

PROGRAMACIÓN II

Trabajo Práctico 6: Colecciones y Sistema de Stock

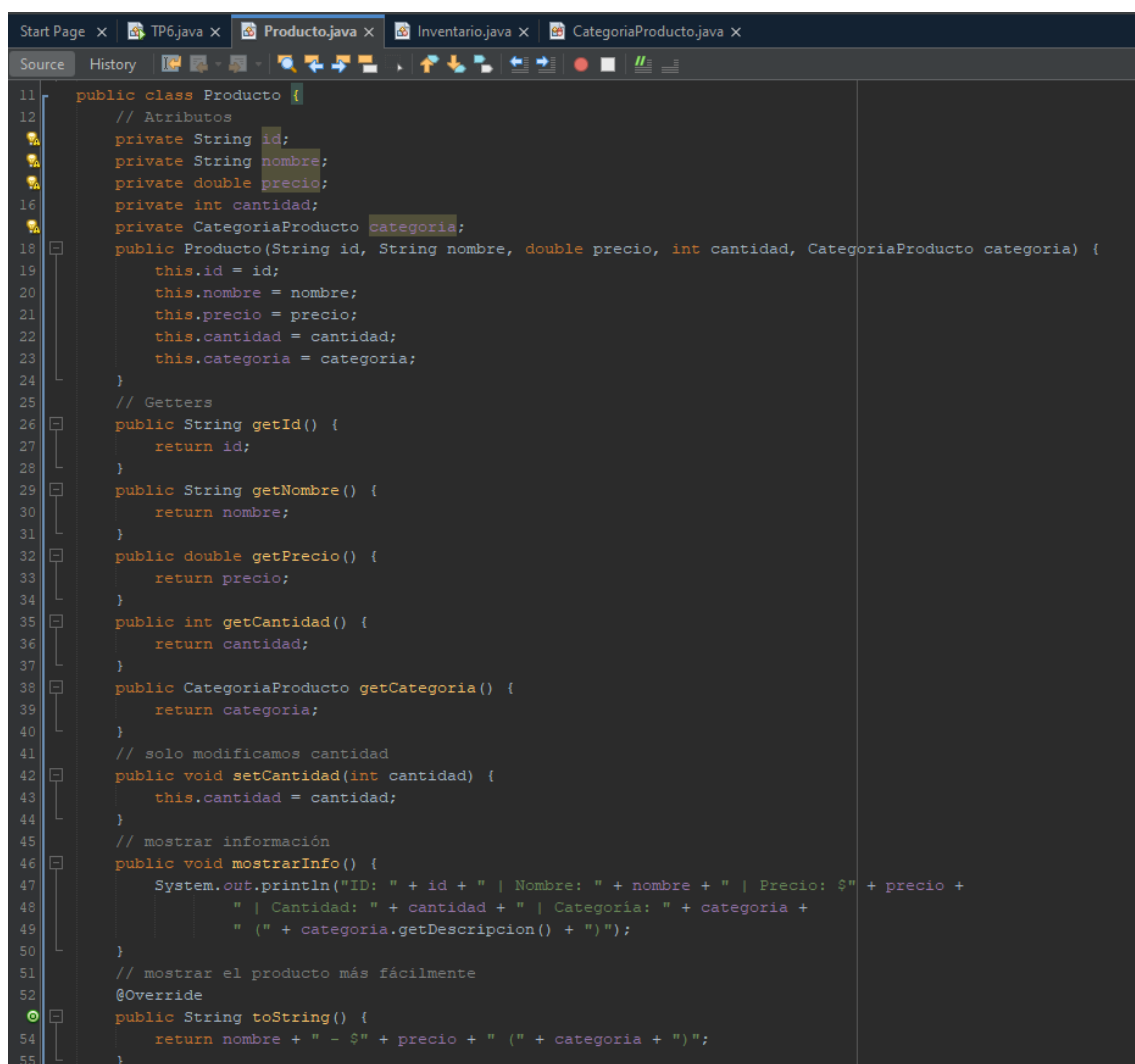
Alumno: Matias Luna

GitHub: <https://github.com/matiaslunaa?tab=repositories>

Caso Práctico 1

Clases a implementar

Clase Producto:



```
11 public class Producto {
12     // Atributos
13     private String id;
14     private String nombre;
15     private double precio;
16     private int cantidad;
17     private CategoriaProducto categoria;
18     public Producto(String id, String nombre, double precio, int cantidad, CategoriaProducto categoria) {
19         this.id = id;
20         this.nombre = nombre;
21         this.precio = precio;
22         this.cantidad = cantidad;
23         this.categoria = categoria;
24     }
25     // Getters
26     public String getId() {
27         return id;
28     }
29     public String getNombre() {
30         return nombre;
31     }
32     public double getPrecio() {
33         return precio;
34     }
35     public int getCantidad() {
36         return cantidad;
37     }
38     public CategoriaProducto getCategoria() {
39         return categoria;
40     }
41     // solo modificamos cantidad
42     public void setCantidad(int cantidad) {
43         this.cantidad = cantidad;
44     }
45     // mostrar información
46     public void mostrarInfo() {
47         System.out.println("ID: " + id + " | Nombre: " + nombre + " | Precio: $" + precio +
48             " | Cantidad: " + cantidad + " | Categoria: " + categoria +
49             " (" + categoria.getDescripcion() + ")");
50     }
51     // mostrar el producto más fácilmente
52     @Override
53     public String toString() {
54         return nombre + " - $" + precio + " (" + categoria + ")";
55     }
56 }
```

Clase Inventario:

```
Start Page x TP6.java x Producto.java x Inventario.java x CategoriaProducto.java x
Source History
10  */
11  public class Inventario {
12      private ArrayList<Producto> productos;
13      public Inventario() {
14          productos = new ArrayList<>();
15      }
16      // 1. Agregar producto
17      public void agregarProducto(Producto p) {
18          productos.add(p);
19      }
20      // 2. Listar productos
21      public void listarProductos() {
22          for (Producto p : productos) {
23              p.mostrarInfo();
24          }
25      }
26      // 3. Buscar por ID
27      public Producto buscarProductoPorId(String id) {
28          for (Producto p : productos) {
29              if (p.getId().equalsIgnoreCase(id)) {
30                  return p;
31              }
32          }
33          return null;
34      }
35      // 4. Eliminar por ID
36      public void eliminarProducto(String id) {
37          productos.removeIf(p -> p.getId().equalsIgnoreCase(id));
38      }
39      // 5. Actualizar stock
40      public void actualizarStock(String id, int nuevaCantidad) {
41          Producto p = buscarProductoPorId(id);
42          if (p != null) {
43              p.setCantidad(nuevaCantidad);
44          }
45      }
46      // 6. Filtrar por categoría
47      public void filtrarPorCategoria(CategoriaProducto categoria) {
48          for (Producto p : productos) {
49              if (p.getCategoria() == categoria) {
50                  p.mostrarInfo();
51              }
52          }
53      }
}
```

```

54 // 7. Total de stock
55 public int obtenerTotalStock() {
56     int total = 0;
57     for (Producto p : productos) {
58         total += p.getCantidad();
59     }
60     return total;
61 }
62 // 8. Producto con mayor stock
63 public Producto obtenerProductoConMayorStock() {
64     if (productos.isEmpty()) return null;
65
66     Producto max = productos.get(0);
67     for (Producto p : productos) {
68         if (p.getCantidad() > max.getCantidad()) {
69             max = p;
70         }
71     }
72     return max;
73 }
74 // 9. Filtrar por precio
75 public void filtrarProductosPorPrecio(double min, double max) {
76     for (Producto p : productos) {
77         if (p.getPrecio() >= min && p.getPrecio() <= max) {
78             p.mostrarInfo();
79         }
80     }
81 }
82 // 10. Mostrar categorías disponibles
83 public void mostrarCategoriasDisponibles() {
84     for (CategoriaProducto c : CategoriaProducto.values()) {
85         System.out.println(c + ": " + c.getDescripcion());
86     }
87 }
88 }
89

```

Categoría Producto:

```

Start Page x TP6.java x Producto.java x Inventario.java x CategoriaProducto.java x
Source History
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5   package com.mycompany.tp6;
6
7   /**
8    *
9    * @author Mati
10   */
11  public enum CategoriaProducto {
12      ALIMENTOS("Productos comestibles"),
13      ELECTRONICA("Dispositivos electrónicos"),
14      ROPA("Prendas de vestir"),
15      HOGAR("Artículos para el hogar");
16      private final String descripcion;
17      // Constructor del enum
18      CategoriaProducto(String descripcion) {
19          this.descripcion = descripcion;
20      }
21      // Método para obtener la descripción
22      public String getDescripcion() {
23          return descripcion;
24      }
25  }
26

```

MAIN:

```
Start Page x TP6.java x Producto.java x Inventario.java x CategoriaProducto.java x
Source History
11 public class TP6 {
12     public static void main(String[] args) {
13         Inventario inventario = new Inventario();
14         // 1. Crear productos
15         Producto p1 = new Producto("A1", "Leche", 1200, 10, CategoriaProducto.ALIMENTOS);
16         Producto p2 = new Producto("B2", "Televisor", 250000, 5, CategoriaProducto.ELECTRONICA);
17         Producto p3 = new Producto("C3", "Campera", 18000, 15, CategoriaProducto.ROPA);
18         Producto p4 = new Producto("D4", "Taza", 1500, 40, CategoriaProducto.HOGAR);
19         Producto p5 = new Producto("E5", "Cereal", 2100, 25, CategoriaProducto.ALIMENTOS);
20         // 2. Agregar productos al inventario
21         inventario.agregarProducto(p1);
22         inventario.agregarProducto(p2);
23         inventario.agregarProducto(p3);
24         inventario.agregarProducto(p4);
25         inventario.agregarProducto(p5);
26         // 3. Listar todos los productos
27         System.out.println("\n--- Listado completo ---");
28         inventario.listarProductos();
29         // 4. Buscar producto por ID
30         System.out.println("\n--- Buscar producto por ID (C3) ---");
31         Producto encontrado = inventario.buscarProductoPorId("C3");
32         if (encontrado != null) encontrado.mostrarInfo();
33         // 5. Filtrar por categoria
34         System.out.println("\n--- Filtrar por categoria: ALIMENTOS ---");
35         inventario.filtrarPorCategoria(CategoriaProducto.ALIMENTOS);
36         // 6. Eliminar un producto
37         System.out.println("\n--- Eliminar producto con ID B2 ---");
38         inventario.eliminarProducto("B2");
39         inventario.listarProductos();
40         // 7. Actualizar stock
41         System.out.println("\n--- Actualizar stock de A1 a 30 unidades ---");
42         inventario.actualizarStock("A1", 30);
43         inventario.buscarProductoPorId("A1").mostrarInfo();
44         // 8. Total de stock
45         System.out.println("\n--- Total de stock disponible ---");
46         System.out.println("Total: " + inventario.obtenerTotalStock());
47         // 9. Producto con mayor stock
48         System.out.println("\n--- Producto con mayor stock ---");
49         Producto mayor = inventario.obtenerProductoConMayorStock();
50         if (mayor != null) mayor.mostrarInfo();
51         // 10. Filtro por precio
52         System.out.println("\n--- Productos con precio entre 1000 y 3000 ---");
53         inventario.filtrarProductosPorPrecio(1000, 3000);
54
55         // 11. Categorías disponibles
56         System.out.println("\n--- Categorías disponibles ---");
57         inventario.mostrarCategoriasDisponibles();
58     }
59 }
```

Ejecutamos y por consola nos devuelve:

```
Start Page x TP6.java x Producto.java x Inventario.java x CategoriaProducto.java x
Source History
company.com:1234
Notifications Output - Run (TP6) x

--- exec:3.1.0:exec (default-cli) @ TP6 ---

--- Listado completo ---
ID: A1 | Nombre: Leche | Precio: $1200.0 | Cantidad: 10 | Categoria: ALIMENTOS (Productos comestibles)
ID: B2 | Nombre: Televisor | Precio: $250000.0 | Cantidad: 5 | Categoria: ELECTRONICA (Dispositivos electronicos)
ID: C3 | Nombre: Campera | Precio: $18000.0 | Cantidad: 15 | Categoria: ROPA (Prendas de vestir)
ID: D4 | Nombre: Taza | Precio: $1500.0 | Cantidad: 40 | Categoria: HOGAR (Articulos para el hogar)
ID: E5 | Nombre: Cereal | Precio: $2100.0 | Cantidad: 25 | Categoria: ALIMENTOS (Productos comestibles)

--- Buscar producto por ID (C3) ---
ID: C3 | Nombre: Campera | Precio: $18000.0 | Cantidad: 15 | Categoria: ROPA (Prendas de vestir)

--- Filtrar por categoria: ALIMENTOS ---
ID: A1 | Nombre: Leche | Precio: $1200.0 | Cantidad: 10 | Categoria: ALIMENTOS (Productos comestibles)
ID: E5 | Nombre: Cereal | Precio: $2100.0 | Cantidad: 25 | Categoria: ALIMENTOS (Productos comestibles)

--- Eliminar producto con ID B2 ---
ID: A1 | Nombre: Leche | Precio: $1200.0 | Cantidad: 10 | Categoria: ALIMENTOS (Productos comestibles)
ID: C3 | Nombre: Campera | Precio: $18000.0 | Cantidad: 15 | Categoria: ROPA (Prendas de vestir)
ID: D4 | Nombre: Taza | Precio: $1500.0 | Cantidad: 40 | Categoria: HOGAR (Articulos para el hogar)
ID: E5 | Nombre: Cereal | Precio: $2100.0 | Cantidad: 25 | Categoria: ALIMENTOS (Productos comestibles)

--- Actualizar stock de A1 a 30 unidades ---
ID: A1 | Nombre: Leche | Precio: $1200.0 | Cantidad: 30 | Categoria: ALIMENTOS (Productos comestibles)

--- Total de stock disponible ---
Total: 110

--- Producto con mayor stock ---
ID: D4 | Nombre: Taza | Precio: $1500.0 | Cantidad: 40 | Categoria: HOGAR (Articulos para el hogar)

--- Productos con precio entre 1000 y 3000 ---
ID: A1 | Nombre: Leche | Precio: $1200.0 | Cantidad: 30 | Categoria: ALIMENTOS (Productos comestibles)
ID: D4 | Nombre: Taza | Precio: $1500.0 | Cantidad: 40 | Categoria: HOGAR (Articulos para el hogar)
ID: E5 | Nombre: Cereal | Precio: $2100.0 | Cantidad: 25 | Categoria: ALIMENTOS (Productos comestibles)

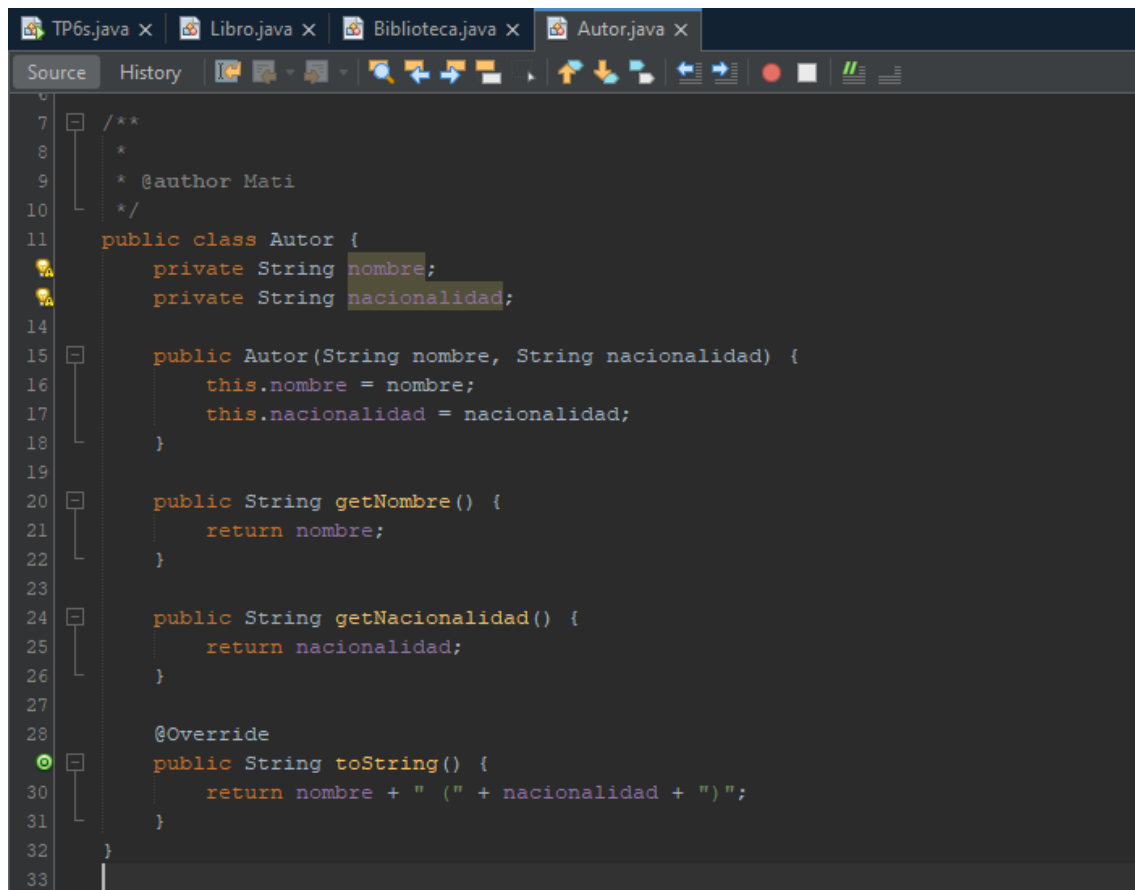
--- Categorias disponibles ---
ALIMENTOS: Productos comestibles
ELECTRONICA: Dispositivos electronicos
ROPA: Prendas de vestir
HOGAR: Articulos para el hogar

-----
BUILD SUCCESS
-----

Total time: 2.992 s
Finished at: 2025-10-23T20:59:10-03:00
-----
```

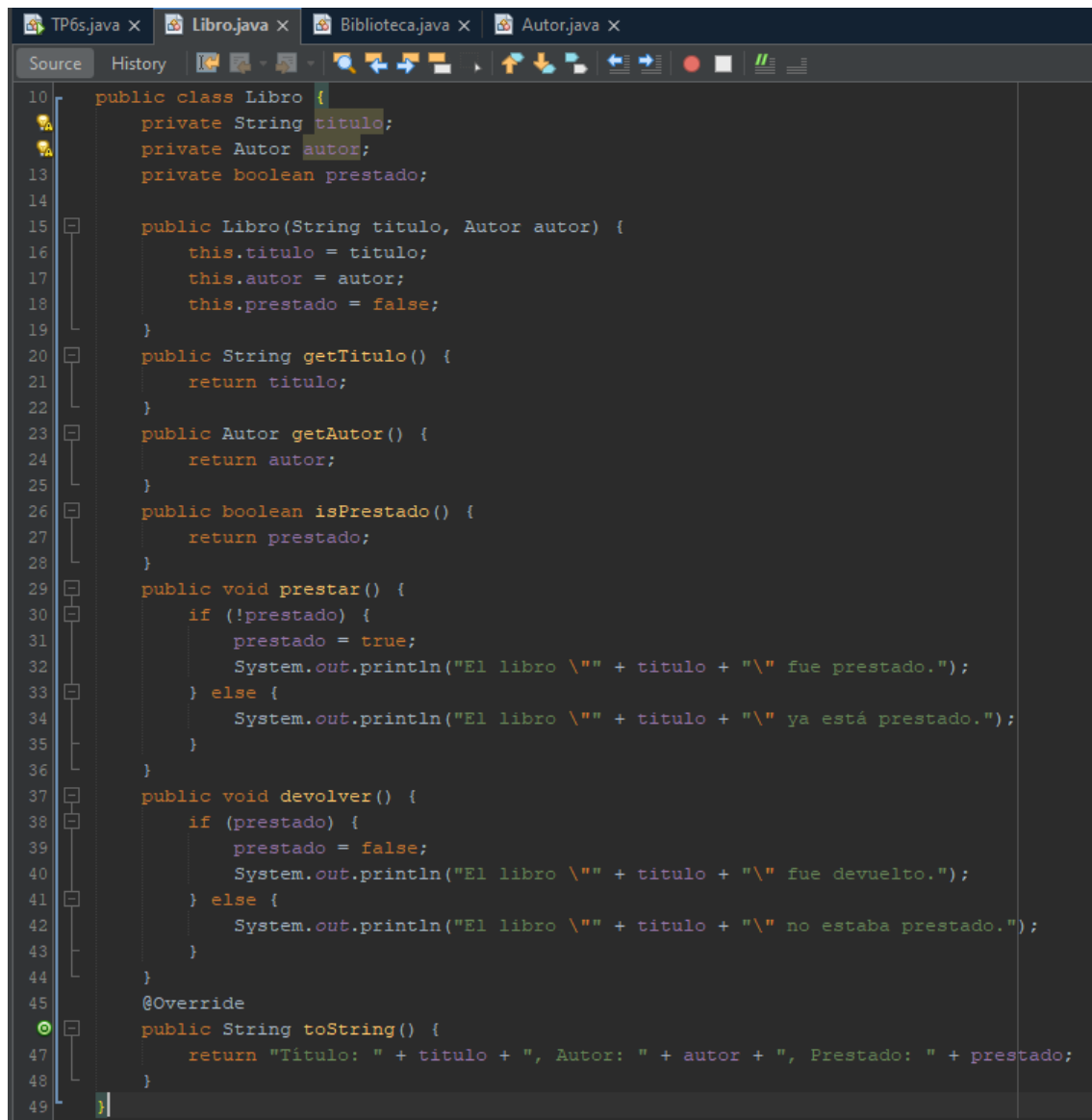
Nuevo Ejercicio Propuesto 2: Biblioteca y Libros

Clase Autor:



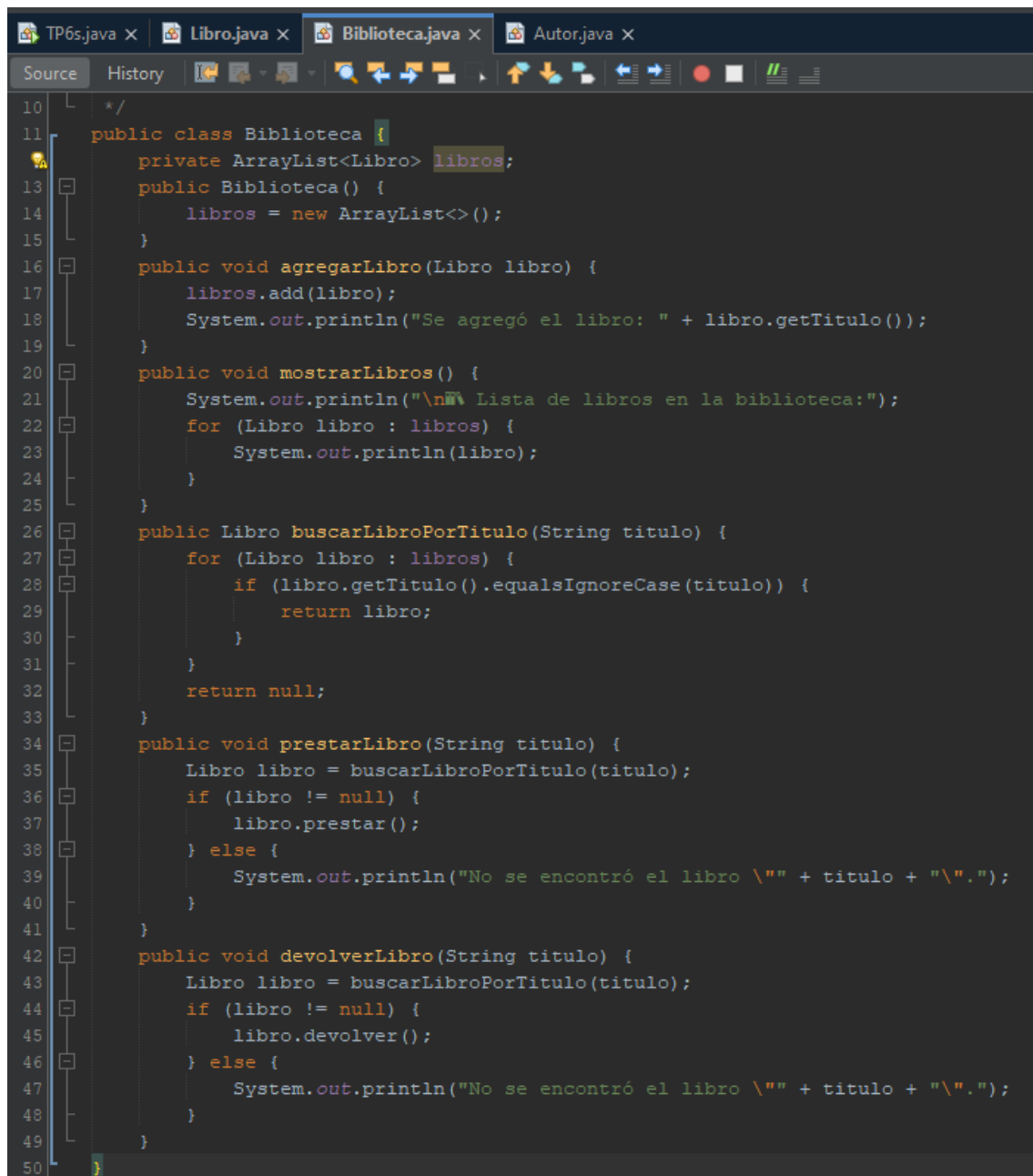
```
6
7  /**
8   *
9   * @author Mati
10  */
11  public class Autor {
12      private String nombre;
13      private String nacionalidad;
14
15      public Autor(String nombre, String nacionalidad) {
16          this.nombre = nombre;
17          this.nacionalidad = nacionalidad;
18      }
19
20      public String getNombre() {
21          return nombre;
22      }
23
24      public String getNacionalidad() {
25          return nacionalidad;
26      }
27
28      @Override
29      public String toString() {
30          return nombre + " (" + nacionalidad + ")";
31      }
32  }
33
```

Clase Libro:



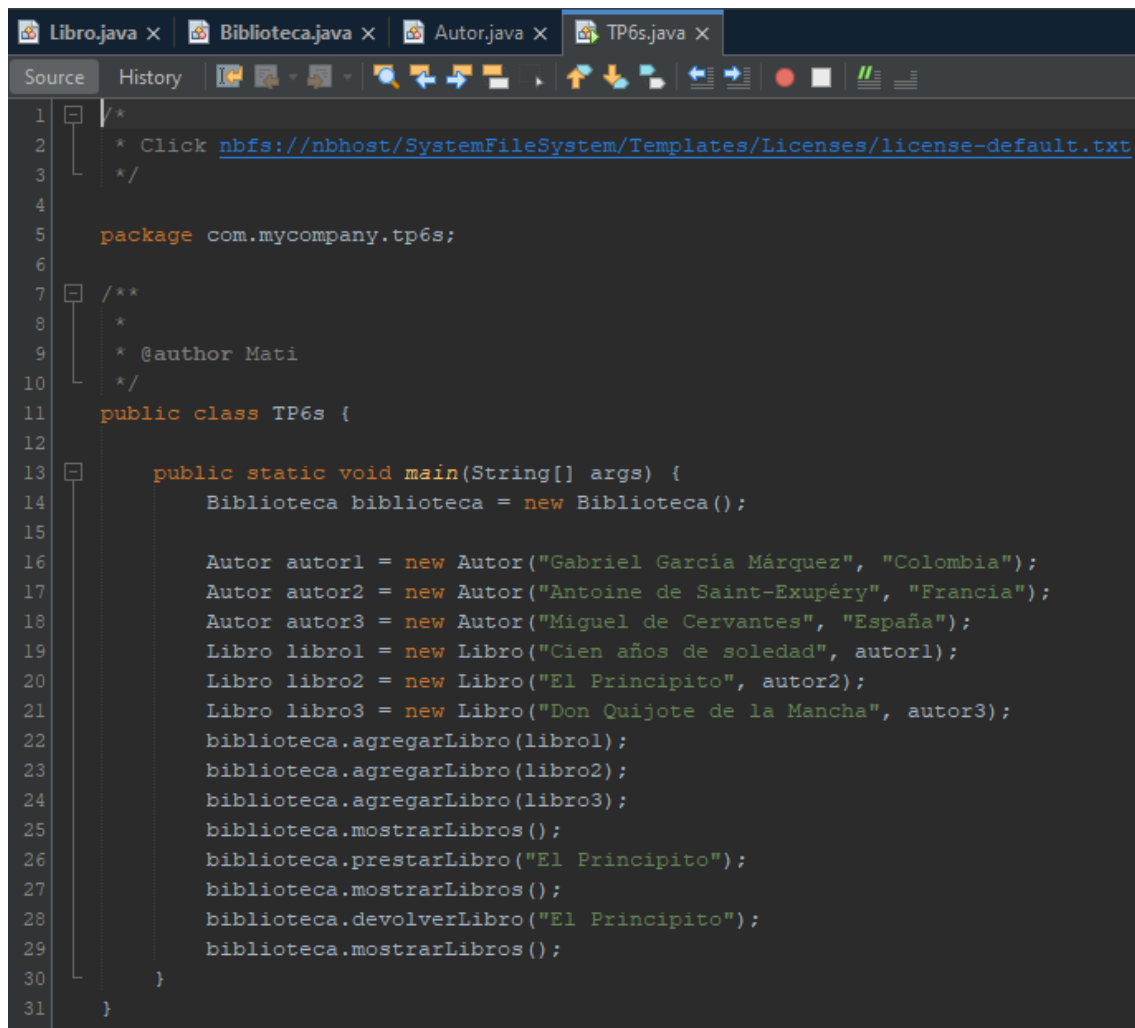
```
10 public class Libro {
11     private String titulo;
12     private Autor autor;
13     private boolean prestado;
14
15     public Libro(String titulo, Autor autor) {
16         this.titulo = titulo;
17         this.autor = autor;
18         this.prestado = false;
19     }
20     public String getTitulo() {
21         return titulo;
22     }
23     public Autor getAutor() {
24         return autor;
25     }
26     public boolean isPrestado() {
27         return prestado;
28     }
29     public void prestar() {
30         if (!prestado) {
31             prestado = true;
32             System.out.println("El libro \"" + titulo + "\" fue prestado.");
33         } else {
34             System.out.println("El libro \"" + titulo + "\" ya está prestado.");
35         }
36     }
37     public void devolver() {
38         if (prestado) {
39             prestado = false;
40             System.out.println("El libro \"" + titulo + "\" fue devuelto.");
41         } else {
42             System.out.println("El libro \"" + titulo + "\" no estaba prestado.");
43         }
44     }
45     @Override
46     public String toString() {
47         return "Titulo: " + titulo + ", Autor: " + autor + ", Prestado: " + prestado;
48     }
49 }
```

Clase Biblioteca:



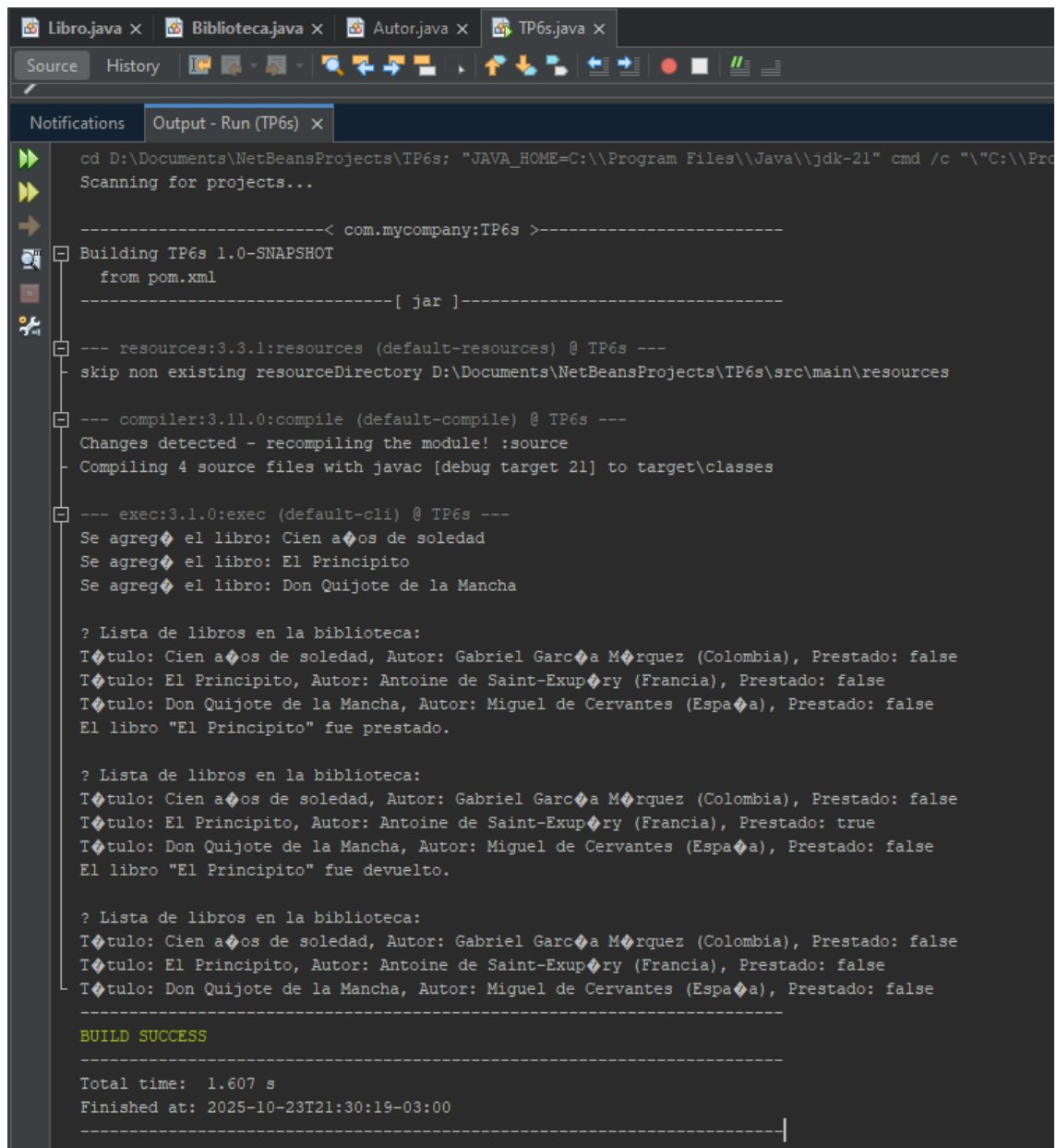
```
10  */
11  public class Biblioteca {
12      private ArrayList<Libro> libros;
13      public Biblioteca() {
14          libros = new ArrayList<>();
15      }
16      public void agregarLibro(Libro libro) {
17          libros.add(libro);
18          System.out.println("Se agregó el libro: " + libro.getTitulo());
19      }
20      public void mostrarLibros() {
21          System.out.println("\n\n Lista de libros en la biblioteca:");
22          for (Libro libro : libros) {
23              System.out.println(libro);
24          }
25      }
26      public Libro buscarLibroPorTitulo(String titulo) {
27          for (Libro libro : libros) {
28              if (libro.getTitulo().equalsIgnoreCase(titulo)) {
29                  return libro;
30              }
31          }
32          return null;
33      }
34      public void prestarLibro(String titulo) {
35          Libro libro = buscarLibroPorTitulo(titulo);
36          if (libro != null) {
37              libro.prestar();
38          } else {
39              System.out.println("No se encontró el libro \"" + titulo + "\".");
40          }
41      }
42      public void devolverLibro(String titulo) {
43          Libro libro = buscarLibroPorTitulo(titulo);
44          if (libro != null) {
45              libro.devolver();
46          } else {
47              System.out.println("No se encontró el libro \"" + titulo + "\".");
48          }
49      }
50  }
```

MAIN:



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
3   */
4
5  package com.mycompany.tp6s;
6
7  /**
8   *
9   * @author Mati
10  */
11  public class TP6s {
12
13      public static void main(String[] args) {
14          Biblioteca biblioteca = new Biblioteca();
15
16          Autor autor1 = new Autor("Gabriel García Márquez", "Colombia");
17          Autor autor2 = new Autor("Antoine de Saint-Exupéry", "Francia");
18          Autor autor3 = new Autor("Miguel de Cervantes", "España");
19          Libro libro1 = new Libro("Cien años de soledad", autor1);
20          Libro libro2 = new Libro("El Principito", autor2);
21          Libro libro3 = new Libro("Don Quijote de la Mancha", autor3);
22          biblioteca.agregarLibro(libro1);
23          biblioteca.agregarLibro(libro2);
24          biblioteca.agregarLibro(libro3);
25          biblioteca.mostrarLibros();
26          biblioteca.prestarLibro("El Principito");
27          biblioteca.mostrarLibros();
28          biblioteca.devolverLibro("El Principito");
29          biblioteca.mostrarLibros();
30      }
31  }
```

Ejecutamos y por consola nos devuelve:



```
cd D:\Documents\NetBeansProjects\TP6s; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C:\\Pro
Scanning for projects...

-----< com.mycompany:TP6s >-----
Building TP6s 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ TP6s ---
skip non existing resourceDirectory D:\Documents\NetBeansProjects\TP6s\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ TP6s ---
Changes detected - recompiling the module! :source
Compiling 4 source files with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ TP6s ---
Se agreg el libro: Cien a os de soledad
Se agreg el libro: El Principito
Se agreg el libro: Don Quijote de la Mancha

? Lista de libros en la biblioteca:
T tulo: Cien a os de soledad, Autor: Gabriel Garc a M rquez (Colombia), Prestado: false
T tulo: El Principito, Autor: Antoine de Saint-Exup ry (Francia), Prestado: false
T tulo: Don Quijote de la Mancha, Autor: Miguel de Cervantes (Espa a), Prestado: false
El libro "El Principito" fue prestado.

? Lista de libros en la biblioteca:
T tulo: Cien a os de soledad, Autor: Gabriel Garc a M rquez (Colombia), Prestado: false
T tulo: El Principito, Autor: Antoine de Saint-Exup ry (Francia), Prestado: true
T tulo: Don Quijote de la Mancha, Autor: Miguel de Cervantes (Espa a), Prestado: false
El libro "El Principito" fue devuelto.

? Lista de libros en la biblioteca:
T tulo: Cien a os de soledad, Autor: Gabriel Garc a M rquez (Colombia), Prestado: false
T tulo: El Principito, Autor: Antoine de Saint-Exup ry (Francia), Prestado: false
T tulo: Don Quijote de la Mancha, Autor: Miguel de Cervantes (Espa a), Prestado: false

BUILD SUCCESS

Total time: 1.607 s
Finished at: 2025-10-23T21:30:19-03:00
```

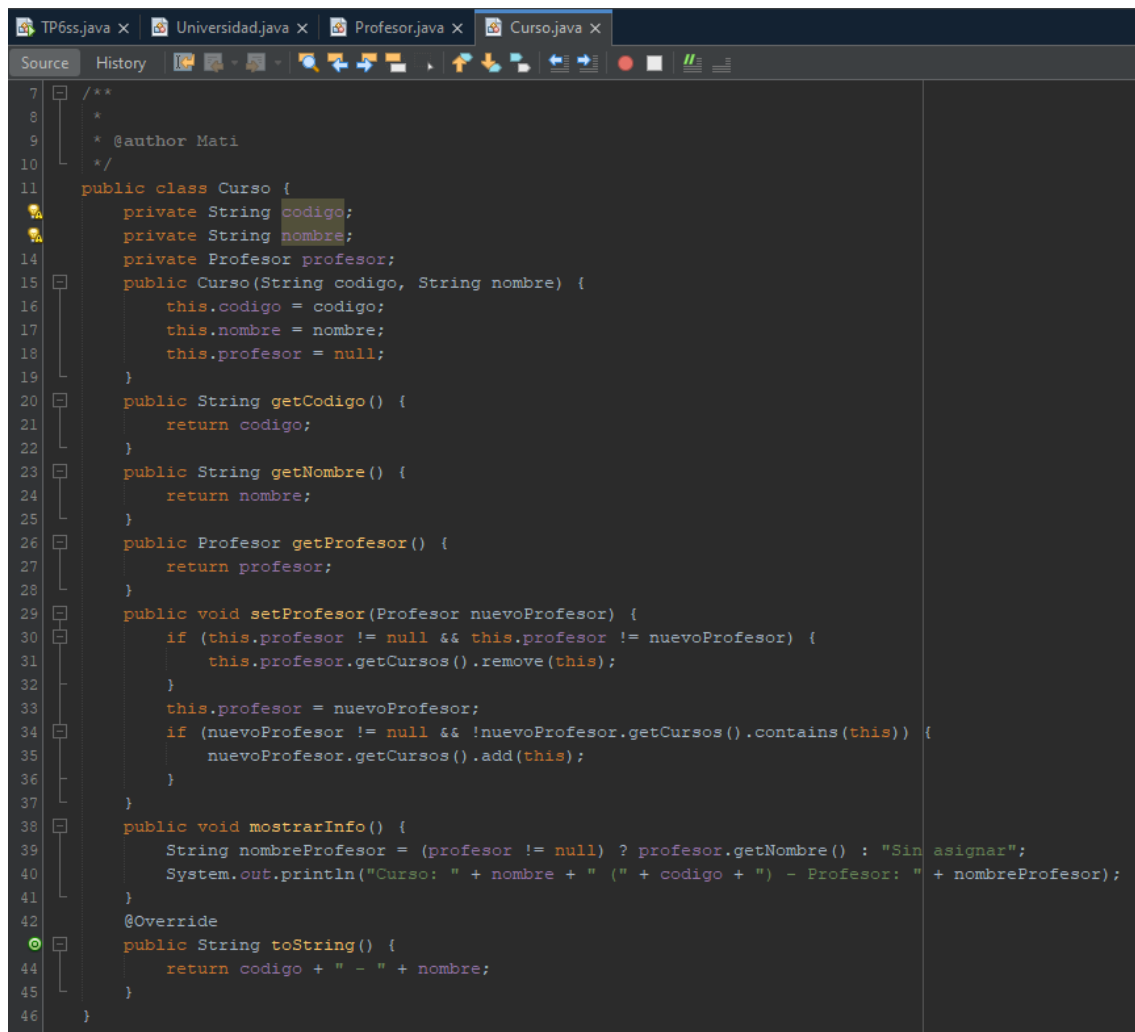
Ejercicio: Universidad, Profesor y Curso (bidireccional 1 a N)

Clase Profesor:

```
TP6ss.java x Universidad.java x Profesor.java x Curso.java x
Source History
10
11 public class Profesor {
12     private String id;
13     private String nombre;
14     private String especialidad;
15     private ArrayList<Curso> cursos;
16     public Profesor(String id, String nombre, String especialidad) {
17         this.id = id;
18         this.nombre = nombre;
19         this.especialidad = especialidad;
20         this.cursos = new ArrayList<>();
21     }
22     public String getId() {
23         return id;
24     }
25     public String getNombre() {
26         return nombre;
27     }
28     public String getEspecialidad() {
29         return especialidad;
30     }
31
32     public ArrayList<Curso> getCursos() {
33         return cursos;
34     }
35     public void agregarCurso(Curso curso) {
36         if (!cursos.contains(curso)) {
37             cursos.add(curso);
38             curso.setProfesor(this);
39         }
40     }
41     public void eliminarCurso(Curso curso) {
42         if (cursos.contains(curso)) {
43             cursos.remove(curso);
44             if (curso.getProfesor() == this) {
45                 curso.setProfesor(null);
46             }
47         }
48     }
49     public void listarCursos() {
50         System.out.println("\nCursos dictados por " + nombre + ":");
51         for (Curso c : cursos) {
52             System.out.println("- " + c.getCodigo() + ": " + c.getNombre());
53         }
54     }
}
```

```
}
public void mostrarInfo() {
    System.out.println("Profesor: " + nombre + " (" + especialidad + ") - Cursos: " + cursos.size());
}
@Override
public String toString() {
    return nombre + " - " + especialidad;
}
}
```

Clase Curso:



```
7  /**
8   *
9   * @author Mati
10  */
11  public class Curso {
12      private String codigo;
13      private String nombre;
14      private Profesor profesor;
15      public Curso(String codigo, String nombre) {
16          this.codigo = codigo;
17          this.nombre = nombre;
18          this.profesor = null;
19      }
20      public String getCodigo() {
21          return codigo;
22      }
23      public String getNombre() {
24          return nombre;
25      }
26      public Profesor getProfesor() {
27          return profesor;
28      }
29      public void setProfesor(Profesor nuevoProfesor) {
30          if (this.profesor != null && this.profesor != nuevoProfesor) {
31              this.profesor.getCursos().remove(this);
32          }
33          this.profesor = nuevoProfesor;
34          if (nuevoProfesor != null && !nuevoProfesor.getCursos().contains(this)) {
35              nuevoProfesor.getCursos().add(this);
36          }
37      }
38      public void mostrarInfo() {
39          String nombreProfesor = (profesor != null) ? profesor.getNombre() : "Sin asignar";
40          System.out.println("Curso: " + nombre + " (" + codigo + ") - Profesor: " + nombreProfesor);
41      }
42      @Override
43      public String toString() {
44          return codigo + " - " + nombre;
45      }
46  }
```

Clase Universidad:

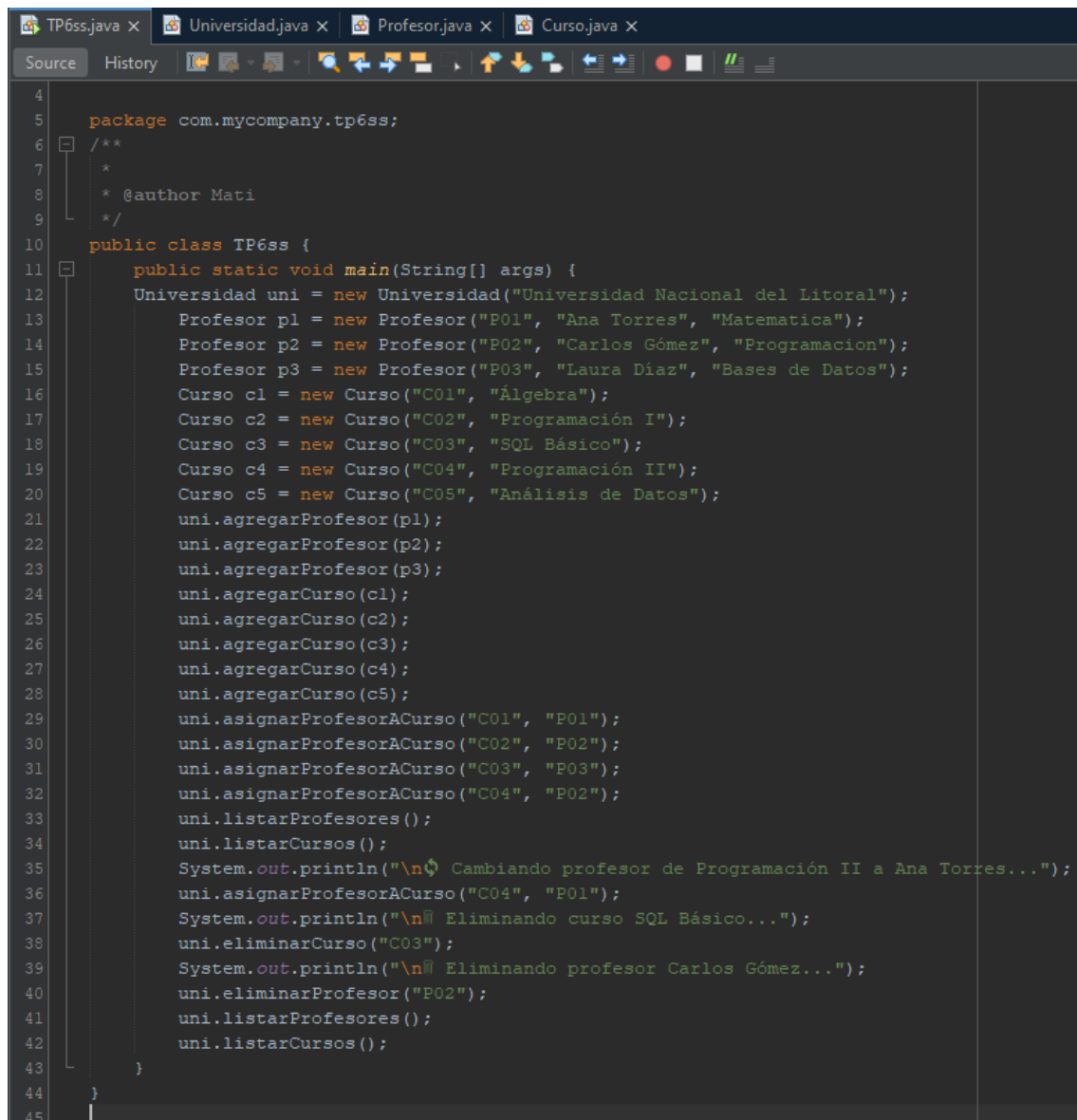
```
TP6ss.java x Universidad.java x Profesor.java x Curso.java x
Source History
10  */
11  public class Universidad {
12      private String nombre;
13      private ArrayList<Profesor> profesores;
14      private ArrayList<Curso> cursos;
15      public Universidad(String nombre) {
16          this.nombre = nombre;
17          profesores = new ArrayList<>();
18          cursos = new ArrayList<>();
19      }
20      public void agregarProfesor(Profesor p) {
21          profesores.add(p);
22      }
23      public void agregarCurso(Curso c) {
24          cursos.add(c);
25      }
26      public Profesor buscarProfesorPorId(String id) {
27          for (Profesor p : profesores) {
28              if (p.getId().equals(id)) return p;
29          }
30          return null;
31      }
32      public Curso buscarCursoPorCodigo(String codigo) {
33          for (Curso c : cursos) {
34              if (c.getCodigo().equals(codigo)) return c;
35          }
36          return null;
37      }
38      public void asignarProfesorACurso(String codigoCurso, String idProfesor) {
39          Curso curso = buscarCursoPorCodigo(codigoCurso);
40          Profesor profesor = buscarProfesorPorId(idProfesor);
41
42          if (curso != null && profesor != null) {
43              curso.setProfesor(profesor);
44              System.out.println("Profesor " + profesor.getNombre() + " asignado a " + curso.getNombre());
45          } else {
46              System.out.println("X No se pudo asignar. Verifique los IDs.");
47          }
48      }
49      public void eliminarCurso(String codigo) {
50          Curso curso = buscarCursoPorCodigo(codigo);
51          if (curso != null) {
52              if (curso.getProfesor() != null) {
53                  curso.getProfesor().eliminarCurso(curso);
54              }
55          }
56      }
57  }
```

```

55         cursos.remove(curso);
56         System.out.println("Curso eliminado: " + codigo);
57     }
58 }
59 public void eliminarProfesor(String id) {
60     Profesor prof = buscarProfesorPorId(id);
61     if (prof != null) {
62         for (Curso c : new ArrayList<>(prof.getCursos())) {
63             c.setProfesor(null);
64         }
65         profesores.remove(prof);
66         System.out.println("Profesor eliminado: " + prof.getNombre());
67     }
68 }
69 public void listarProfesores() {
70     System.out.println("\n👤 Profesores registrados:");
71     for (Profesor p : profesores) {
72         p.mostrarInfo();
73     }
74 }
75 public void listarCursos() {
76     System.out.println("\n📖 Cursos disponibles:");
77     for (Curso c : cursos) {
78         c.mostrarInfo();
79     }
80 }
81 }
82

```

MAIN:



```
4
5 package com.mycompany.tp6ss;
6 /**
7  *
8  * @author Mati
9  */
10 public class TP6ss {
11     public static void main(String[] args) {
12         Universidad uni = new Universidad("Universidad Nacional del Litoral");
13         Profesor p1 = new Profesor("P01", "Ana Torres", "Matematica");
14         Profesor p2 = new Profesor("P02", "Carlos Gómez", "Programacion");
15         Profesor p3 = new Profesor("P03", "Laura Díaz", "Bases de Datos");
16         Curso c1 = new Curso("C01", "Álgebra");
17         Curso c2 = new Curso("C02", "Programación I");
18         Curso c3 = new Curso("C03", "SQL Básico");
19         Curso c4 = new Curso("C04", "Programación II");
20         Curso c5 = new Curso("C05", "Análisis de Datos");
21         uni.agregarProfesor(p1);
22         uni.agregarProfesor(p2);
23         uni.agregarProfesor(p3);
24         uni.agregarCurso(c1);
25         uni.agregarCurso(c2);
26         uni.agregarCurso(c3);
27         uni.agregarCurso(c4);
28         uni.agregarCurso(c5);
29         uni.asignarProfesorACurso("C01", "P01");
30         uni.asignarProfesorACurso("C02", "P02");
31         uni.asignarProfesorACurso("C03", "P03");
32         uni.asignarProfesorACurso("C04", "P02");
33         uni.listarProfesores();
34         uni.listarCursos();
35         System.out.println("\n🔄 Cambiando profesor de Programación II a Ana Torres...");
36         uni.asignarProfesorACurso("C04", "P01");
37         System.out.println("\n🗑 Eliminando curso SQL Básico...");
38         uni.eliminarCurso("C03");
39         System.out.println("\n🗑 Eliminando profesor Carlos Gómez...");
40         uni.eliminarProfesor("P02");
41         uni.listarProfesores();
42         uni.listarCursos();
43     }
44 }
45
```

Ejecutamos y por consola nos devuelve:

```
TP6ss.java x Universidad.java x Profesor.java x Curso.java x
Source History
Notifications Output - Run (TP6ss) x

- skip non existing resourceDirectory D:\Documents\NetBeansProjects\TP6ss\src\main\resources
- --- compiler:3.11.0:compile (default-compile) @ TP6ss ---
  Changes detected - recompiling the module! :source
  Compiling 4 source files with javac [debug target 21] to target\classes
- --- exec:3.1.0:exec (default-cli) @ TP6ss ---
  Profesor Ana Torres asignado a lgebra
  Profesor Carlos Gmez asignado a Programaci3n I
  Profesor Laura Daz asignado a SQL B3sico
  Profesor Carlos Gmez asignado a Programaci3n II

  ??? Profesores registrados:
  Profesor: Ana Torres (Matematica) - Cursos: 1
  Profesor: Carlos Gmez (Programacion) - Cursos: 2
  Profesor: Laura Daz (Bases de Datos) - Cursos: 1

  ? Cursos disponibles:
  Curso: lgebra (C01) - Profesor: Ana Torres
  Curso: Programaci3n I (C02) - Profesor: Carlos Gmez
  Curso: SQL B3sico (C03) - Profesor: Laura Daz
  Curso: Programaci3n II (C04) - Profesor: Carlos Gmez
  Curso: An3lisis de Datos (C05) - Profesor: Sin asignar

  ? Cambiando profesor de Programaci3n II a Ana Torres...
  Profesor Ana Torres asignado a Programaci3n II

  ?? Eliminando curso SQL B3sico...
  Curso eliminado: C03

  ?? Eliminando profesor Carlos Gmez...
  Profesor eliminado: Carlos Gmez

  ??? Profesores registrados:
  Profesor: Ana Torres (Matematica) - Cursos: 2
  Profesor: Laura Daz (Bases de Datos) - Cursos: 0

  ? Cursos disponibles:
  Curso: lgebra (C01) - Profesor: Ana Torres
  Curso: Programaci3n I (C02) - Profesor: Sin asignar
  Curso: Programaci3n II (C04) - Profesor: Ana Torres
  Curso: An3lisis de Datos (C05) - Profesor: Sin asignar

  BUILD SUCCESS

  Total time: 1.631 s
  Finished at: 2025-10-23T21:50:51-03:00
```