Desarrollé un programa en Java con el objetivo de simular de forma sencilla el funcionamiento de un sistema de biblioteca. La idea principal fue aplicar los conceptos de la programación orientada a objetos mientras se construía un sistema que permitiera visualizar información de un libro, un usuario, y un préstamo. Durante el proceso me enfoqué en estructurar el código de manera clara y comprensible, procurando que cada clase tuviera una responsabilidad específica dentro del programa.

Para comenzar, definí una clase “person”, la cual funciona como base para representar a cualquier persona dentro del sistema. En esta clase incluí el atributo “name”, que guarda el nombre de la persona, y un método llamado “getName” que permite obtener ese dato sin acceder directamente a la variable. Esto me ayudó a aplicar el concepto de encapsulamiento, ya que la información se mantiene protegida y solo puede consultarse mediante métodos. También declaré el método abstracto “showRole”, que obliga a las clases que heredan de esta a definir cómo mostrar su rol, lo que me permitió trabajar con el principio de abstracción.

Después implementé la clase “LibraryUser”, que hereda de la clase Person. En esta clase agregué el atributo “id”, el cual identifica al usuario dentro del sistema. Para acceder a este dato utilicé el método “getId”. Además, sobrescribí el método “showRole()” para que muestre el mensaje correspondiente al rol del usuario. Aquí pude aplicar herencia, ya que el comportamiento del método cambia dependiendo del tipo de objeto.

La siguiente parte fue la creación de la clase “Book”, que representa el libro dentro del sistema. En esta clase utilicé los atributos privados “title”, “author” y “available”, los cuales almacenan el título del libro, el nombre del autor y si el libro está disponible o no. Implementé el método “showBook” para mostrar la información del libro en pantalla, el método “borrowBook” para simular un préstamo cambiando el estado de disponibilidad y el método “returnBook” para devolverlo.

También desarrollé la clase “lLoan”, que se encarga de representar el registro del préstamo. En ella incluí los atributos “startDate” y “endDate”, que guardan las fechas de inicio y fin del préstamo. El método “showLoan” muestra estos datos, lo que ayuda a visualizar la información asociada a la operación realizada.

En la clase principal, donde se encuentra el método “main”, implementé el menú interactivo que permite ejecutar las diferentes funciones del sistema. Para capturar los datos utilicé la clase “Scanner”, lo que facilita la interacción con el usuario desde la consola. Dentro de este método creé objetos de tipo Book, LibraryUser y Loan con datos de ejemplo, de modo que el programa pudiera probarse sin necesidad de ingresar información compleja desde el inicio.

El menú se diseñó usando una estructura de repetición “do-while”, lo que permite que el programa se mantenga en ejecución hasta que el usuario seleccione la opción de salir. Para manejar las opciones utilicé una estructura “switch”, lo que hace que el código sea más ordenado y fácil de entender. También añadí una validación básica para asegurar que la opción ingresada sea un número, evitando que el programa falle por una entrada incorrecta.

Durante la etapa de pruebas, ejecuté el programa varias veces seleccionando cada opción del menú para verificar que todo funcionara correctamente. Al elegir la opción de ver el libro, se mostraron los datos almacenados en los atributos “title” y “author”, junto con el estado de disponibilidad. Al consultar el usuario, el sistema mostró el rol definido en “showRole”, además de su nombre y su identificador. Cuando probé la opción de realizar un préstamo, el método “borrowBook” cambió el estado del libro a no disponible y se mostró la información del préstamo mediante “showLoan”. lo, cuál, me permitió confirmar que la lógica del programa estaba funcionando como esperaba.

Además, logré que el menú se repitiese correctamente después de cada acción, lo que facilita la interacción continua con el sistema. También verifiqué que el programa finaliza de manera adecuada al seleccionar la opción de salida, cerrando el objeto “Scanner” para evitar problemas de recursos.

Al final de todo y para concluir, desarrollar este sistema de gestión de bibliotecas me permitió comprender de forma práctica cómo se construye un programa orientado a objetos aunando todo lo que se aprendió durante esta materia, además de reforzar conceptos y aprendizajes como lo son por ejemplo la herencia, el encapsulamiento, y la abstracción; además de aprender la importancia de probar el sistema para asegurar que cada parte cumpla su función, y de no olvidar estar subiendo las diferentes versiones a mi repositorio de GitHub, lo cuál a su vez y de algún modo, me ayudó a aprender a como debo de organizarme mejor y pensar de manera lógica y estructurada.