Component.java:4827)
    at java.desktop/java.awt.EventQueue.dispatchEventImpl(EventQueue.java:775)
    at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:720)
    at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:714)
    at java.base/java.security.AccessController.doPrivileged(AccessController.java:400)
    at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:87)
    at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:747)
    at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:745)
    at java.base/java.security.AccessController.doPrivileged(AccessController.java:400)
    at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:87)
    at java.desktop/java.awt.EventQueue.dispatchEvent(EventQueue.java:744)
    at java.desktop/java.awt.EventDispatchThread.pumpOneEventForFilters(EventDispatchThread.java:203)
    at java.desktop/java.awt.EventDispatchThread.pumpEventsForFilter(EventDispatchThread.java:124)
    at java.desktop/java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:113)
    at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:109)
    at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:101)
    at java.desktop/java.awt.EventDispatchThread.run(EventDispatchThread.java:90)
```

- No es adecuado dado que no debería generar el error.
- Revisen el código asociado a **adicionar** en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método en la capa de dominio?
- Presentación:

```
1 usage new *
private void actionAdd(){
    if (courses.getText().trim().equals("")){
        programs.addCourse(name.getText(),discount.getText());
    }else{
        programs.addSpecialization(name.getText(),discount.getText(),courses.getText());
    }
}
```

- Domain:

```
/**
 * Add a new course
 * @param name
 * @param price
 */
1 usage new *
public void addCourse(String name, String price){
    Course nc=new Course(name,Integer.parseInt(price));
    programs.add(nc);
    courses.put(name.toUpperCase(),nc);
}
```

- Realicen ingeniería reversa para la capa de dominio para **adicionar**. Capturen los resultados de las pruebas de unidad.

```

@Test
public void shouldAddCourses() {
    IEMOIS caja = new IEMOIS();
    caja.addCourse(name:"Programaci3n Orientada a Objetos", price:"7");
    caja.addCourse(name:"Algoritmos y Estructuras de Datos", price:"7");
    caja.addCourse(name:"Modelos de Bases de Datos", price:"7");
    caja.addCourse(name:"Algoritmos y Programacion", price:"7");
    assertEquals(15, caja.numberPrograms());
}

```

BlueJ: Probar Resultados

✓ test.IEMOISTest.shouldAddCourses()



Ejecuciones: 1

✗Errores:0

✗Fallos:0

Tiempo Total: 12ms

4. Revisen el código asociado a **listar** en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método en la capa de dominio?

- Presentación:

```

private void actionList(){
    textDetails.setText(programs.toString());
}

```

- Dominio:

```

/**
 * Return the data of all programs
 *
 * @return
 */
public String toString() {
    return data(programs);
}

```



```

/**
 * Consult selected programs
 *
 * @param selected
 * @return
 */
public String data(LinkedList<Program> selected) {
    StringBuffer answer = new StringBuffer();
    answer.append(programs.size() + " programas\n");
    for (Program p : programs) {
        try {
            answer.append(p.data());
            answer.append(str+"\n");
        } catch (IEMOISException e) {
            answer.append("**** " + e.getMessage());
        }
    }
    return answer.toString();
}

```

5. Realicen ingeniería reversa para la capa de dominio para **listar**.  
Capturen los resultados de las pruebas de unidad.
6. Propongan y ejecuten una prueba de aceptación.
  - Vamos agregar los cursos que tenemos indicados en la prueba de shouldAddCourses y vamos listando al final.

Listar

Adicionar

Buscar

nombre

Algoritmos y Estructuras de Datos

precio o descuento

7

Listar

Adicionar

Buscar

16 programas

Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41  
Introduction to Computer Science and Programming Using Python. Semanas:75  
Databases: Modeling and Theory. Semanas:50  
Databases: Relational Databases and SQL. Semanas:50  
Databases: Advances Topics in SQL. Semanas:50  
Databases: Semistructured Data. Semanas:50  
Machine Learning. Semanas:95  
Natural Computing. Semanas:5  
Developing Databases. Stanford Online. Edx. Semanas:50  
Advanced Topics of Databases. Stanford Online. Edx. Semanas:10  
Artificial Intelligence. Proyecto: 3  
Natural Computing. Semanas:5  
Machine Learning. Semanas:95  
Programación Orientada a Objetos. Semanas:7  
Algoritmos y Estructuras de Datos. Semanas:7  
Algoritmos y Estructuras de Datos. Semanas:7  
Modelos de Bases de Datos. Semanas:7  
Algoritmos y Programacion. Semanas:7

**Adicionar un programa: curso o *nanodegree*. Funcionalidad robusto** [\[En lab04.doc,](#)

[IEMOIS.astay \\*.java\]](#)

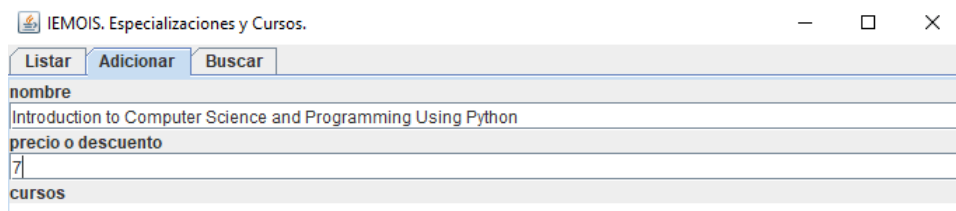
(NO OLVIDEN BDD – MDD)

El objetivo es perfeccionar la funcionalidad de adicionar un curso para hacerla más robusta.

**Para cada uno de los siguientes casos realice los pasos del 1 al 4.**

a. ¿Y si el nombre del curso ya existe?

1. Vamos de nuevo a agregar el curso: Introduction to Computer Science and Programming Using Python

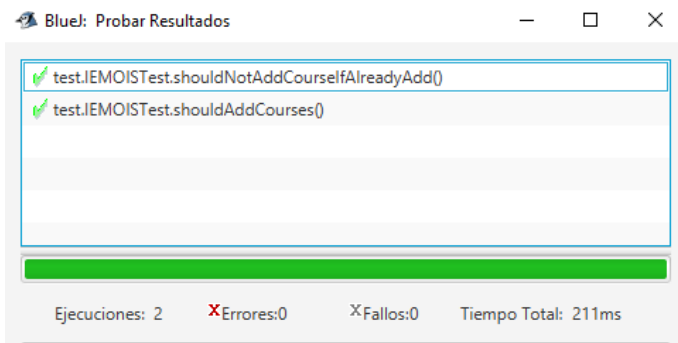


```
12 programas
Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41
Introduction to Computer Science and Programming Using Python. Semanas:75
Databases: Modeling and Theory. Semanas:50
Databases: Relational Databases and SQL. Semanas:50
Databases: Advances Topics in SQL. Semanas:50
Databases: Semistructured Data. Semanas:50
Machine Learning. Semanas:95
Natural Computing. Semanas:5
Developing Databases. Stanford Online. Edx.. Semanas:50
Advanced Topics of Databases. Standford Online. Edx.. Semanas:10
Artificial Intelligence. Proyecto: 3
Natural Computing. Semanas:5
Machine Learning. Semanas:95
Introduction to Computer Science and Programming Using Python. Semanas:7
```

2. La excepción la lanza el método de addCourse de la clase IEMOIS de domain, dado que ahí es donde yo puedo hacer la validación si el curso ya esta, en este caso como el método addCourse de una es llamado en el método actionAdd este sería el que de una la resuelve.

3.

```
@Test
public void shouldNotAddCourseIfAlreadyAdd() {
    IEMOIS program = new IEMOIS();
    try {
        program.addCourse(name:"Programación Orientada a Objetos", price:"7");
        program.addCourse(name:"Algoritmos y Estructuras de Datos", price:"7");
        program.addCourse(name:"Programación Orientada a Objetos", price:"7");
        fail("Not Threw an exception.");
    } catch (Exception e) {
        assertEquals(e.getMessage(), IEMOISException.COURSE_EXIST);
    }
}
```



4.

<p>11 programas</p> <p>Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41</p> <p>Introduction to Computer Science and Programming Using Python. Semanas:75</p> <p>Databases: Modeling and Theory. Semanas:50</p> <p>Databases: Relational Databases and SQL. Semanas:50</p> <p>Databases: Advances Topics in SQL. Semanas:50</p> <p>Databases: Semistructured Data. Semanas:50</p> <p>Machine Learning. Semanas:95</p> <p>Natural Computing. Semanas:5</p> <p>Developing Databases. Stanford Online. Edx. Semanas:50</p> <p>Advanced Topics of Databases. Stanford Online. Edx. Semanas:10</p> <p>Artificial Intelligence. Proyecto: 3</p> <p>Natural Computing. Semanas:5</p> <p>Machine Learning. Semanas:95</p>
<p><b>nombre</b></p> <p>Introduction to Computer Science and Programming Using Python</p>
<p><b>precio o descuento</b></p> <p>5</p>
<p><b>cursos</b></p>

<p>11 programas</p> <p>Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41</p> <p>Introduction to Computer Science and Programming Using Python. Semanas:75</p> <p>Databases: Modeling and Theory. Semanas:50</p> <p>Databases: Relational Databases and SQL. Semanas:50</p> <p>Databases: Advances Topics in SQL. Semanas:50</p> <p>Databases: Semistructured Data. Semanas:50</p> <p>Machine Learning. Semanas:95</p> <p>Natural Computing. Semanas:5</p> <p>Developing Databases. Stanford Online. Edx. Semanas:50</p> <p>Advanced Topics of Databases. Stanford Online. Edx. Semanas:10</p> <p>Artificial Intelligence. Proyecto: 3</p> <p>Natural Computing. Semanas:5</p> <p>Machine Learning. Semanas:95</p>
--

b. ¿Y si en semanas no da un número? ¿o no da un número negativo?

1.Vamos agregar el curso de Algoritmos y Estructuras de Datos con unas semanas de -2 y el curso de Modelos de Bases de Datos sin un numero de semanas.

<p>IEMOIS. Especializaciones y Cursos.</p>
<p>Listar Adicionar Buscar</p>
<p><b>nombre</b></p> <p>Algoritmos y Estructuras de Datos</p>
<p><b>precio o descuento</b></p> <p>-2</p>
<p><b>cursos</b></p>

Listar	Adicionar	Buscar
nombre		
Modelos de Bases de Datos		
precio o descuento		
cursos		

12 programas

Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41  
 Introduction to Computer Science and Programming Using Python. Semanas:75  
 Databases: Modeling and Theory. Semanas:50  
 Databases: Relational Databases and SQL. Semanas:50  
 Databases: Advances Topics in SQL. Semanas:50  
 Databases: Semistructured Data. Semanas:50  
 Machine Learning. Semanas:95  
 Natural Computing. Semanas:5  
 Developing Databases. Stanford Online. Edx.. Semanas:50  
 Advanced Topics of Databases. Standford Online. Edx.. Semanas:10  
 Artificial Intelligence. Proyecto: 3  
 Natural Computing. Semanas:5  
 Machine Learning. Semanas:95  
 Algoritmos y Estructuras de Datos. Semanas:-2

Vemos que para el caso del valor nulo lanza un mensaje por consola. Dado que es lanzada un nullPointer por el método addionAdd que al momento de llamar el método addCourse al discount le aplica el método getText que lanza la excepcion

```
C:\Repositorio\POOB\LAB_4>java -cp bin presentation/IEMOISGUI
For input string: ""
```

2. En este caso vamos a directamente propagar en el método de IEMOIS addCourse dado que esa clase Course ya tiene un metodo que me hace la verificación si ingresan un valor nulo o negativo, y se atenderia de la misma manera que el anterior.

3.

```
@Test
public void shouldNotAddCoureIfTheWeeksNumberIsWrong() {
    IEMOIS program = new IEMOIS();
    try {
        program.addCourse(name:"Programación Orientada a Objetos", price:"7");
        program.addCourse(name:"Algoritmos y Estructuras de Datos", price:"7");
        program.addCourse(name:"Modelos de Bases de Datos", price:"-7");
        fail("Not Threw an exception.");
    } catch (Exception e) {
        assertEquals(e.getMessage(), IEMOISException.WEEKS_ERROR);
    }
}

@Test
public void shouldNotAddCoureIfTheWeeksNumberIsEmpty() {
    IEMOIS program = new IEMOIS();
    try {
        program.addCourse(name:"Programación Orientada a Objetos", price:"7");
        program.addCourse(name:"Algoritmos y Estructuras de Datos", price:"7");
        program.addCourse(name:"Modelos de Bases de Datos", price:"");
        fail("Not Threw an exception.");
    } catch (Exception e) {
        assertEquals(e.getMessage(), IEMOISException.WEEKS_EMPTY);
    }
}
```

✓ test.IEMOISTest.shouldNotAddCourseIfAlreadyAdd()
✗ test.IEMOISTest.shouldNotAddCourseIfTheWeeksNumberIsEmpty()
✓ test.IEMOISTest.shouldNotAddCourseIfTheWeeksNumberIsWrong()
✓ test.IEMOISTest.shouldAddCourses()

Ejecuciones: 4    ✗ Errores: 0    ✗ Fallos: 1    Tiempo Total: 232ms

4.

— □ ×

Listar   Adicionar   Buscar

nombre

Algoritmos y Estructuras de Datos

precio o descuento

-2

cursos

---

— □ ×

Listar   Adicionar   Buscar

11 programas

Aprendiendo a Aprender. MacMaster-California. Coursera. Semanas:41

Introduction to Computer Science and Programming Using Python. Semanas:75

Databases: Modeling and Theory. Semanas:50

Databases: Relational Databases and SQL. Semanas:50

Databases: Advances Topics in SQL. Semanas:50

Databases: Semistructured Data. Semanas:50

Machine Learning. Semanas:95

Natural Computing. Semanas:5

Developing Databases. Stanford Online. Edx. Semanas:50

Advanced Topics of Databases. Stanford Online. Edx. Semanas:10

Artificial Intelligence. Proyecto: 3

Natural Computing. Semanas:5

Machine Learning. Semanas:95

### c. Proponga una nueva condición.

En este caso implementaremos que en el caso anterior, en vez de lanzar error le va colocar el promedio de las semanas de los proyectos.

1

— □ ×

Listar   Adicionar   Buscar

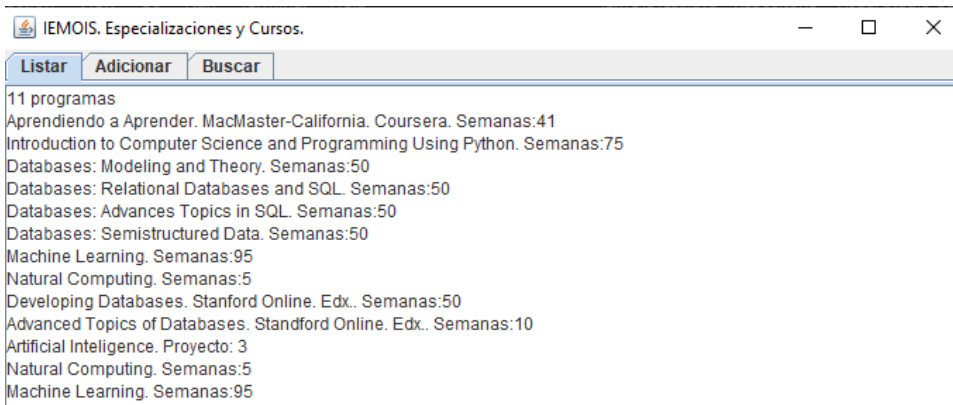
nombre

Algoritmos y Estructuras de Datos

precio o descuento

-2

cursos



Sucede lo que pudimos corregir en el inciso b, que no se agregue.

2. esta vez en ves de propagar las vamos atender y les cambiamos el numero erróneo o el vacío se coloque el promedio.

3.

```
@Test
public void shouldAddCoursesWithWrongAndEmptyWeeks() {
    IEMOIS caja = new IEMOIS();
    try {
        caja.addCourse("Programación Orientada a Objetos", "7");
        caja.addCourse("Algoritmos y Estructuras de Datos", "-7");
        caja.addCourse("Modelos de Bases de Datos", "");
        assertEquals(14, caja.numberPrograms());
    } catch (Exception e) {
        fail("Throw an Exception");
    }
}
```

El objetivo es perfeccionar la funcionalidad de adicionar una *nanodegree* para hacerla más robusta.

**Para cada uno de los siguientes casos realice los pasos del 1 al 4.**

- a. ¿Y si el nombre del *nanodegree* ya existe?
- b. ¿Y si las semanas de proyecto no da un número? ¿o no da un número negativo?
- c. ¿Y si alguno de los cursos que lo componen no existen?
- d. **Proponga una nueva condición**
  1. Propongan una prueba de aceptación que genere el fallo.
  2. Analicen el diseño realizado. Para hacer el software robusto: ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique claramente.
  3. Construya la solución propuesta. Capture los resultados de las pruebas de unidad.
  4. Ejecuten nuevamente la aplicación con el caso de aceptación propuesto en 1. ¿Qué sucede ahora? Capture la pantalla.

### Consultando por patrones. ¡ No funciona y queda sin funcionar!

[En IEMOIS.asta, IEMOIS.log, lab04.java y \*.java]

**(NO OLVIDEN BDD - MDD)**

1. Consulten un combo especial que inicie con I. ¿Qué sucede? ¿Qué creen que pasó? Capturen el resultado. ¿Quién debe conocer y quien NO debe conocer esta información?
2. Explore el método **record** de la clase **Log** ¿Qué servicio presta?
3. Analicen el punto adecuado para que **EN ESTE CASO** se presente un mensaje especial de alerta al usuario, se guarde la información del error en el registro y continúe la ejecución. Expliquen y construyan la solución.

4. Ejecuten nuevamente la aplicación con el caso propuesto en 1. ¿Qué mensaje salió en pantalla? ¿La aplicación termina? ¿Qué información tiene el archivo de errores?
5. ¿Es adecuado que la aplicación continúe su ejecución después de sufrir un incidente como este? ¿de qué dependería continuar o parar?
6. Modifiquen la aplicación para garantizar que **SIEMPRE** que haya un error se maneje de forma adecuada. ¿Cuál fue la solución implementada?

### Consultando por patrones. ¡Ahora si funciona!

[En IEMOIS.asta, IEMOIS.log, lab04.java y \*.java]

#### (NO OLVIDEN BDD - MDD)

1. Revisen el código asociado a **buscar** en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método es responsable en la capa de dominio?
2. Realicen ingeniería reversa para la capa de dominio para **buscar**. Capturen los resultados de las pruebas. Deben fallar.
3. ¿Cuál es el error? Solúcenlo. Capturen los resultados de las pruebas.
4. Ejecuten la aplicación nuevamente con el caso propuesto. ¿Qué tenemos en pantalla? ¿Qué información tiene el archivo de errores?
5. Refactorice la funcionalidad para que sea más amable con el usuario. ¿Cuál es la propuesta? ¿Cómo la implementa?

### RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)
  - 18h ambos
2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?
  - Nos falta los dos últimos ciclos, dado que nos demoramos mucho en los ciclos anteriores.
3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?
  - Desi
4. ¿Cuál consideran fue el mayor logro? ¿Por qué?
  - Lograr compilar y correr por consola, dado que con ciertos comandos habíamos tenido dificultades.
5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?
  - No tuvimos grandes problemas técnicos.
6. ¿Qué hicieron bien como actividades? ¿Qué se comprometen a hacer para mejorar los resultados?
  - Seguimos trabajando muy bien juntos, a mejorar estaría el lograr soluciones más acertadas en un menor tiempo posible.