Consigna:

Una biblioteca desea administrar los préstamos de sus libros.
Cada libro tiene una ficha de préstamos en donde se asientan todos los préstamos realizados del libro. Cada socio tiene una lista de préstamos donde se asientan los préstamos en su poder. Los socios pueden ser estudiantes o docentes. Un socio estudiante puede pedir prestado un libro sólo si no tiene más de 3 libros en su poder y si no tiene ningún libro vencido (20 días después de la fecha de retiro). Un socio docente no tiene límite en cuanto a la cantidad de libros prestados, pero solo puede pedir prestado un libro si no tiene alguna devolución vencida (superados **los días de préstamos que tiene cada docente,** desde la fecha de retiro). Inicialmente todos los docentes tienen 5 días de préstamo, pero si es responsable se le van agregan días de préstamo, individualmente.
Al pedir un préstamo, éste se crea con fecha de retiro solamente; la biblioteca asienta el préstamo en la ficha del libro y en la ficha del socio. Cuando se devuelve el libro, en el préstamo se indica la fecha de devolución.

La administración de la biblioteca necesita informes que indiquen:

- Qué cantidad de socios de un determinado tipo tienen (según un tipo de socio dado).
- Que libros están prestados, cuyo plazo de devolución se encuentre vencido.
- Que socio tiene en su poder determinado Libro.
- Que docentes devolvieron los libros siempre antes de su vencimiento (o el mismo día).

1.1. Clase Biblioteca

- **prestarLibro ():** Crea el préstamo, y lo agrega en el libro y el socio.
- **devolverLibro (Libro):** Asigna la fecha de devolución del préstamo con la fecha actual.
- **prestamosVencidos():** devuelve una colección con los préstamos vencidos al día de la fecha.
- **docentesResponsables():** devuelve un colección con los docentes responsables.
- **quienTieneElLibro(String):** devuelve el nombre del Socio que tiene el libro con el título ingresado, y si no se encuentra prestado devuelve "El libro se encuentra en la biblioteca".
- **buscarSocio(int):** devuelve el Socio que tiene el dni pasado como parámetro, o null si no lo encuentra.
- **listaDeSocios():** devuelve un String según formato1.
- **listaDeLibros():**devuelve un String según formato2.
- **listaDeDocentesResponsables():**devuelve un String según formato3.

1.2. Clase Libro

- **getPrestamo():** retorna el último préstamo del libro.
- **prestado():** devuelve true si el libro se encuentra prestado.
- **toString():** devuelve el siguiente String:

Titulo: *<<titulo>>*
Ejemplo:
Titulo: JAVA. Como Programar

1.3. Clase Socio

- **cantLibrosPrestados():** devuelve la cantidad de libros en poder del Socio.
- **toString():** devuelve el siguiente String:

D.N.I.: *<<dni>>* || *<<nombre y apellido>>* ([Docente|Estudiante])
Ejemplo:
D.N.I.: 14524782 || Juan Perez (Docente) || Libros Prestados: 6

- **puedePedir():** devuelve true si el socio no tiene ningún préstamo vencido.

1.4. Clase Docente

- **esResponsable():** devuelve true si el Docente nunca tuvo ni tiene un préstamo vencido.
- **soyDeLaClase():** devuelve el String "Docente".
- **agregarDiasDePrestamos():** adiciona días de préstamo al docente. Es un premio a la responsabilidad.

1.5. Clase Estudiante

- **soyDeLaClase():** devuelve el String "Estudiante".
- **puedePedir():** devuelve true si el estudiante no tiene ningún préstamo vencido y si no tiene más de 3 libros prestados.

1.6. Clase Prestamo

- **vencido(Calendar):** devuelve true si la fecha pasada como parámetro es mayor que la fecha de vencimiento (fecha de retiro más los días de préstamos asignados).
- **toString():** devuelve el siguiente String:

Retiro: *<<fecha de retiro>>* - Devolución: *<<fecha de devolución>>*
Libro: *<<título del libro>>*
Socio: *<<nombre del socio>>*

Ejemplo:

Retiro: 2009/10/15 - Devolución: 2009/10/28
Libro: JAVA. Como Programar
Socio: Juan Perez

Formatos de listados

Formato1: Listado de Socios

Lista de Socios:

1) D.N.I.: *<<dni>>* || *<<nombre y apellido>>* (*<<tipo>>*) || Libros Prestados: *<<cant. prést. actuales >>*
2) D.N.I.: *<<dni>>* || *<<nombre y apellido>>* (*<<tipo>>*) || Libros Prestados: *<<cant. prést. actuales >>*
3) D.N.I.: *<<dni>>* || *<<nombre y apellido>>* (*<<tipo>>*) || Libros Prestados: *<<cant. prést. actuales >>*

Ejemplo:

Lista de Socios:

1) D.N.I.: 14524782 || Juan Perez (Docente) || Libros Prestados: 6
2) D.N.I.: 17982110 || Juan Fernández (Docente) || Libros Prestados: 1
3) D.N.I.: 10912002 || María Alegre (Docente) || Libros Prestados: 0
4) D.N.I.: 28987498 || Francisco Paenza (Estudiante) || Libros Prestados: 2
5) D.N.I.: 31987123 || Cesar Milstein (Estudiante) || Libros Prestados: 3
6) D.N.I.: 32874012 || Karina Leloir (Estudiante) || Libros Prestados: 0

****
Cant. Socios tipo Estudiante: 3
Cant. Socios tipo Docente: 3
****

Formato2: Listado de Libros

Lista de Libros:

1) Titulo: *<<titulo>>* || Prestado: (*<<Si|No>>*)
2) Titulo: *<<titulo>>* || Prestado: (*<<Si|No>>*)
3) Titulo: *<<titulo>>* || Prestado: (*<<Si|No>>*)

Ejemplo:

Lista de Libros:

1) Titulo: JAVA. Como Programar || Prestado: (No)
2) Titulo: Longman. Diccionario Pocket || Prestado: (Si)
3) Titulo: Vivir para contarla || Prestado: (No)

Formato3: Listado de Docentes Responsables

Lista de Socios:

* D.N.I.: *<<dni>>* || *<<nombre y apellido>>* (*<<tipo>>*) || Libros Prestados: *<<cant. prést. actuales >>*
* D.N.I.: *<<dni>>* || *<<nombre y apellido>>* (*<<tipo>>*) || Libros Prestados: *<<cant. prést. actuales >>*

Ejemplo:

Lista de Docentes Responsables:

* D.N.I.: 14524782 || Juan Perez (Docente) || Libros Prestados: 6
* D.N.I.: 17982110 || Juan Fernández (Docente) || Libros Prestados: 1

2. Crear una clase **GestionBiblioteca**, donde se deben crear objetos de las clases declaradas previamente, y obtener los listados descriptos. Los datos pueden ser ingresados por teclado o como constantes.

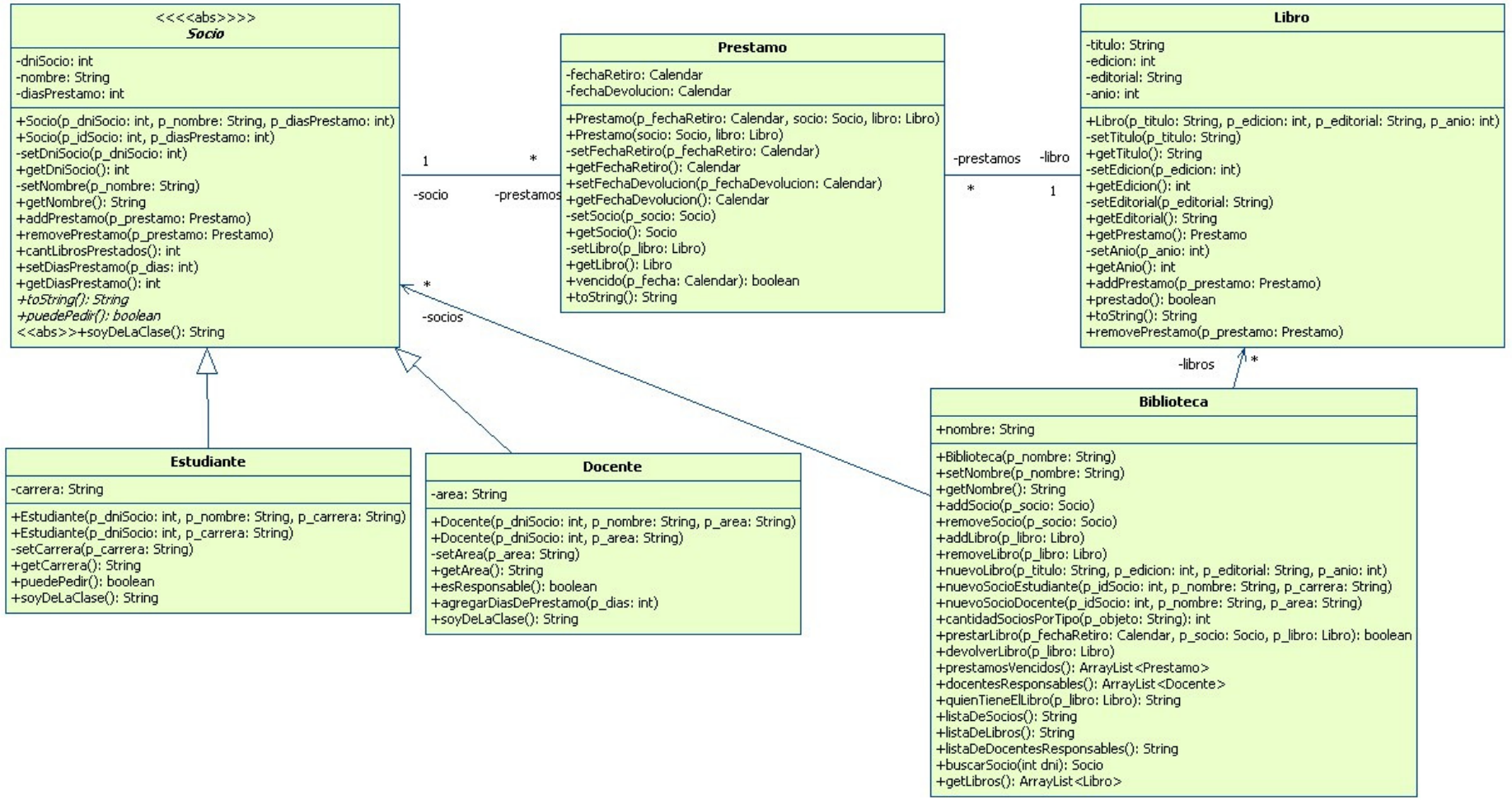
Usar los métodos apropiados para responder a las siguientes preguntas:

- ¿Qué cantidad de socios de tipo Estudiante hay?
- ¿Cuál es la lista de docentes que nunca han adeudado ni adeudan libros?
- ¿Cuál es la lista de libros? ¿Y la de socios?
- ¿Qué socio tiene prestado el libro “Programando con JAVA”?

3. Generación y presentación del trabajo:

3.1. En todos los casos se deberá usar colecciones genéricas

3.2. Todas las clases deben estar dentro del paquete “biblioteca”.


<<<<abs>>>>
Socio
-dniSocio: int
-nombre: String
-diasPrestamo: int
+Socio(p_dniSocio: int, p_nombre: String, p_diasPrestamo: int)
+Socio(p_idSocio: int, p_diasPrestamo: int)
-setDniSocio(p_dniSocio: int)
+getDniSocio(): int
-setNombre(p_nombre: String)
+getNombre(): String
+addPrestamo(p_prestamo: Prestamo)
+removePrestamo(p_prestamo: Prestamo)
+cantLibrosPrestados(): int
+setDiasPrestamo(p_dias: int)
+getDiasPrestamo(): int
+toString(): String
+puedePedir(): boolean
<<abs>>+soyDeLaClase(): String
1
*
-socio
-prestamos
*
-socios
Prestamo
-fechaRetiro: Calendar
-fechaDevolucion: Calendar
+Prestamo(p_fechaRetiro: Calendar, socio: Socio, libro: Libro)
+Prestamo(socio: Socio, libro: Libro)
-setFechaRetiro(p_fechaRetiro: Calendar)
+getFechaRetiro(): Calendar
+setFechaDevolucion(p_fechaDevolucion: Calendar)
+getFechaDevolucion(): Calendar
-setSocio(p_socio: Socio)
+getSocio(): Socio
-setLibro(p_libro: Libro)
+getLibro(): Libro
+vencido(p_fecha: Calendar): boolean
+toString(): String
-prestamos
*
-libro
1
Libro
-titulo: String
-edicion: int
-editorial: String
-anio: int
+Libro(p_titulo: String, p_edicion: int, p_editorial: String, p_anio: int)
-setTitulo(p_titulo: String)
+getTitulo(): String
-setEdicion(p_edicion: int)
+getEdicion(): int
-setEditorial(p_editorial: String)
+getEditorial(): String
+getPrestamo(): Prestamo
-setAnio(p_anio: int)
+getAnio(): int
+addPrestamo(p_prestamo: Prestamo)
+prestado(): boolean
+toString(): String
+removePrestamo(p_prestamo: Prestamo)
-libros
*
Biblioteca
+nombre: String
+Biblioteca(p_nombre: String)
+setNombre(p_nombre: String)
+getNombre(): String
+addSocio(p_socio: Socio)
+removeSocio(p_socio: Socio)
+addLibro(p_libro: Libro)
+removeLibro(p_libro: Libro)
+nuevoLibro(p_titulo: String, p_edicion: int, p_editorial: String, p_anio: int)
+nuevoSocioEstudiante(p_idSocio: int, p_nombre: String, p_carrera: String)
+nuevoSocioDocente(p_idSocio: int, p_nombre: String, p_area: String)
+cantidadSociosPorTipo(p_objeto: String): int
+prestarLibro(p_fechaRetiro: Calendar, p_socio: Socio, p_libro: Libro): boolean
+devolverLibro(p_libro: Libro)
+prestamosVencidos(): ArrayList<Prestamo>
+docentesResponsables(): ArrayList<Docente>
+quienTieneElLibro(p_libro: Libro): String
+listaDeSocios(): String
+listaDeLibros(): String
+listaDeDocentesResponsables(): String
+buscarSocio(int dni): Socio
+getLibros(): ArrayList<Libro>
Estudiante
-carrera: String
+Estudiante(p_dniSocio: int, p_nombre: String, p_carrera: String)
+Estudiante(p_dniSocio: int, p_carrera: String)
-setCarrera(p_carrera: String)
+getCarrera(): String
+puedePedir(): boolean
+soyDeLaClase(): String
Docente
-area: String
+Docente(p_dniSocio: int, p_nombre: String, p_area: String)
+Docente(p_dniSocio: int, p_area: String)
-setArea(p_area: String)
+getArea(): String
+esResponsable(): boolean
+agregarDiasDePrestamo(p_dias: int)
+soyDeLaClase(): String