PROGRAMACIÓN II

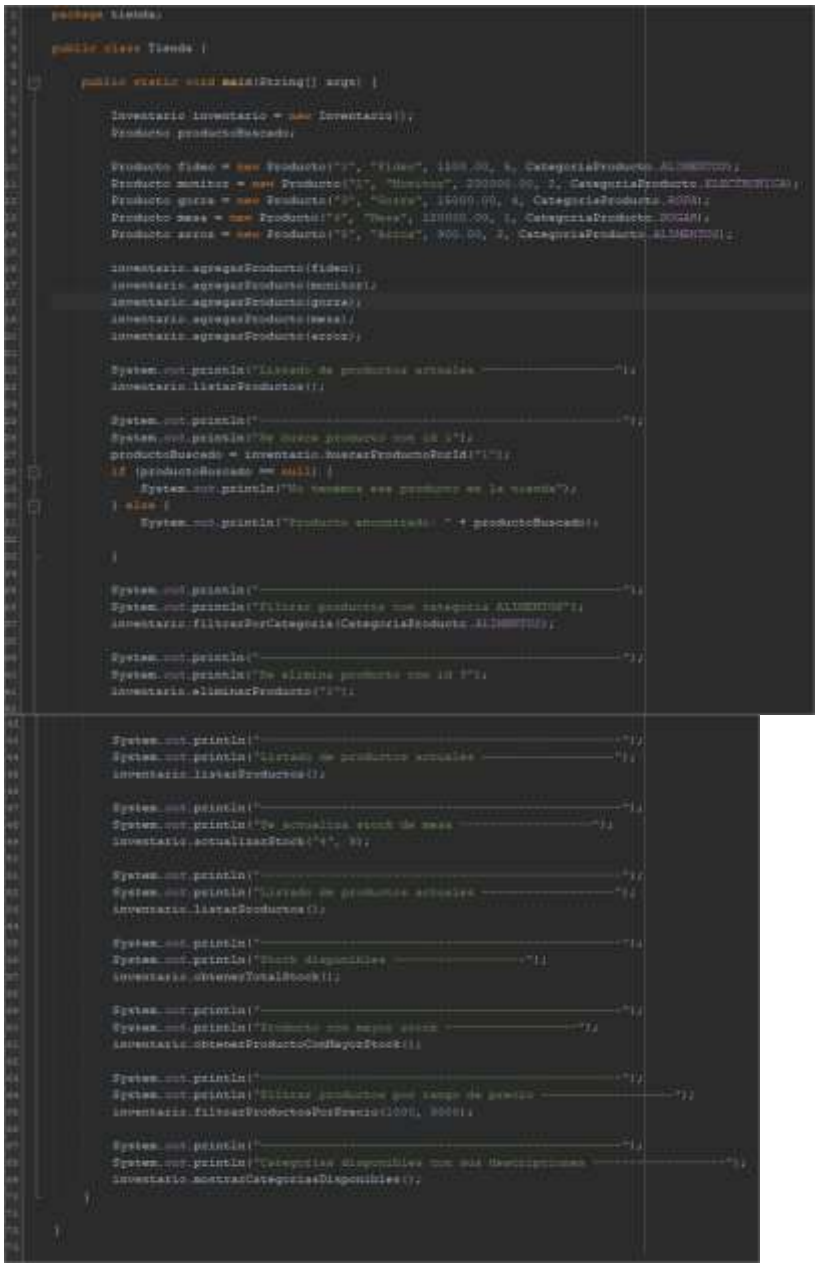
Trabajo Práctico 6 – Colecciones

Alumno: Chiavón, Facundo
Github: https://github.com/Farvon/UTN-TUPaD-P2.git

Caso Práctico 1

Se debe desarrollar un sistema de stock que permita gestionar productos en una tienda, controlando su disponibilidad, precios y categorías. La información se modelará utilizando clases, colecciones dinámicas y enumeraciones en Java.

Clase Main



Clase Producto

```java
package tienda;

public class Producto {

    private String id;
    private String nombre;
    private double precio;
    private int cantidad;
    private CategoriaProducto categoria;

    public Producto(String id, String nombre, double precio, int cantidad, CategoriaProducto categoria) {
        this.id = id;
        this.nombre = nombre;
        this.precio = precio;
        this.cantidad = cantidad;
        this.categoria = categoria;
    }

    public void mostrarInfo() {
        System.out.println(this);
    }

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

    public String getNombre() {
        return this.nombre;
    }

    public CategoriaProducto getCategoria() {
        return categoria;
    }

    public void setStock(int cantidad) {
        this.cantidad = cantidad;
        System.out.println("Stock actualizado");
    }

    public int getStock() {
        return this.cantidad;
    }

    public double getPrecio() {
        return this.precio;
    }

    @Override
    public String toString() {
        return "Producto{" + "id=" + id + ", nombre=" + nombre + ", precio=" + precio + ", cantidad=" + cantidad + ", categoria=" + categoria + '}';
    }
```

Enum CategoriaProductos

```java
package tienda;

public enum CategoriaProducto {

    ALIMENTOS("Productos comestibles"),
    ELECTRONICA("Dispositivos electrónicos"),
    ROPA("Prendas de vestir"),
    HOGAR("Artículos para el hogar");

    private final String descripcion;

    CategoriaProducto(String descripcion) {
        this.descripcion = descripcion;
    }

    public String getDescripcion() {
        return descripcion;
    }

}
```

CHIAVON, FACUNDO
COMISIÓN 11

Clase Inventario

```java
package tienda;

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

public class Inventario {

    ArrayList<Producto> productos = new ArrayList<>();

    public void agregarProducto(Producto p) {
        productos.add(p);
    }

    public void listarProductos() {
        for (Producto producto : productos) {
            System.out.println(producto);
        }
    }

    public Producto buscarProductoPorId(String id) {
        Producto productoEncontrado = null;
        Iterator<Producto> it = this.productos.iterator();
        while (it.hasNext() && productoEncontrado == null) {
            Producto next = it.next();
            if (next.getId().equalsIgnoreCase(id)) {
                productoEncontrado = next;
            }
        }
        return productoEncontrado;

    }

    public void eliminarProducto(String id) {
        Producto productoABorrar = buscarProductoPorId(id);
        productos.remove(productoABorrar);
    }

    public void actualizarStock(String id, int nuevaCantidad) {
        Producto buscado = buscarProductoPorId(id);
        if (buscado == null) {
            System.out.println("No se puede actualizar un producto que no existe");
        } else {
            buscado.setStock(nuevaCantidad);
        }
    }

    public void filtrarPorCategoria(CategoriaProducto categoria) {
        ArrayList<Producto> productosFiltrados = new ArrayList<>();
        for (Producto producto : productos) {
            if (producto.getCategoria().equals(categoria)) {
                productosFiltrados.add(producto);
            }
        }

        System.out.println("Productos con la categoria " + categoria + " : ");
        for (Producto productosFiltrado : productosFiltrados) {
            System.out.println(productosFiltrado);
        }
    }

    public void obtenerTotalStock() {
        for (Producto producto : productos) {
            System.out.println(producto.getNombre() + " : " + producto.getStock());
        }

    }

    public void obtenerProductoConMayorStock() {
        Producto prodStockMax = null;
        int stockMax = 0;

        for (Producto producto : productos) {
            if (producto.getStock() > stockMax) {
                prodStockMax = producto;
                stockMax = producto.getStock();
            }
        }

        System.out.println(prodStockMax);
    }
```

CHIAVON, FACUNDO
COMISIÓN 11

```
    public void filtrarProductosPorPrecio(double min, double max) {
        ArrayList<Producto> productosFiltrados = new ArrayList<>();
        for (Producto producto : productos) {
            if (producto.getPrecio() >= min && producto.getPrecio() <= max) {
                productosFiltrados.add(producto);
            }
        }

        System.out.println("Productos con precio entre " + min + " y " + max + " : ");
        for (Producto productosFiltrado : productosFiltrados) {
            System.out.println(productosFiltrado);
        }
    }

    public void mostrarCategoriasDisponibles() {
        for (Producto producto : productos) {
            System.out.println(producto.getNombre() + " - " + producto.getCategoria().getDescripcion());
        }
    }

}
```

Resultados

```
run:
Listado de productos actuales ------------------
Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
Producto{id=2, nombre=Monitor, precio=230000.0, cantidad=2, categoria=ELECTRONICA}
Producto{id=3, nombre=Gorra, precio=15000.0, cantidad=4, categoria=ROPA}
Producto{id=4, nombre=Mesa, precio=120000.0, cantidad=1, categoria=HOGAR}
Producto{id=5, nombre=Arroz, precio=900.0, cantidad=3, categoria=ALIMENTOS}
------------------------------------------------
Se busca producto con id 1
Producto encontrado: Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
------------------------------------------------
Filtrar productos con categoria ALIMENTOS
Productos con la categoria ALIMENTOS :
Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
Producto{id=5, nombre=Arroz, precio=900.0, cantidad=3, categoria=ALIMENTOS}
------------------------------------------------
Se elimina producto con id 3
------------------------------------------------
Listado de productos actuales ------------------
Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
Producto{id=2, nombre=Monitor, precio=230000.0, cantidad=2, categoria=ELECTRONICA}
Producto{id=4, nombre=Mesa, precio=120000.0, cantidad=1, categoria=HOGAR}
Producto{id=5, nombre=Arroz, precio=900.0, cantidad=3, categoria=ALIMENTOS}
------------------------------------------------
Se actualiza stock de mesa ------------------
Stock actualizado
------------------------------------------------
Listado de productos actuales ------------------
Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
Producto{id=2, nombre=Monitor, precio=230000.0, cantidad=2, categoria=ELECTRONICA}
Producto{id=4, nombre=Mesa, precio=120000.0, cantidad=9, categoria=HOGAR}
Producto{id=5, nombre=Arroz, precio=900.0, cantidad=3, categoria=ALIMENTOS}
------------------------------------------------
Stock disponibles ------------------
Fideo : 5
Monitor : 2
Mesa : 9
Arroz : 3
------------------------------------------------
Producto con mayor stock ------------------
Producto{id=4, nombre=Mesa, precio=120000.0, cantidad=9, categoria=HOGAR}
------------------------------------------------
Filtrar productos por rango de precio ------------------
Productos con precio entre 1000.0 y 3000.0 :
Producto{id=1, nombre=Fideo, precio=1200.0, cantidad=5, categoria=ALIMENTOS}
------------------------------------------------
Categorias disponibles con sus descripciones ------------------
Fideo - Productos comestibles
Monitor - Dispositivos electr□nicos
Mesa - Art□culos para el hogar
Arroz - Productos comestibles
BUILD SUCCESSFUL (total time: 0 seconds)
```

CHIAVON, FACUNDO
COMISIÓN 11

Caso Práctico 2

Se debe desarrollar un sistema para gestionar una biblioteca, en la cual se registren los libros disponibles y sus autores. La relación central es de composición 1 a N: una Biblioteca contiene múltiples Libros, y cada Libro pertenece obligatoriamente a una Biblioteca. Si la Biblioteca se elimina, también se eliminan sus Libros..

Clase Main

```
package biblioteca;

public class Gestor_Biblioteca {

    public static void main(String[] args) {
        Biblioteca biblioteca = new Biblioteca();

        Autor marquez = new Autor("1", "Gabriel García Márquez", "Colombiano");
        Autor cortazar = new Autor("2", "Julio Cortazar", "Argentino");
        Autor borges = new Autor("3", "Jorge Lui Borges", "Argentino");

        biblioteca.agregarLibro("978-9500745093", "Cien años de soledad", 1967, marquez);
        biblioteca.agregarLibro("950-0708167", "El amor en los tiempos del cólera", 1985, marquez);
        biblioteca.agregarLibro("978-8402076939", "Crónica de una muerte anunciada", 1981, marquez);
        biblioteca.agregarLibro("84-487-0403-7", "Rayuela", 1985, cortazar);
        biblioteca.agregarLibro("978-8420633121", "Ficciones", 1944, borges);

        System.out.println("Listado de libros actuales ------------------");
        biblioteca.listarLibros();

        System.out.println("-------------------------------------------------------");
        System.out.println("Se busca libro por ISBN = 84-487-0403-7 ------------------");
        Libro libroBuscado = biblioteca.buscarLibroPorIsbn("84-487-0403-7");
        if (libroBuscado == null) {
            System.out.println("No existe ese libro en la biblioteca");
        } else {
            System.out.println("Libro encontrado: ");
            System.out.println(libroBuscado);
        }

        System.out.println("-------------------------------------------------------");
        System.out.println("Filtramos Libros según fecha ------------------");
        biblioteca.filtrarLibrosPorAnio(1985);

        System.out.println("-------------------------------------------------------");
        System.out.println("Se elimina libro por su ISBN : 84-487-0403-7");
        biblioteca.eliminarLibro("84-487-0403-7");

        System.out.println("-------------------------------------------------------");
        System.out.println("Listado de libros actuales ------------------");
        biblioteca.listarLibros();

        System.out.println("-------------------------------------------------------");
        biblioteca.obtenerCantidadLibros();

        System.out.println("-------------------------------------------------------");
        biblioteca.mostrarAutoresDisponibles();

    }

}
```

Clase Autor

```java
package biblioteca;

public class Autor {

    private String id;
    private String nombre;
    private String nacionalidad;

    public Autor(String id, String nombre, String nacionalidad) {
        this.id = id;
        this.nombre = nombre;
        this.nacionalidad = nacionalidad;
    }

    public void mostrarInfo() {
        System.out.println(this);
    }

    public String getNombre() {
        return this.nombre;
    }

    @Override
    public String toString() {
        return "Autor{" + "id=" + id + ", nombre=" + nombre + ", nacionalidad=" + nacionalidad + '}';
    }

}
```

Clase Libro

```java
package biblioteca;

public class Libro {

    private String isbn;
    private String titulo;
    private int anioPublicacion;
    private Autor autor;

    public Libro(String isbn, String titulo, int anioPublicacion, Autor autor) {
        this.isbn = isbn;
        this.titulo = titulo;
        this.anioPublicacion = anioPublicacion;
        this.autor = autor;
    }

    public String getIsbn() {
        return this.isbn;
    }

    public int getAnio() {
        return this.anioPublicacion;
    }

    public Autor getAutor() {
        return this.autor;
    }

    @Override
    public String toString() {
        return "Libro{" + "isbn=" + isbn + ", titulo=" + titulo + ", anioPublicacion=" + anioPublicacion + ", autor=" + autor + '}';
    }

}
```

Clase Biblioteca

```java
package biblioteca;

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

public class Biblioteca {

    private String nombre;
    private List<Libro> libros = new ArrayList<>();

    public void agregarLibro(String isbn, String titulo, int anioPublicacion, Autor autor) {
        Libro libro = new Libro(isbn, titulo, anioPublicacion, autor);
        libros.add(libro);

    }

    public void listarLibros() {
        for (Libro libro : libros) {
            System.out.println(libro);

        }
    }

    public Libro buscarLibroPorIsbn(String isbn) {
        Libro libroEncontrado = null;
        Iterator<Libro> it = this.libros.iterator();
        while (it.hasNext() && libroEncontrado == null) {
            Libro next = it.next();
            if (next.getIsbn().equalsIgnoreCase(isbn)) {
                libroEncontrado = next;
            }
        }
        return libroEncontrado;

    }

    public void filtrarLibrosPorAnio(int anio) {
        ArrayList<Libro> librosFiltrados = new ArrayList<>();
        for (Libro libro : libros) {
            if (libro.getAnio() == anio) {
                librosFiltrados.add(libro);
            }
        }
        System.out.println("Libros publicados en " + anio + " : ");
        for (Libro librosFiltrado : librosFiltrados) {
            System.out.println(librosFiltrado);
        }
    }

    public void eliminarLibro(String isbn) {
        Libro libroBuscado = buscarLibroPorIsbn(isbn);
        if (libroBuscado == null) {
            System.out.println("No existe el libro");
        } else {
            libros.remove(libroBuscado);
            System.out.println("Libro Borrado");
        }

    }

    public void obtenerCantidadLibros() {
        System.out.println("Cantidad de libros en Biblioteca: " + libros.size());
    }

    public void mostrarAutoresDisponibles() {
        System.out.println("Autores disponibles");

        ArrayList<Autor> autoresDisponibles = new ArrayList<>();

        for (Libro libro : libros) {
            Autor autor = libro.getAutor();
            if (!autoresDisponibles.contains(autor)) {
                autoresDisponibles.add(autor);
            }
        }

        for (Autor autor : autoresDisponibles) {
            System.out.println(autor.getNombre());
        }
    }
}
```

Resultados

```
run:
Listado de libros actuales ------------------
Libro{isbn=978-9500745093, titulo=Cien a◆os de soledad, anioPublicacion=1967, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=950-0708147, titulo=El amor en los tiempos del c◆lera, anioPublicacion=1985, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=978-8402074939, titulo=Cr◆nica de una muerte anunciada, anioPublicacion=1981, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=84-487-0403-7, titulo=Rayuela, anioPublicacion=1985, autor=Autor{id=2, nombre=Julio Cortazar, nacionalidad=Argentino}}
Libro{isbn=978-8420633121, titulo=Ficciones, anioPublicacion=1944, autor=Autor{id=3, nombre=Jorge Lui Borges, nacionalidad=Argentino}}
---------------------------------------------------
Se busca libro por ISBN = 84-487-0403-7 ------------------
Libro encontrado:
Libro{isbn=84-487-0403-7, titulo=Rayuela, anioPublicacion=1985, autor=Autor{id=2, nombre=Julio Cortazar, nacionalidad=Argentino}}
---------------------------------------------------
Filtramos Libros seg◆n fecha ------------------
Libros publicados en 1985 :
Libro{isbn=950-0708147, titulo=El amor en los tiempos del c◆lera, anioPublicacion=1985, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=84-487-0403-7, titulo=Rayuela, anioPublicacion=1985, autor=Autor{id=2, nombre=Julio Cortazar, nacionalidad=Argentino}}
---------------------------------------------------
Se elimina libro por su ISBN : 84-487-0403-7
Libro Borrado
---------------------------------------------------
Listado de libros actuales ------------------
Libro{isbn=978-9500745093, titulo=Cien a◆os de soledad, anioPublicacion=1967, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=950-0708147, titulo=El amor en los tiempos del c◆lera, anioPublicacion=1985, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=978-8402074939, titulo=Cr◆nica de una muerte anunciada, anioPublicacion=1981, autor=Autor{id=1, nombre=Gabriel Garc◆a M◆rquez, nacionalidad=Colombiano}}
Libro{isbn=978-8420633121, titulo=Ficciones, anioPublicacion=1944, autor=Autor{id=3, nombre=Jorge Lui Borges, nacionalidad=Argentino}}
---------------------------------------------------
Cantidad de libros en Biblioteca: 4
---------------------------------------------------
Autores disponibles
Gabriel Garc◆a M◆rquez
Jorge Lui Borges
BUILD SUCCESSFUL (total time: 0 seconds)
```

Caso Práctico 3

Se debe modelar un sistema académico donde un Profesor dicta muchos Cursos y cada Curso tiene exactamente un Profesor responsable.

Clase Main

```
package Academia;

public class Academia {

    public static void main(String[] args) {
        Universidad universidad = new Universidad();

        Profesor facu = new Profesor("1", "Facundo", "Computacion");
        Profesor cris = new Profesor("2", "Cristian", "Matemática");
        Profesor ana = new Profesor("3", "Anahi", "RRHH");

        Curso programacion = new Curso("1-0020", "Programacion 1");
        Curso organizacion = new Curso("3-0010", "Organizacion Empresarial");
        Curso ecuaciones = new Curso("2-0220", "Ecuaciones Lineales");
        Curso bbdd = new Curso("1-0025", "Base de Datos");
        Curso liquidaciones = new Curso("3-0005", "Liquidaciones");

        universidad.agregarProfesor(facu);
        universidad.agregarProfesor(cris);
        universidad.agregarProfesor(ana);

        universidad.agregarCurso(programacion);
        universidad.agregarCurso(organizacion);
        universidad.agregarCurso(ecuaciones);
        universidad.agregarCurso(bbdd);
        universidad.agregarCurso(liquidaciones);

        System.out.println("Lista de Profesores ----------");
        universidad.listarProfesores();

        System.out.println("Lista de Cursos ----------");
        universidad.listarCursos();

        System.out.println("------------------------------");
        System.out.println("Se asignan Profesores a Cursos ");
        universidad.asignarProfesorACurso("1-0020", "1");
        universidad.asignarProfesorACurso("3-0010", "3");
        universidad.asignarProfesorACurso("2-0220", "2");
        universidad.asignarProfesorACurso("1-0025", "1");
        universidad.asignarProfesorACurso("3-0005", "3");
        System.out.println("------------------------------");

        System.out.println("Lista de Profesores ----------");
        universidad.listarProfesores();

        System.out.println("Lista de Cursos ----------");
        universidad.listarCursos();

        System.out.println("------------------------------");
        System.out.println("Se cambia curso 3-0010 a Profesor 2");
        universidad.asignarProfesorACurso("3-0010", "2");
        System.out.println("------------------------------");

        System.out.println("Lista de Profesores ----------");
        universidad.listarProfesores();

        System.out.println("Lista de Cursos ----------");
        universidad.listarCursos();

        System.out.println("------------------------------");
        System.out.println("Se elimina Profesor 2");
        universidad.eliminarProfesor("2");
        System.out.println("------------------------------");

        System.out.println("Lista de Profesores ----------");
        universidad.listarProfesores();

        System.out.println("Lista de Cursos ----------");
        universidad.listarCursos();

    }

}
```

Clase Universidad

```java
package Academia;

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

public class Universidad {

    private String nombre;
    private List<Profesor> profesores = new ArrayList<>();
    private List<Curso> cursos = new ArrayList<>();

    public void agregarProfesor(Profesor p) {
        if (p != null) {
            profesores.add(p);
        } else {
            System.out.println("Debe proporcionar un valor válido");
        }
    }

    public void agregarCurso(Curso c) {
        if (c != null) {
            cursos.add(c);
        } else {
            System.out.println("Debe proporcionar un valor válido");
        }
    }

    // Usa setProfesor del curso
    public void asignarProfesorACurso(String codigoCurso, String idProfesor) {
        Profesor profesor = buscarProfesorPorId(idProfesor);
        Curso curso = buscarCursoPorCodigo(codigoCurso);

        Profesor profeViejo = curso.getProfesor();
        if (profeViejo != null) {
            profeViejo.eliminarCurso(curso);
        }

        curso.setProfesor(profesor);

    }

    public void listarProfesores() {
        for (Profesor profesor : profesores) {
            System.out.println(profesor);
        }
    }

    public void listarCursos() {
        for (Curso curso : cursos) {
            System.out.println(curso);
        }
    }
```

```java
    public Profesor buscarProfesorPorId(String id) {
        Profesor profesorBuscado = null;
        Iterator<Profesor> it = this.profesores.iterator();
        while (it.hasNext() && profesorBuscado == null) {
            Profesor next = it.next();
            if (next.getId().equalsIgnoreCase(id)) {
                profesorBuscado = next;
            }
        }
        return profesorBuscado;
    }

    public Curso buscarCursoPorCodigo(String codigo) {
        Curso cursoBuscado = null;
        Iterator<Curso> it = this.cursos.iterator();
        while (it.hasNext() && cursoBuscado == null) {
            Curso next = it.next();
            if (next.getCodigo().equalsIgnoreCase(codigo)) {
                cursoBuscado = next;
            }
        }
        return cursoBuscado;
    }

    //Debe romper la relación con su profesor si la hubiera.
    public void eliminarCurso(String codigo) {
        Curso cursoBuscado = buscarCursoPorCodigo(codigo);
        Profesor profesor = cursoBuscado.getProfesor();

        if (profesor != null) {
            profesor.eliminarCurso(cursoBuscado);
        }

    }

    //Antes de remover, dejar null los cursos que dictaba.
    public void eliminarProfesor(String id) {
        Profesor profesor = buscarProfesorPorId(id);
        for (Curso curso : cursos) {
            if (curso.getProfesor() == profesor) {
                curso.eliminarProfesor();
            }
        }
        profesores.remove(profesor);

    }
}
```

Clase Profesor

```java
package Academia;

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

public class Profesor {

    private String id;
    private String nombre;
    private String especialidad;
    private List<Curso> cursos = new ArrayList<>();

    public Profesor(String id, String nombre, String especialidad) {
        this.id = id;
        this.nombre = nombre;
        this.especialidad = especialidad;
    }

    public String getNombre() {
        return this.nombre;
    }

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

    public Profesor getProfesor() {
        return this;
    }

    public void agregarCurso(Curso c) {
        if (!cursos.contains(c)) {
            cursos.add(c);
            c.setProfesor(this);
        } else {
            System.out.println("El profesor ya tiene asignado ese curso");
        }
    }

    public void eliminarCurso(Curso c) {
        if (cursos.contains(c)) {
            cursos.remove(c);
            if (c.getProfesor() == this) {
                c.eliminarProfesor();
            }
        } else {
            System.out.println("El profesor no tiene asignado ese curso");
        }

    }

    public Curso buscarCurso(Curso c) {
        Curso cursoBuscado = null;
        Iterator<Curso> it = this.cursos.iterator();
        while (it.hasNext() && cursoBuscado == null) {
            Curso next = it.next();
            if (next == c) {
                cursoBuscado = next;
            }
        }
        return cursoBuscado;
    }

    public void listarCursos() {
        for (Curso curso : cursos) {
            System.out.println(curso.getCodigo() + " - " + curso.getNombre());

        }
    }

    public void mostrarInfo() {
        System.out.println(this);
    }

    @Override
    public String toString() {
        return "Profesor{" + "id=" + id + ", nombre=" + nombre + ", especialidad=" + especialidad + ", cursos=" + cursos.size() + '}';
    }

}
```

Clase Curso

```java
package Academia;

public class Curso {

    private String codigo;
    private String nombre;
    private Profesor profesor;

    public Curso(String codigo, String nombre) {
        this.codigo = codigo;
        this.nombre = nombre;
    }

    public String getNombre() {
        return this.nombre;
    }

    public String getCodigo() {
        return this.codigo;
    }

    public void setProfesor(Profesor p) {
        this.profesor = p;
        Curso curso = p.buscarCurso(this);
        if (curso == null) {
            profesor.agregarCurso(this);
        }

    }

    public Profesor getProfesor() {
        return this.profesor;
    }

    public void eliminarProfesor() {
        this.profesor = null;
    }

    public Curso getCurso() {
        return this;
    }

    public void mostrarInfo() {
        System.out.println(this);
    }

    public String mostrarProfesor() {
        if (this.profesor == null) {
            return "";
        } else {
            return ", profesor= " + this.profesor.getNombre();
        }
    }

    @Override
    public String toString() {
        return "Curso{" + "codigo=" + codigo + ", nombre=" + nombre + mostrarProfesor() + "}";
    }
```

Resultados

```
run:
Lista de Profesores ------------------
Profesor{id=1, nombre=Facundo, especialidad=Computaci□n, cursos=0}
Profesor{id=2, nombre=Cristian, especialidad=Matem□tica, cursos=0}
Profesor{id=3, nombre=Anah□, especialidad=RRHH, cursos=0}
Lista de Cursos ------------------
Curso{codigo=1-0020, nombre=Programaci□n 1}
Curso{codigo=3-0010, nombre=Organizaci□n Empresarial}
Curso{codigo=2-0220, nombre=Ecuaciones Lineales}
Curso{codigo=1-0025, nombre=Base de Datos}
Curso{codigo=3-0005, nombre=Liquidaciones}
----------------------------
Se asignan Profesores a Cursos
----------------------------
Lista de Profesores ------------------
Profesor{id=1, nombre=Facundo, especialidad=Computaci□n, cursos=2}
Profesor{id=2, nombre=Cristian, especialidad=Matem□tica, cursos=1}
Profesor{id=3, nombre=Anah□, especialidad=RRHH, cursos=2}
Lista de Cursos ------------------
Curso{codigo=1-0020, nombre=Programaci□n 1, profesor= Facundo}
Curso{codigo=3-0010, nombre=Organizaci□n Empresarial, profesor= Anah□}
Curso{codigo=2-0220, nombre=Ecuaciones Lineales, profesor= Cristian}
Curso{codigo=1-0025, nombre=Base de Datos, profesor= Facundo}
Curso{codigo=3-0005, nombre=Liquidaciones, profesor= Anah□}
----------------------------
Se cambia curso 3-0010 a Profesor 2
----------------------------
Lista de Profesores ------------------
Profesor{id=1, nombre=Facundo, especialidad=Computaci□n, cursos=2}
Profesor{id=2, nombre=Cristian, especialidad=Matem□tica, cursos=2}
Profesor{id=3, nombre=Anah□, especialidad=RRHH, cursos=1}
Lista de Cursos ------------------
Curso{codigo=1-0020, nombre=Programaci□n 1, profesor= Facundo}
Curso{codigo=3-0010, nombre=Organizaci□n Empresarial, profesor= Cristian}
Curso{codigo=2-0220, nombre=Ecuaciones Lineales, profesor= Cristian}
Curso{codigo=1-0025, nombre=Base de Datos, profesor= Facundo}
Curso{codigo=3-0005, nombre=Liquidaciones, profesor= Anah□}
----------------------------
Se elimina Profesor 2
----------------------------
Lista de Profesores ------------------
Profesor{id=1, nombre=Facundo, especialidad=Computaci□n, cursos=2}
Profesor{id=3, nombre=Anah□, especialidad=RRHH, cursos=1}
Lista de Cursos ------------------
Curso{codigo=1-0020, nombre=Programaci□n 1, profesor= Facundo}
Curso{codigo=3-0010, nombre=Organizaci□n Empresarial}
Curso{codigo=2-0220, nombre=Ecuaciones Lineales}
Curso{codigo=1-0025, nombre=Base de Datos, profesor= Facundo}
Curso{codigo=3-0005, nombre=Liquidaciones, profesor= Anah□}
BUILD SUCCESSFUL (total time: 0 seconds)
```