**Proyecto: BiblioMóvil**

**Plan de pruebas**

**Paquete: Dominio**

* **Clase: BiblioMovil**

-Método descripciónLibro(String titulo)

**Particiones equivalencias:** [Libro encontrado],[Libro no encontrado]

**Valores límites:** No existen.

**Valores a probar:** Cien años de soledad, libro Inventado.

**Nº de pruebas:** 2.

-Método registrarSocio(Socio nuevoSocio), getSocios();

Se hizo en conjunto para saber si se añadían los socios al array.

**Particiones equivalencias:** [Socio añadido],[Socio no añadido]

**Valores límites** No existen.

**Valores posibles:** Socio nuevoSocio=new Socio("Juan","González","14-07-1996","juang@gmail.com","123456"), no añadir socio.

**Nº de pruebas:** 2. (Se prueban en un solo método, según lo visto en la tutoría)

-Método Comprobar\_email(String email)

**Particiones equivalencias:** [Email coincide],[Email no coincide]

**Valores límites:** No existen.

**Valores posibles:** [villaltamaria@gmail.com](mailto:villaltamaria@gmail.com), noexiste@gmail.com

**Nº de pruebas:** 2

- Método Comprobar\_contrasenya(String contrasenya)

**Particiones equivalencias:** [Contraseña coincide],[Contraseña no coincide]

**Valores límites:** No existen.

**Valores posibles:**123456, incorrecto

**Nº de pruebas:** 2

-Método tituloLibro(String nombre)

**Particiones equivalencias:** [Libro existe en biblioteca],[Libro no existe en biblioteca]

**Valores límites:** No existen.

**Valores posibles:** Cien años de soledad, Este Libro No Existe.

**Nº de pruebas:** 2

-Método getDisponibilidad(String titulo)

**Particiones equivalencias:** [Disponible],[No disponible]

**Valores límites:** No existen.

**Valores posibles:** Cien años de soledad, La chica de nieve.

**Nº de pruebas:** 2

-Método Libro Libro(String titulo)

**Particiones equivalencias:** [Objeto Libro Existe],[Objeto Libro No Existe]

**Valores límites:** No existen.

**Valores posibles:** Cien años de soledad, Matemática 4

**Nº de pruebas:** 2

-Método getLibros()

**Particiones equivalencias:** [Lista libros]

**Valores límites:** No existen.

**Valores posibles:** ArrayList<Libro> esperado=listaLibros;

**Nº de pruebas:** 1.

-Método setSocios() y .getSocios()

**Particiones equivalencias:** [Lista socios coincide] [Lista socios no coincide]

**Valores límites:** No existen.

**Valores posibles:** ArrayList<Libro> esperado=listaLibros, ArrayList<Libro> esperado=listaLibros con el Objeto BiblioMovil miBiblio, ArrayList<Libro> esperado=listaLibros, ArrayList<Libro> esperado=listaLibros con el Objeto BiblioMovil miBiblio1.

**Nº de pruebas:** 2

**Nº total de pruebas de la clase BiblioMovil:17**

* **Clase: Libro**

-Método getTitulo

**Particiones equivalencia:** [“Devuelve un título”]

**Valores limites:** No existen

**Valores posibles:** Lugares asombrosos.

**Nº de pruebas:** 1

-Método getAnyo\_publicacion

**Particiones equivalencia:** [“Devuelve el año de publicación”]

**Valores limites:** No existen

**Valores posibles**: 2019.

**Nº de pruebas:** 1

-Método getAutor

**Particiones equivalencia:** [“Devuelve el nombre del autor”]

**Valores limites:** No existen

**Valores posibles**: Luis Villar.

**Nº de pruebas:** 1

-Método getEditorial

**Particiones equivalencia:** [“Devuelve la editorial”]

**Valores limites:** No existen

**Valores posibles:** ALFAGUARA JUVENIL.

**Nº de pruebas:** 1

-Método getDisponibilidad

**Particiones equivalencia:** [Disponible] [No disponible]

**Valores limites:** No existen.

**Valores posibles:** Libro mi\_libro1 = new Libro("Lugares asombrosos", 2019, "Luis Villar", "ALFAGUARA JUVENIL", true, Formato.FISICO, "La curiosidad es el motor de las aventuras que lo lanzan hasta los terrenos más inhospitos y misteriosos.") **àDisponible** , Libro mi\_libro2 = new Libro("Don Quijote de la Mancha", 1615, "Miguel de Cervantes", "Francisco de Robles", false, Formato.DIGITAL, "Es la primera obra genuinamente desmitificadora de la tradición caballeresca y cortés por su tratamiento burlesco.") **àNo disponible**

**Nº de pruebas:** 2.

-Método getFormato

**Particiones equivalencia:** [Formato Físico], [Formato Digital]

**Valores limites:** No existen

**Valores posibles**: Libro mi\_libro1 = new Libro("Lugares asombrosos", 2019, "Luis Villar", "ALFAGUARA JUVENIL", true, Formato.FISICO, "La curiosidad es el motor de las aventuras que lo lanzan hasta los terrenos más inhospitos y misteriosos.") **àFormato Físico** , Libro mi\_libro2 = new Libro("Don Quijote de la Mancha", 1615, "Miguel de Cervantes", "Francisco de Robles", false, Formato.DIGITAL, "Es la primera obra genuinamente desmitificadora de la tradición caballeresca y cortés por su tratamiento burlesco.") **àFormato Digital**

**Nº de pruebas:** 2.

-Método getDescripcion

**Particiones equivalencia:** [“Devuelve la descripción del libro”]

**Valores limites:** No existen

**Valores posibles**: Libro mi\_libro1 = new Libro("Lugares asombrosos", 2019, "Luis Villar", "ALFAGUARA JUVENIL", true, Formato.FISICO, "La curiosidad es el motor de las aventuras que lo lanzan hasta los terrenos más inhospitos y misteriosos.") **àDescripción** “La curiosidad es el motor de las aventuras que lo lanzan hasta los terrenos más inhospitos y misteriosos”.

**Nº de pruebas:** 1.

**Nº total de pruebas de la clase Libro:9**

* **Clase: Prestamo**

-Método anyadirPrestamo(Prestamo prestamo), getLibrosPrestados():

Se hizo en conjunto para saber si se añadían los prestamos al array.

**Particiones equivalencias:** [Prestamo añadido],[Prestamo no añadido]

**Valores límites:** No existen.

**Valores posibles:** Prestamo nuevoPrestamo=new Prestamo(Prestamo.fecha\_inicio(2020, Calendar.JANUARY, 4),Prestamo.fecha\_fin(2020,Calendar.JANUARY,4),"chaconkevim@gmail.com","Mientras te olvido"), no añadir prestamo.

**Nº de pruebas:** 2

-Método tituloCoincide(String email,String titulo)

EMAIL

**Particiones equivalencias:** [Email coincide],[Email no coincide]

**Valores límites:** No existen.

**Valores posibles:** villaltamaria@gmail.com, [estemailnoexiste@gmail.com](mailto:estemailnoexiste@gmail.com).

TITULO

**Particiones equivalencias:** [Titulo coincide], [Titulo no coincide]

**Valores límites:** No existen.

**Valores posibles:** Cien años de soledad, Este libro no está en préstamo.

**Nº de pruebas:** 4

-Método obtenerFechaPrestamo(String emailSocio,String titulo)

EMAIL

**Particiones equivalencias:** [Email existente], [Email no existente],

**Valores límites:** No existen.

**Valores posibles:**villaltamaria@gmail.com, estemailnoexiste@gmail.com.

TITULO

**Particiones equivalencias:** [Titulo existente], [Titulo no existente],

**Valores límites:** No existen.

**Valores posibles:**Cien años de soledad,La chica de nieve

**Nº de pruebas:** 4

-Método devolverLibro(String emailSocio,String titulo)

EMAIL

**Particiones equivalencias:** [Email existente], [Email no existente],

**Valores límites:** No existen.

**Valores posibles:** jonh612@outlook.com, estemailnoexiste@gmail.com.

TITULO

**Particiones equivalencias:** [Titulo existente], [Titulo no existente],

**Valores límites:** No existen.

**Valores posibles:** El Alquimista, La chica de nieve.

**Nº de pruebas:** 4.

-Método getEmailSocio, setEmailSocio(String emailSocio)

**Particiones equivalencias:** [Obtener email]

**Valores límites:** No existen.

**Valores posibles:** villaltamaria@gmail.com

**Nº de pruebas:** 1.

-Método getTituloLibroPrestamo() y setTituloLibroPrestamo(librosEnPrestamo)

**Particiones equivalencias:** [Obtener Lista Libros en Prestamo],

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, pero se prueba comparando con un arraylistArrayList<Prestamo> esperado=librosEnPrestamo, y un objeto BiblioMovil.

**Nº de pruebas:** 1.

-Método getFecha\_inicio1()

Usamos el objeto *Prestamo prestamo=new Prestamo(Prestamo.fecha\_inicio(2020, Calendar.APRIL, 1),Prestamo.fecha\_fin(2020,Calendar.APRIL,1),"villaltamaria@gmail.com","Lugares asombrosos")*

**Particiones equivalencias:** [Obtener String con fecha inicio],

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, por lo que invocamos el método usando el objeto *préstamo* de arriba.

**Nº de pruebas:** 1.

-Método getFecha\_fin1()

Usamos el objeto *Prestamo prestamo=new Prestamo(Prestamo.fecha\_inicio(2020, Calendar.APRIL, 1),Prestamo.fecha\_fin(2020,Calendar.APRIL,1),"villaltamaria@gmail.com","Lugares asombrosos")*

**Particiones equivalencias:** [Obtener String con fecha fin],

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, por lo que invocamos el método usando el objeto *préstamo* de arriba.

**Nº de pruebas:** 1.

-Método getFecha\_inicio()

Usamos el objeto *Prestamo prestamo=new Prestamo(Prestamo.fecha\_inicio(2020, Calendar.APRIL, 1),Prestamo.fecha\_fin(2020,Calendar.APRIL,1),"villaltamaria@gmail.com","Lugares asombrosos")*

**Particiones equivalencias:** [Obtener Objeto Calendar correspondiente al Objecto Prestamo con su fecha de inicio],

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, por lo que invocamos el método usando el objeto préstamo de arriba.

**Nº de pruebas:** 1.

-Método getFecha\_fin ()

Usamos el objeto *Prestamo prestamo=new Prestamo(Prestamo.fecha\_inicio(2020, Calendar.APRIL, 1),Prestamo.fecha\_fin(2020,Calendar.APRIL,1),"villaltamaria@gmail.com","Lugares asombrosos")*

**Particiones equivalencias:** [Obtener Objeto Calendar correspondiente al Objecto Prestamo con su fecha de fin],

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, por lo que invocamos el método usando el objeto *préstamo* de arriba.

**Nº de pruebas:** 1.

-Método fecha\_inicio(int anyo,int mes,int dia)

DIA

**Particiones equivalencias:** [1,28],[1,29],[1,30],[1,31]

**Valores límites:**1, 28,29,30,31

**Valores posibles:** 1,28,29,30,31

MES

**Particiones equivalencias:** [1,12]

**Valores límites:**1, 12

**Valores posibles:** 1,2,4,12.

AÑO

**Particiones equivalencias:** Infinito

**Valores límites:**No tiene limites.

**Valores posibles:** 2020(Año bisiesto), 2021(Año no bisiesto)

**Casos totales a probar:**

Dia:5

Mes:4

Años 2

**Nº casos:**40 casos.

-Método fecha\_fin(int anyo,int mes, int dia)

DIA

**Particiones equivalencias:** [1,28],[1,29],[1,30],[1,31]

**Valores límites:**1, 28,29,30,31

**Valores posibles:** 1,28,29,30,31

MES

**Particiones equivalencias:** [1,12]

**Valores límites:**, 12

**Valores posibles:** 1,2,4,12.

AÑO

**Particiones equivalencias:** Infinito

**Valores límites:**1, 12

**Valores posibles:** 2020(Año bisiesto), 2021(Año no bisiesto)

**Casos totales a probar:**

Dia:5

Mes:4

Años 2

**Nº casos:**40 casos.

-Método numeroPrestamos(String email)

EMAIL

**Particiones equivalencias:** [Email existente en prestamos],[Email no existente en prestamos]

**Valores límites:** No existen.

**Valores posibles:** villaltamaria@gmail.com, anto123@gmail.com.

**Nº de pruebas: 2**.

**Nº total de pruebas de la clase Sanción:122**

* **Clase: Sanción**

-Método miSancion()

**Particiones equivalencias:** [Objeto Sancion existe], [Objeto Sancion no existe]

**Valores límites:** No existen.

**Valores posibles:** Objeto exista en sanción como: Sancion=new Sancion("zapatacamila@gmail.com","Cien años de soledad"), Objeto no exista en sanción como: Sancion=new Sancion("villaltamaria @gmail.com","Cien años de soledad")

**Nº de pruebas:** 2

-Método comprobarSancion(String email)

**Particiones equivalencias:** [Existe Sancion], [No Existe Sancion]

**Valores límites:** No existen.

**Valores posibles:** zapatacamila@gmail.com, villaltamaria@gmail.com

**Nº de pruebas:** 2

-Método anyadirSancion(Sancion sanción),

Se hizo en conjunto para saber si se añadían los prestamos al array.

**Particiones equivalencias:** [Sancion añadido],[Sancion no añadido]

**Valores límites:** No existen.

**Valores posibles:** Sancion nuevaSancion=new Sancion("lau\_aguirre@hotmail.com","La chica de nieve"), no añadir prestamo.

**Nº de pruebas:** 1.

-Método getNumero\_dias\_sancion()

Usamos el objeto *Sancion nuevaSancion=new Sancion("lau\_aguirre@hotmail.com","La chica de nieve");*

**Particiones equivalencias:** [Número días sanción]

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, pero se prueba invocando el método con el objeto nuevaSancion.

**Nº de pruebas:** 1.

-Método getEmailSocio()

Usamos el objeto *Sancion nuevaSancion=new Sancion("lau\_aguirre@hotmail.com","La chica de nieve");*

**Particiones equivalencias:** [Email del sancionado]

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, pero se prueba invocando el método con el objeto nuevaSancion.

**Nº de pruebas:** 1

-Método getTitulo\_libro\_prestamo()

Usamos el objeto *Sancion nuevaSancion=new Sancion("lau\_aguirre@hotmail.com","La chica de nieve");*

**Particiones equivalencias:** [Titulo libro sancionado]

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, pero se prueba invocando el método con el objeto nuevaSancion.

**Nº de pruebas:** 1

-Método getMis\_sanciones()

**Particiones equivalencias:** [Lista prestamos]

**Valores límites:** No existen.

**Valores posibles:** No se tiene puesto a que no recibe nada por parámetro, solo devuelve, pero se prueba comparando con un arraylist ArrayList<Sancion> esperado=listadoSanciones, y un objeto Sancion.

**Nº de pruebas:** 1

**Nº total de pruebas de la clase Sanción:9**

* **Clase: Usuario**

-Método Registrarse(String nombre,String apellidos,String fecha,String email,String contrasenya1)

EMAIL

**Particiones equivalencias:** [Nombre]

**Valores límites:** No existen.

**Valores posibles:** Juan

APELLIDO

**Particiones equivalencias:** [Apellido]

**Valores límites:** No existen.

**Valores posibles:** Gomez

EMAIL

**Particiones equivalencias:** [Email]

**Valores límites:** No existen.

**Valores posibles:**juan@gmail.com

CONTRASEÑA

**Particiones equivalencias:** [Contraseña]

**Valores límites:** No existen.

**Valores posibles:** 123.

**Nº total de pruebas de la clase Sanción:1**

* **Clase: Socio**

En todos los métodos se usa el objeto *Socio mi\_socio= new Socio("Antonella", "Gutierrez","12-12-1999", "anto123@gmail.com", "anto123"))*, y de allí hacemos las pruebas.

-Método getNombre

**Particiones equivalencia:** [“Devuelve el nombre del socio”].

**Valores limites:** No existen

**Valores posibles:** Antonella.

**Nº de pruebas:** 1.

-Método getApellidos

**Particiones equivalencia:** [“Devuelve el apellido del socio”].

**Valores limites:** No existen

**Valores posibles:** Gutierrez.

**Nº de pruebas:** 1.

-Método getFecha\_nacimiento

**Particiones equivalencia:** [“Devuelve la fecha de nacimiento del socio”].

**Valores limites:** No existen

**Valores posibles:**: 12-12-1999.

**Nº de pruebas:** 1.

-Método getEmail

**Particiones equivalencia:** [“Devuelve el Email del socio”]

**Valores limites:** No existen

**Valores posibles:** anto123@gmail.com.

**Nº de pruebas:** 1.

-Método getContrasenya

**Particiones equivalencia:** [“Devuelve la contraseña del socio”]

**Valores limites:** No existen

**Valores posibles:** anto123

**Nº de pruebas:** 1.

**Nº total de pruebas de la clase Socio:5**

* **Paquete Presentación**

**Clase: ClasePrincipal**

-Método formatoFechaCorrecto(String fecha)

**Particiones equivalencia:** [Fecha formato correcto], [Fecha formato incorrecto]

**Valores limites:** No existen

**Valores posibles:** “14-08-1998", "no-es-una-fecha"

**Nº de pruebas:** 2.

**Nº total de pruebas de la clase ClasePrincipal:2**

**Pruebas totales a realizar**=165.

**40% pruebas**=99 pruebas.