

Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

**OBLIGATORIO ALGORITMOS Y
ESTRUCTURAS DE DATOS 1**

DOCUMENTO



Nicolas Gimenez – 291950



Cristian García – 317010

Grupo N3D

Docente: Rafael Cohen

Analista en tecnologías de la información

21/11/2024

Índice

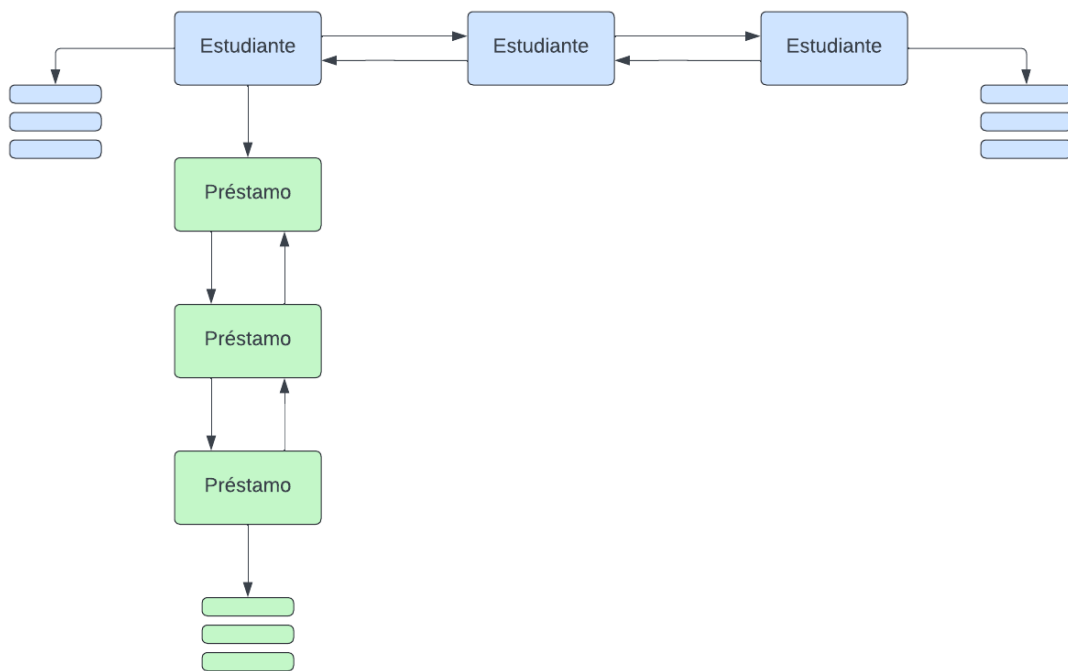
| | | |
|------|--|----|
| 1. | Representación gráfica de estructuras..... | 3 |
| 2. | Juego de prueba y evidencia | 9 |
| 2.1 | Crear sistema de gestión..... | 9 |
| 2.2 | Agregar un estudiante..... | 9 |
| 2.3 | Obtener estudiante..... | 11 |
| 2.4 | Eliminar estudiante..... | 12 |
| 2.5 | Agregar Libro..... | 13 |
| 2.6 | Prestar libro..... | 14 |
| 2.7 | Reservar libro..... | 16 |
| 2.8 | Devolver libro..... | 18 |
| 2.9 | Eliminar libro..... | 20 |
| 2.10 | Listar estudiantes..... | 21 |
| 2.11 | Listar libros..... | 22 |
| 2.12 | Listar libros por categoría..... | 22 |
| 2.13 | Listar préstamos de un estudiante..... | 23 |
| 2.14 | Libros más prestados..... | 24 |
| 2.15 | Deshacer n eliminaciones..... | 24 |
| 2.16 | Cantidad de préstamos activos..... | 25 |
| 2.17 | Ranking de categorías..... | 26 |
| 2.18 | Evidencia de test..... | 27 |

1. Representación gráfica de estructuras.

Lista doble encadenada de **estudiantes** ordenado creciente por número de estudiante.

Se define este tipo de dato para cumplir por ejemplo con el requerimiento 3.1: Se listan los datos de los estudiantes ordenados creciente por número, cargando el resultado en el valor String del retorno.

Cada estudiante contiene una lista de sus préstamos.



ESTUDIANTE

Nombre: string

Apellido: string

Número: int

PrestamosActivos: ListaDoble<Préstamo>

PRESTAMO

Libro: Libro

Estudiante: Estudiante

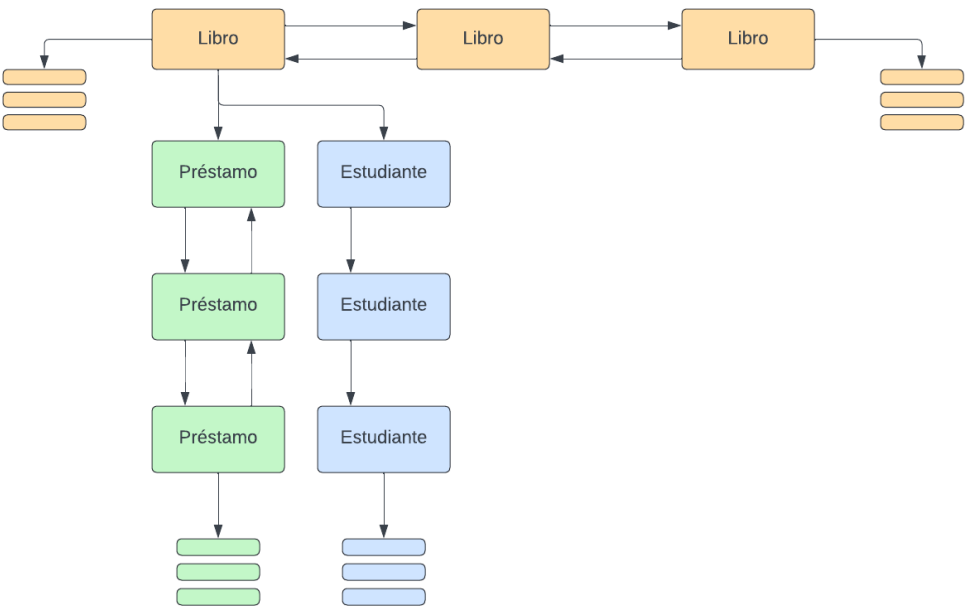
Fecha: LocalDateTime

Activo: boolean

Lista doble encadenada de **libros** ordenada creciente por ISBN.

Se define este tipo de dato para cumplir por ejemplo con el requerimiento 3.2: Se listan los datos de todos los libros de la biblioteca ordenados creciente por ISBN cargando el resultado en el valor String del retorno.

El libro cuenta con una lista de préstamos y una cola de estudiantes que han hecho reservas de ese libro.



LIBRO

Nombre: string

ISBN: string

Categoria: string

Total: int

ESTUDIANTE

Nombre: string

Apellido: string

Número: int

PrestamosActivos: ListaDoble<Prestamo>

PRESTAMO

Libro: Libro

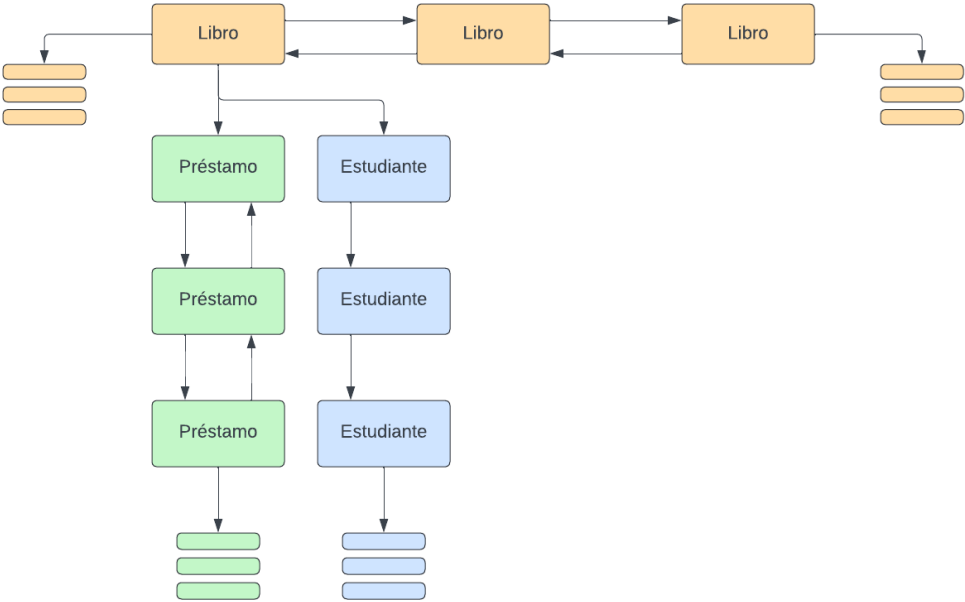
Estudiante: Estudiante

Fecha: LocalDateTime

Activo: boolean

Lista doble encadenada de **libros** ordenada decreciente por cantidad de préstamos.

Lista de libros indexada por cantidad de préstamos del libro.



| LIBRO |
|-------------------|
| Nombre: string |
| ISBN: string |
| Categoria: string |
| Total: int |

| ESTUDIANTE |
|--|
| Nombre: string |
| Apellido: string |
| Número: int |
| PrestamosActivos: ListaDoble<Prestamo> |

| PRESTAMO |
|------------------------|
| Libro: Libro |
| Estudiante: Estudiante |
| Fecha: LocalDateTime |
| Activo: boolean |

Lista doble encadenada de **préstamos** ordenada creciente por ISBN.

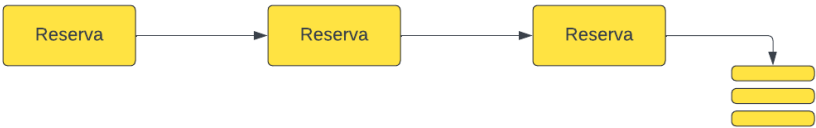
Se define este tipo de dato para cumplir por ejemplo con el requerimiento 3.2: Se deben listar el/los libros más prestados ordenado por ISBN, cargando el resultado en el valor String del retorno. En caso de existir libros con la misma cantidad “máxima de préstamos”, se deberán mostrar todos.



| |
|------------------------|
| PRESTAMO |
| Libro: Libro |
| Estudiante: Estudiante |
| Fecha: LocalDateTime |
| Activo: boolean |

Cola de **reservas**.

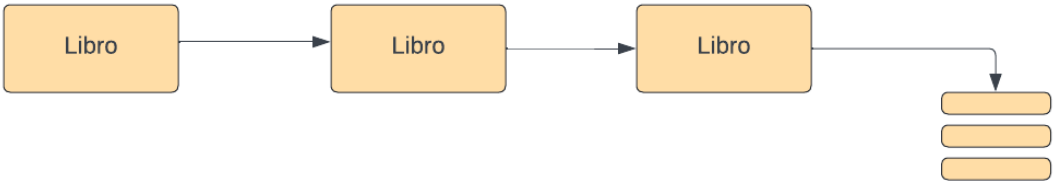
Se define este tipo de dato para cumplir por ejemplo con el requerimiento 2.7: Registrar una reserva de un ejemplar de un libro para un estudiante. Solo se puede realizar una reserva si no existe stock disponible en el momento para realizar el préstamo.



| |
|------------------------|
| RESERVA |
| Libro: Libro |
| Estudiante: Estudiante |
| Fecha: LocalDateTime |

Pila de **libros**.

Se define este tipo de dato para cumplir por ejemplo con el requerimiento 3.6: Se deben deshacer las ultimas n eliminaciones de libros realizadas en el sistema. Si existen menos de n eliminaciones, se deshacen todas las realizadas. Se deberá cargar el resultado de los libros recuperados en el valor String del retorno, según el orden de recuperación (primer mostrar el último cancelado y recuperado)



| |
|-------------------|
| LIBRO |
| Nombre: string |
| ISBN: string |
| Categoria: string |
| Total: int |

2. Juego de prueba y evidencia

Se realizó un juego de prueba dentro del proyecto donde se ejecutan los métodos solicitados y se muestran los resultados obtenidos.

A continuación se detalla el código con la prueba y print del resultado obtenido.

2.1 Crear sistema de gestión.

```
p.ver(s.crearSistemaDeGestion().resultado, Retorno.Resultado.OK, "se crea sistema");
```

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se crea sistema  
.....OK.....  
-----
```

2.2 Agregar un estudiante.

Agregar estudiante caso de éxito.

```
p.ver(s.agregarEstudiante("Luis", "Suarez", 1).resultado, Retorno.Resultado.OK, "OK:  
Se agrego a Luis Suarez");
```

Resultado:

```
----- Testeo -----  
  
Comentario: OK: Se agrego a Luis Suarez  
.....OK.....  
-----
```

Agregar estudiante caso error 1.

```
p.ver(s.agregarEstudiante("", "Polenta", 11).resultado, Retorno.Resultado.ERROR_1, "ERROR 1 : Faltan parametros");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: Faltan parametros  
.....OK.....  
-----
```

Agregar estudiante caso error 2.

```
p.ver(s.agregarEstudiante("Diego", "Aguirre", -1).resultado, Retorno.Resultado.ERROR_2, "ERROR 2 : Numero fuera de rango");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: Numero fuera de rango  
.....OK.....  
-----
```

Agregar estudiante caso error 3.

```
p.ver(s.agregarEstudiante("Luis", "Suarez", 1).resultado, Retorno.Resultado.ERROR_3, "Error 3: Se intento agregar un estudiante existente");
```

Resultado:

```
----- Testeo -----  
  
Comentario: Error 3: Se intento agregar un estudiante existente  
.....OK.....  
-----
```

2.3 Obtener estudiante.

Obtener estudiante caso de éxito.

p.ver(s.obtenerEstudiante(1).resultado, Retorno.Resultado.OK, "OK: se obtuvo estudiante");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se obtuvo estudiante  
.....OK.....  
  
-----
```

Obtener estudiante error 1.

p.ver(s.obtenerEstudiante(-1).resultado, Retorno.Resultado.ERROR_1, "Error 1: numero fuera de rango");

Resultado:

```
----- Testeo -----  
  
Comentario: Error 1: numero fuera de rango  
.....OK.....  
  
-----
```

Obtener estudiante error 2.

p.ver(s.obtenerEstudiante(20).resultado, Retorno.Resultado.ERROR_2, "Error 2 - el estudiante 20 no existe");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: el estudiante 20 no existe  
.....OK.....  
  
-----
```

2.4 Eliminar estudiante.

Eliminar estudiante caso de éxito.

p.ver(s.eliminarEstudiante(10).resultado, Retorno.Resultado.OK, "OK: Se elimina estudiante 10");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: Se elimina estudiante 10  
.....OK.....
```

Eliminar estudiante error 1.

p.ver(s.eliminarEstudiante(-10).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: Se intenta eliminar estudiante con nro fuera de rango");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: Se intenta eliminar estudiante con nro fuera de rango  
.....OK.....
```

Eliminar estudiante error 2.

p.ver(s.eliminarEstudiante(20).resultado, Retorno.Resultado.ERROR_2, "ERROR 2: no existe ese estudiante");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: no existe ese estudiante  
.....OK.....
```

Eliminar estudiante error 3.

p.ver(s.eliminarEstudiante(1).resultado, Retorno.Resultado.ERROR_3, "ERROR 3: Se intenta eliminar estudiante con prestamos");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 3: Se intenta eliminar estudiante con prestamos  
.....OK.....
```

2.5 Agregar Libro.

Agregar libro caso de éxito.

p.ver(s.agregarLibro("Libro 15", "15", "categoria 1", 1).resultado, Retorno.Resultado.OK, "OK: se agrega libro 15 con 1 ejemplar");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se agrega libro 15 con 1 ejemplar  
.....OK.....
```

Agregar libro error 1.

p.ver(s.agregarLibro("", "11", "categoria 3", 1).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: se intenta agregar libro con falta de parametros");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: se intenta agregar libro con falta de parametros  
.....OK.....
```

Agregar libro error 2.

```
p.ver(s.agregarLibro("Libro 10", "10", "categoria 3", 1).resultado,
Retorno.Resultado.ERROR_2, "ERROR 2: Se intenta agregar libro existente");
```

Resultado:

```
----- Testeo -----
Comentario: ERROR 2: Se intenta agregar libro existente
.....OK.....
-----
```

Agregar libro error 3.

```
p.ver(s.agregarLibro("Libro 13", "20", "ategoria 3", -1).resultado,
Retorno.Resultado.ERROR_3, "OK: se intenta agregar libro con cantidad de ejemplares
incorrecta");
```

Resultado:

```
----- Testeo -----
Comentario: ERROR 3: se intenta agregar libro con cantidad de ejemplares incorrecta
.....OK.....
-----
```

2.6 Prestar libro.

Prestar libro caso de éxito.

```
p.ver(s.prestarLibro("3", 1).resultado, Retorno.Resultado.OK, "OK: se presto Libro 1
ejemplar 1 libro 3");
```

Resultado:

```
----- Testeo -----
Comentario: OK: se presto Libro 1 ejemplar 1 libro 3
.....OK.....
-----
```

Prestar libro error 1.

p.ver(s.prestarLibro("", 1).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: ISBN vacio");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: ISBN vacio  
.....OK.....  
-----
```

Prestar libro error 2.

p.ver(s.prestarLibro("25", 1).resultado, Retorno.Resultado.ERROR_2, "ERROR 2: no existe libro con ese ISBN");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: no existe libro con ese ISBN  
.....OK.....  
-----
```

Prestar libro error 3.

p.ver(s.prestarLibro("3", -1).resultado, Retorno.Resultado.ERROR_3, "ERROR 3: numero de estudiante fuera de rango");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 3: numero de estudiante fuera de rango  
.....OK.....  
-----
```

Prestar libro error 4.

p.ver(s.prestarLibro("3", 31).resultado, Retorno.Resultado.ERROR_4, "ERROR 4: no Existe ese estudiante");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 4: no Existe ese estudiante  
.....OK.....  
-----
```

Prestar libro error 5.

p.ver(s.prestarLibro("10", 6).resultado, Retorno.Resultado.ERROR_5, "ERROR 5: Libro sin stock");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 5: Libro sin stock  
.....OK.....
```

Prestar libro error 6.

p.ver(s.prestarLibro("3", 1).resultado, Retorno.Resultado.ERROR_6, "ERROR 6: ya existe ese prestamo para ese estudiante libro");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 6: ya existe ese prestamo para ese estudiante libro  
.....OK.....
```

2.7 Reservar libro.

Reservar libro caso de éxito.

p.ver(s.reservarLibro("15", 2).resultado, Retorno.Resultado.OK, "OK: Se reserva libro 15");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: Se reserva libro 15  
.....OK.....
```


Reservar libro error 1.

```
p.ver(s.reservarLibro("", 1).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: ISBN null");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: ISBN null  
.....OK.....  
-----
```

Reservar libro error 2.

```
p.ver(s.reservarLibro("34", 1).resultado, Retorno.Resultado.ERROR_2, "ERROR 2: no existe ese libro");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: no existe ese libro  
.....OK.....  
-----
```

Reservar libro error 3.

```
p.ver(s.reservarLibro("3", -1).resultado, Retorno.Resultado.ERROR_3, "ERROR 3: numero fuera de rango");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 3: numero fuera de rango  
.....OK.....  
-----
```

Reservar libro error 4.

p.ver(s.reservarLibro("3", 22).resultado, Retorno.Resultado.ERROR_4, "ERROR 4: no existe ese estudiante");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 4: no existe ese estudiante  
.....OK.....
```

Reservar libro error 5.

p.ver(s.reservarLibro("2", 5).resultado, Retorno.Resultado.ERROR_5, "ERROR 5: ese libro aún tiene stock");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 5: ese libro aún tiene stock  
.....OK.....
```

2.8 Devolver libro.

Devolver libro caso de éxito.

p.ver(s.devolverLibro("1", 1).resultado, Retorno.Resultado.OK, "OK: se devuelve el libro 1, estudiante 1");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se devuelve el libro 1, estudiante 1  
.....OK.....
```

Devolver libro error 1.

p.ver(s.devolverLibro("", 1).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: ISBN vacío");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: ISBN vacío  
.....OK.....
```

Devolver libro error 2.

p.ver(s.devolverLibro("100", 1).resultado, Retorno.Resultado.ERROR_2, "ERROR 2: no existe libro con ese ISBN");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: no existe libro con ese ISBN  
.....OK.....
```

Devolver libro error 3.

p.ver(s.devolverLibro("1", -1).resultado, Retorno.Resultado.ERROR_3, "ERROR 3: número fuera de rango");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 3: número fuera de rango  
.....OK.....
```

Devolver libro error 4.

p.ver(s.devolverLibro("1", 150).resultado, Retorno.Resultado.ERROR_4, "ERROR 4: no existe estudiante con ese número");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 4: no existe estudiante con ese número  
.....OK.....  
  
-----
```

Devolver libro error 5.

p.ver(s.devolverLibro("7", 1).resultado, Retorno.Resultado.ERROR_5, "ERROR 5: no existe un prestamo activo de ese libro para ese estudiante");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 5: no existe un prestamo activo de ese libro para ese estudiante  
.....OK.....  
  
-----
```

2.9 Eliminar libro.

Eliminar libro caso de éxito.

p.ver(s.eliminarLibro("ISBN2").resultado, Retorno.Resultado.OK, "OK: se elimina el libro ISBN2");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se elimina el libro ISBN2  
.....OK.....  
  
-----
```

Eliminar libro error 1.

```
p.ver(s.eliminarLibro("").resultado, Retorno.Resultado.ERROR_1, "ERROR 1: el ISBN es vacío a null");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: el ISBN es vacío a null  
.....OK.....
```

Eliminar libro error 2.

```
p.ver(s.eliminarLibro("1").resultado, Retorno.Resultado.ERROR_2, "ERROR 2: el libro tiene préstamos");
```

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: el libro tiene préstamos  
.....OK.....
```

2.10 Listar estudiantes.

Listar estudiantes caso de éxito.

```
p.ver(s.listarEstudiantes().resultado, Retorno.Resultado.OK, "OK: se listan los estudiantes");
```

```
System.out.println(s.listarEstudiantes().valorString + "\n");
```

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se listan los estudiantes  
.....OK.....  
  
-----  
Luis#Suarez#1|Edi#Cavani#2|Andres#Lima#3|Darwin#Nuñez#4|Hugo#Olivera#5  
|Carlos#Soca#6|Diego#Maradona#7|Leo#Messi#8|Roberto#Carlos#9
```

2.11 Listar libros.

Listar libros caso de éxito.

```
p.ver(s.listarLibros().resultado, Retorno.Resultado.OK, "OK: se listan los libros");
```

```
System.out.println(s.listarLibros().valorString + "\n");
```

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se listan los libros  
.....OK.....  
  
-----  
  
[Libro 0#0#categoria 0|Libro 1#1#categoria 1|Libro 10#10#categoria 3|Li  
bro 15#15#categoria 1|Libro 2#2#categoria 1|Libro 3#3#categoria 1|Libr  
o 4#4#categoria 2|Libro 5#5#categoria 2|Libro 6#6#categoria 3|Libro 7#  
7#categoria 3|Libro 8#8#categoria 3|Libro 9#9#categoria 3]
```

2.12 Listar libros por categoría.

Listar libros por categoría caso de éxito.

```
p.ver(s.listarLibros("categoria 1").resultado, Retorno.Resultado.OK, "OK: se listan los  
libros de la categoria 1");
```

```
System.out.println(s.listarLibros("categoria 1").valorString + "\n");
```

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se listan los libros de la categoria 1  
.....OK.....  
  
-----  
  
[Libro 1#1#categoria 1|Libro 15#15#categoria 1|Libro 2#2#categoria 1|Li  
bro 3#3#categoria 1]
```

Listar libros por categoría error 1.

p.ver(s.listarLibros("").resultado, Retorno.Resultado.ERROR_1, "ERROR 1: la categoría es vacía o null");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: la categoría es vacía o null  
.....OK.....  
-----
```

2.13 Listar préstamos de un estudiante.

Listar préstamos de un estudiante caso de éxito.

p.ver(s.listarPrestamos(1).resultado, Retorno.Resultado.OK, "OK: se listan los prestamos del estudiante 1");

System.out.println(s.listarPrestamos(1).valorString + "\n");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se listan los prestamos del estudiante 1  
.....OK.....  
-----  
[ Libro 8#8#categoría 3#true|Libro 6#6#categoría 3#true|Libro 5#5#catego  
ria 2#true|Libro 4#4#categoría 2#true|Libro 2#2#categoría 1#true|Libro  
15#15#categoría 1#true|Libro 3#3#categoría 1#true|Libro 1#1#categoría  
1#false ]
```

Listar préstamos de un estudiante error 1.

p.ver(s.listarPrestamos(0).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: el número de estudiante está fuera de rango");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: el número de estudiante está fuera de rango  
.....OK.....  
-----
```

Listar préstamos de un estudiante error 2.

p.ver(s.listarPrestamos(2505).resultado, Retorno.Resultado.ERROR_2, "ERROR 2: no existe un estudiante con ese número");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 2: no existe un estudiante con ese número  
.....OK.....  
  
-----
```

2.14 Libros más prestados.

Libros más prestados caso de éxito.

p.ver(s.librosMasPrestados().resultado, Retorno.Resultado.OK, "OK: se listan el/los libro/s más prestado/s");

System.out.println(s.librosMasPrestados().valorString + "\n");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se listan el/los libro/s más prestado/s  
.....OK.....  
  
-----  
  
Libro 4#4#categoria 2#2
```

2.15 Deshacer n eliminaciones.

Deshacer n eliminaciones de libros caso de éxito.

p.ver(s.deshacerEliminaciones(2).resultado, Retorno.Resultado.OK, "OK: se deshacen las ultimas 2 eliminaciones de libros");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se deshacen las ultimas 2 eliminaciones de libros  
.....OK.....  
  
-----
```


Deshacer n eliminaciones de libros error 1.

p.ver(s.deshacerEliminaciones(0).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: n es menor o igual a 0");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: n es menor o igual a 0  
.....OK.....  
-----
```

2.16 Cantidad de préstamos activos.

Cantidad de préstamos activos caso de éxito.

p.ver(s.cantidadPrestamosActivos("1").resultado, Retorno.Resultado.OK, "OK: se muestra la cantidad de préstamos del libro 1");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se muestra la cantidad de préstamos del libro 1  
.....OK.....  
-----
```

Cantidad de préstamos activos error 1.

p.ver(s.cantidadPrestamosActivos("")).resultado, Retorno.Resultado.ERROR_1, "ERROR 1: el ISBN es vacío o null");

Resultado:

```
----- Testeo -----  
  
Comentario: ERROR 1: el ISBN es vacío o null  
.....OK.....  
-----
```

2.17 Ranking de categorías.

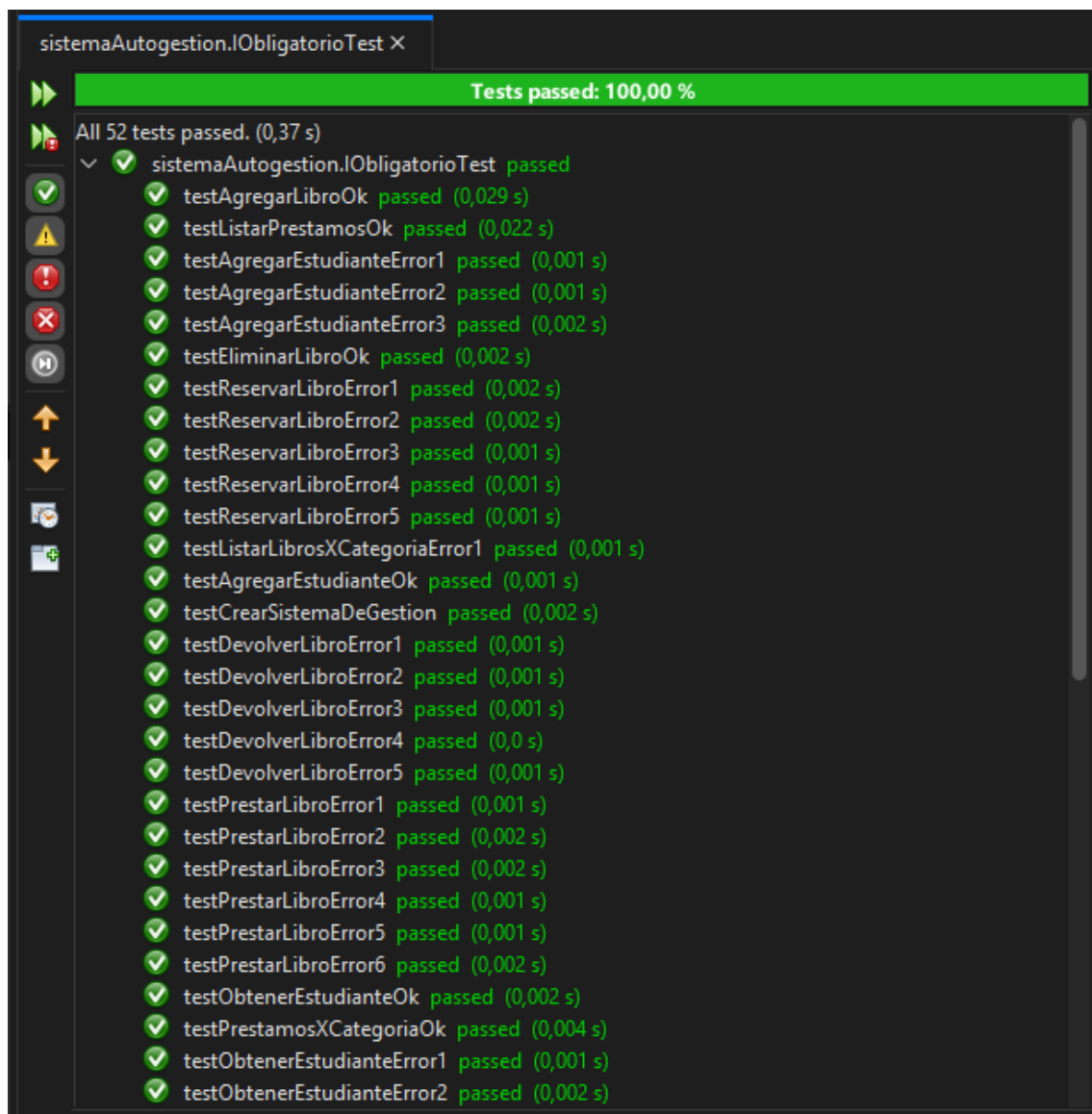
Ranking de categorías caso de éxito.

p.ver(s.prestamosXCategoría().resultado, Retorno.Resultado.OK, "OK: se muestra el ranking de prestamos por categoria");

Resultado:

```
----- Testeo -----  
  
Comentario: OK: se muestra el ranking de prestamos por categoria  
.....OK.....  
-----
```

2.18 Evidencia de test.



```
✓ testObtenerEstudianteError2 passed (0,002 s)
✓ testListarLibros passed (0,001 s)
✓ testLibrosMasPrestadosOk passed (0,002 s)
✓ testCantidadPrestamosActivosOk passed (0,0 s)
✓ testAgregarLibroError1 passed (0,001 s)
✓ testAgregarLibroError2 passed (0,0 s)
✓ testAgregarLibroError3 passed (0,002 s)
✓ testReservarLibroOk passed (0,001 s)
✓ testDevolverLibroOk passed (0,001 s)
✓ testEliminarEstudianteOk passed (0,0 s)
✓ testDeshacerEliminacionesError1 passed (0,0 s)
✓ testLibrosMasPrestadosOk2 passed (0,002 s)
✓ testPrestarLibroOk passed (0,0 s)
✓ testEliminarEstudianteError1 passed (0,001 s)
✓ testEliminarEstudianteError2 passed (0,001 s)
✓ testEliminarEstudianteError3 passed (0,001 s)
✓ testEliminarLibroError1 passed (0,0 s)
✓ testEliminarLibroError2 passed (0,001 s)
✓ testListarEstudiantes passed (0,002 s)
✓ testListarLibrosXCategoriaOk passed (0,0 s)
✓ testDeshacerEliminacionesOk passed (0,001 s)
✓ testListarPrestamosError1 passed (0,0 s)
✓ testListarPrestamosError2 passed (0,002 s)
✓ testCantidadPrestamosActivosError1 passed (0,001 s)
```