PROYECTO: “PROGRAMA PARA LA BIBLIOTECA DE LA UNIVERSIDAD NACIONAL ANDRÉS BELLO”

Autor: Christian Omar Soriano Sandoval

c.sorianosandoval@uandresbello.edu

Profesor: MATÍAS VARGAS MARÍN

PROGRAMACIÓN AVANZADA 202205_1847_APTC101

Carrera: Ingeniería en computación e Informática.

Santiago, abril 2022.

INDICE:

INTRODUCCIÓN:

Este informe consiste en buscar una solución a un problema detectado, mediante las herramientas de ingeniería, y aplicando los conocimientos de programación avanzada aprendidos hasta ahora. El entorno donde se desarrolla la solución al problema es el lenguaje de programación “Java”, y en este caso puntual, fue realizado con la IDE “Intelli J IDEA” siguiendo los parámetros de la programación orientada a objetos.

El caso corresponde a un requerimiento del equipo de la biblioteca de la Universidad Nacional Andrés Bello para la sede de Antonio Varas, que consiste en actualizar el sistema de biblioteca de dicha institución, para ello el establecimiento ha dispuesto de una serie de requerimientos, que serán enumerados más adelante.

El objetivo del proyecto es crear un programa que sea capaz de almacenar datos de Usuarios y libros, de tal forma que pueda existir un control sobre los libros que hay en la biblioteca, además de poder identificar al usuario que lo tiene.

En cuanto al informe, tiene como objetivo explicar como funciona el sistema, como se abordó su diseño y lo que se logró hacer con él.

La estructura del programa en sí es simple, y se visualiza en una interfaz muy poco gráfica, pero cumple con el objetivo planteado en el problema: construir un programa aplicando los fundamentos de la programación orientado al objeto a través del diseño de un diagrama de clases UML, para dar solución a este problema computacional simple, utilizando archivos externos para guardar la base de datos, de tal forma que la información siga ahí cuando el programa deja de estar “vivo”.

DESCRIPCIÓN DEL PROBLEMA:

El equipo de biblioteca de la Universidad Nacional Andrés Bello para la sede de Antonio Varas le ha solicitado a usted y a su equipo actualizar el sistema de biblioteca de dicha institución, para ello el establecimiento ha dispuesto la siguiente lista de requerimientos:

1. Requerimiento usuarios:
    1.1. Cada usuario tiene los siguientes atributos
        1.1.1. Nombre completo.
        1.1.2. RUN.
        1.1.3. Género.
        1.1.4. Carrera.
        1.1.5. Préstamo.
    1.2. Crear y editar usuario
        1.2.1. El RUN no puede repetirse.
        1.2.2. Debe validar formato y dígito verificador del RUN.
        1.2.3. Debe validar que el género del usuario sea M o F.
        1.2.4. Préstamo corresponde a si el usuario tiene en su poder o no un libro, siendo cero no, e ISBN que sí.
    1.3. Eliminar usuario, debe validar que exista.
    1.4. Existen usuarios que son Docentes o Estudiantes. A los docentes adicionalmente se les registra la profesión con sus grados (magíster y/o doctor); y a los estudiantes la carrera que está estudiando.

2. Requerimiento Libro:
    2.1. Cada libro debe contener los siguientes atributos
        2.1.1. ISBN (Código libro).
        2.1.2. Título.
        2.1.3. Autor.
        2.1.4. Cantidad en biblioteca.
        2.1.5. Cantidad disponible préstamo.
        2.1.6. Imagen.
    2.2. Crear Libro
        2.2.1. El ISBN debe ser único.
        2.2.2. Cantidad en biblioteca debe ser mayor a cero
        2.2.3. Cantidad disponible debe ser mayor a cero y menor o igual a cantidad en biblioteca.
    2.3. Eliminar Libro a través de ISBN, debe validar que exista.

3. Requerimiento Préstamo y Devolución
    3.1. Préstamo
        3.1.1. Debe ingresar ISBN del Libro a prestar.

3.1.2. El libro debe existir.
3.1.3. El libro debe tener al menos un ejemplar disponible.
3.1.4. Debe ingresar RUN usuario quien solicitó el libro.
3.1.5. El RUN debe existir.
3.1.6. El usuario debe estar habilitado para préstamo.
3.1.7. De cumplirse todos los puntos anteriores, el usuario debe quedar no disponible para nuevo préstamo a través del atributo préstamo, dejando el ISBN del libro a prestar.
3.1.8. De cumplirse todos los puntos anteriores, debe restar uno a la cantidad de ejemplares disponibles del libro.
3.1.9. Para el nuevo préstamo debe ingresar la fecha actual, de forma automática.
3.1.10. Debe ingresar la cantidad de días que se prestó el libro (si es docente el período máximo de préstamo es de 20 días y si es estudiante el período máximo es 10 días).
3.1.11. Debe ingresar la fecha que debe ser devuelto el libro de forma automática.
3.1.12. Finalmente generar la tarjeta de préstamo que será impresa y pegada en el libro que contenga los datos del préstamo (diseñe un formato ad-hoc).

ANÁLISIS Y DISEÑO DE LA SOLUCIÓN

Para en análisis de la solución, se utilizó el diagrama de clases que se presenta a continuación:

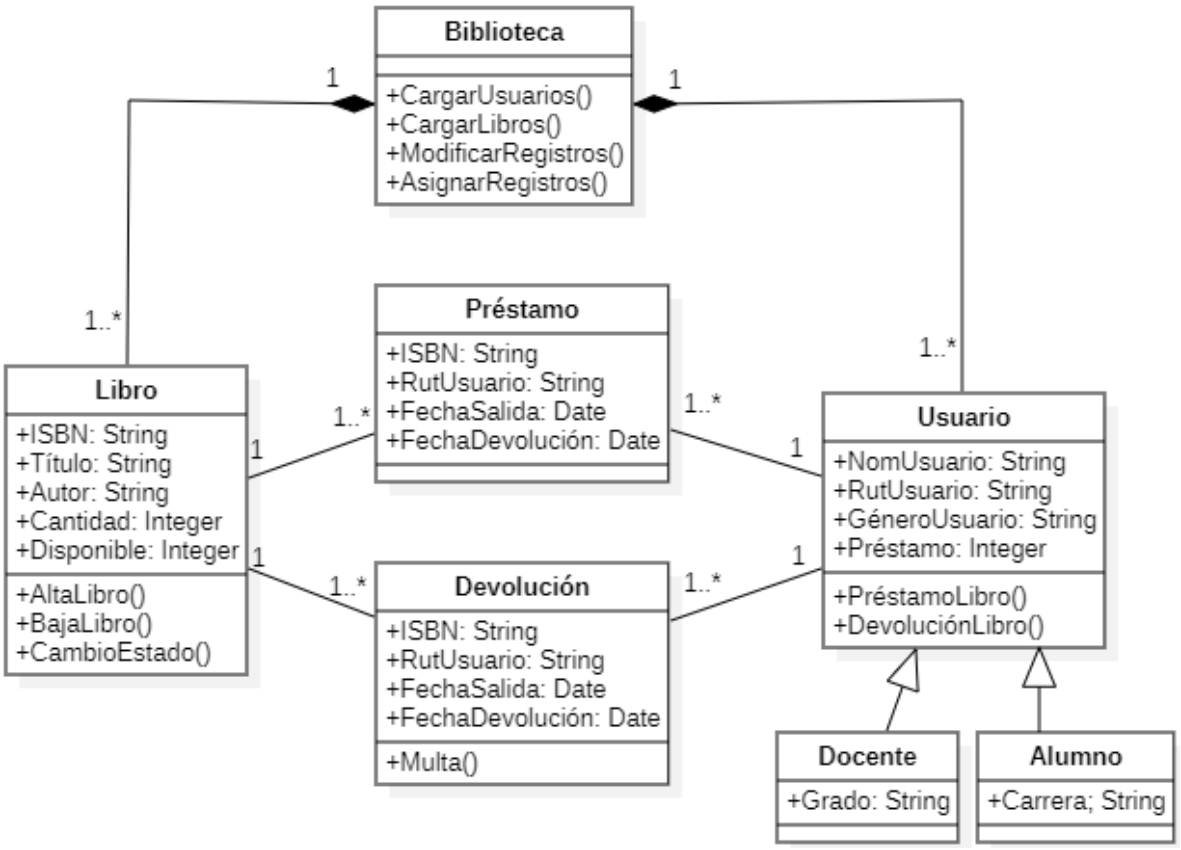


Diagrama de clases, elaboración propia en software "Star UML".

El diagrama se explica de la siguiente forma:

- ✓ En la biblioteca se deben cargar los libros y usuarios para poder contar con los registros que alimentan la base de datos y poder entrelazar los datos entre sí.
- ✓ Cada usuario es único y puede solicitar o devolver 1 o más libros.
- ✓ El usuario se divide en Docente y alumno, heredando todos los atributos de Usuario más el atributo "grado" en el caso del docente y "carrera" en el caso de alumno.
- ✓ Cada préstamo o devolución de cada libro se carga a un usuario.
- ✓ La multa se cobrará dependiendo de la fecha de préstamo y devolución del libro y se hará efectiva al usuario.

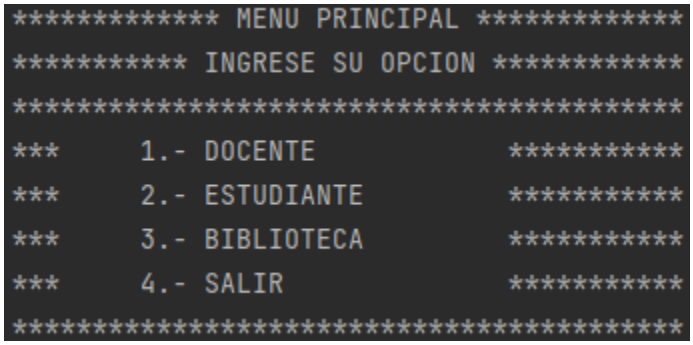
ASPECTOS Y FUNDAMENTOS DE PROGRAMACIÓN:

La programación de este sistema se basa en el almacenamiento de información mediante un archivo .txt que nos servirá para que, una vez cerrado el programa, puedan quedar los registros grabados ahí (con formato de texto plano), de tal forma que cuando abramos nuevamente el programa, se enlace con el archivo .txt y mantengamos los datos que quedaron almacenados ahí.

Con esta forma de almacenar datos, aseguramos que la información que hemos ingresado o modificado, no se borre cuando cerramos el programa.

Para comenzar, se abre el menú principal (Main) que nos permite elegir si vamos a ingresar, ver o modificar información sobre docentes, estudiantes o la biblioteca.

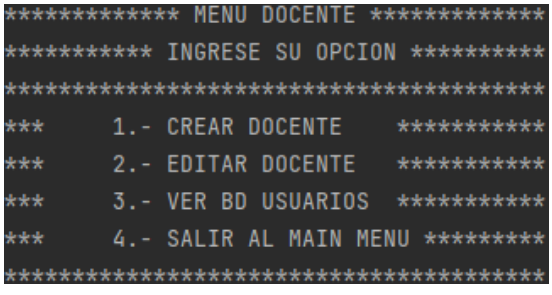
También se encuentra la opción salir, que finaliza el programa.

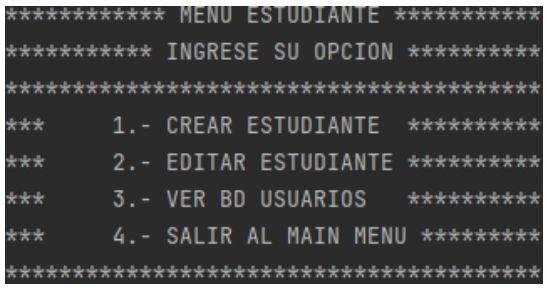


En un principio, se pensó de esta forma ya que se solicitaba definir un atributo especial para docente y estudiante, pero luego que avanzó el desarrollo, descubrí que era mejor juntar todos los atributos en “usuario” y heredarlos a docente y estudiante más su atributo extra.

Se podría mejorar este menú, haciendo una opción que permita ingresar o modificar el usuario y una vez ingresados los datos, que pregunte si eres docente o estudiante, y derivar ese atributo especial; pero debido al poco tiempo que hubo para hacer el programa, no fue posible mejorarlo. Para solucionar de forma rápida este detalle, se identifica al usuario con la palabra “Estudiante” o “Docente” al final de la línea, de forma automática, dependiendo si se entró a la opción docente o estudiante.

Dicho esto, se muestran ambos menús, ya que mantienen la misma estructura:





En un comienzo, tenía la idea de hacer bases de datos separadas de docentes y alumnos, pero luego, al enlazar la información con el libro asignado, era necesario que el programa buscara en un solo

archivo la información, y decidí unirlos en una sola base de datos llamada “DatosUsuarios.txt” ya que puede ser posible que tanto un alumno como un docente puedan solicitar el préstamo de un libro. En el archivo se pueden diferenciar porque tienen un campo adicional que contiene el grado académico para los docentes y la carrera que está cursando el estudiante, entonces nace “usuario” como “padre” y el campo adicional “hijo” que entrega el atributo adicional si es docente o estudiante (mas todos los atributos de usuario).

Como método adicional, se agregó al final de cada línea la palabra “Estudiante” o “Docente” de forma automática, dependiendo si se entró a la opción docente o estudiante, como lo comenté anteriormente.

Para el caso de editar un usuario, el programa solicita el rut, y con ese dato recorre las líneas de la base de datos, y si no lo encuentra indica que el usuario no está creado.

Si lo encuentra, muestra el siguiente mensaje:

```
2
Ingresa el rut del estudiante a buscar
13436078k
SE ENCONTRO EL REGISTRO: 13436078k | christian omar soriano sandoval | M | ingeniero en gestion logistica | doctorado supply chain
INGRESE LOS DATOS PARA MODIFICAR EL REGISTRO:

Ingrese el rut
```

Posteriormente, solicita todos los datos de ingreso del usuario, los guarda en memoria, los mantiene en un archivo temporal y por último pisa el archivo anterior manteniendo los demás datos almacenados. En síntesis, borra la línea e ingresa una nueva.

En el caso del rut, el sistema valida la longitud de caracteres y acepta solo una k como letra al final del número. En el caso del género, valida que la opción sea M o F.

Para la opción 3. Ver BD usuarios se llama al método “mostrararchivos()” que abre “DatosUsuarios.txt”, y recorre las líneas mostrándolas una a una en forma de listado.

La Opción 4 nos lleva al menú principal.

Para el menú “Biblioteca” en main, se muestra el siguiente submenú:

```
************ MENU BIBLIOTECA ************
*********** INGRESE SU OPCION ***********
*****************************************
***     1.- CREAR LIBRO       ***********
***     2.- EDITAR LIBRO      ***********
***     3.- VER BD LIBROS     ***********
***     4.- ASIGNAR LIBRO     ***********
***     5.- VER BD ASIGNACIONES *********
***     6.- SALIR AL MAIN MENU **********
*****************************************
```

Para crear, editar y ver la base de datos, se siguió la misma metodología que con los submenús de usuarios.

En el caso de la opción 4. Asignar libro, lo que hace el programa es abrir y revisar en primer lugar el archivo “DatosUsuarios.txt”, recorrer cada línea hasta encontrar el número de ISBN (validando que tenga 13 caracteres), lo retiene en memoria, abre el siguiente archivo “DatosLibro.txt” y recorre las líneas hasta encontrar el rut (validando que tenga 9 dígitos y que pueda recibir una k) para mantenerlo también en memoria.

Si encuentra los dos datos, los concatena e inserta en el archivo “DatosAsignacion.txt” y crea una nueva línea con los datos del libro y la persona que lo va a tener.

Si no encuentra uno de los dos datos solicitados, el programa arrojará un mensaje diciendo que no se encuentra el libro, o en el caso del usuario, que debe ser creado.

Para la lógica del programa, se debería usar un código adicional interno en los libros que sea distinto al ISBN para que se identifique cada uno de los ejemplares como único, y de esta forma cada una de las impresiones del libro podrían ser asignadas a cada persona. Se piensa como idea para poder mejorarlo más adelante.

En la opción 5, muestra el listado rescatado del archivo “DatosAsignacion.txt” para saber quien tiene cada libro, y si es estudiante o docente.

La opción 6 sale al menú principal.

Las opciones de los menus son validadas de acuerdo a el número de opciones disponibles, en caso de escribir otro número, el programa advierte que debe ingresar una opción válida.

Para las demás opciones requeridas en este ejercicio, como el conteo del inventario de las copias de los libros, o la resta del libro para que quede disponible, no se utilizó un contador. Se sistematizó con la concatenación de datos, entendiendo que el libro que se encuentra con un rut asignado se encuentra fuera de la biblioteca, entendiendo para el ejercicio que en la biblioteca hay una copia por cada libro.

Finalmente, debido a factores de tiempo y feedback, no se alcanzó a insertar las fechas de retiro o devolución del libro, y por tanto el programa no avisará cuando haya una multa. Se asume que el usuario deberá contar los días y cobrar la multa si correspondiera.

Una salida rápida a este problema sería insertar un campo con la fecha y los días que el usuario mantendrá el libro, y asignar una multa si se pasa de x días con un comando “IF” y una simple operación matemática.

CONCLUSIONES:

Con lo aprendido hasta ahora, es posible que el programa no reúna todas las condiciones para ser un entregable final, ya que hay algunos criterios que no han podido ser abordados por el poco tiempo de entrega de este trabajo.

Las validaciones y restricciones son simples y es posible mejorarlo con el tiempo, para lograr hacer un programa robusto y adecuado a todas las necesidades de la biblioteca.

Se logra el objetivo de crear un archivo anexo que permita mantener los datos ingresados una vez que se cierra el programa, de tal forma que no se pierda la información y sea posible mantener una base de datos sencilla pero eficiente y que no se borre cada vez que se finaliza.

Con esta base de datos sencilla, es posible ingresar datos, modificarlos, eliminarlos, y visualizarlos en forma de tabla o listado, logrando obtener información relevante de usuarios y/o libros y mantener el control de que usuario mantiene cual libro.

En comparación al trabajo anterior, estimo que es superior en cuanto a estructura, dificultad y eficiencia, sin embargo, siento que, con más tiempo, será posible mejorar muchas partes de él, y sin duda será una meta personal a lograr para mí.

Lo que me queda como conclusión final es que todo programa se puede mejorar, y a medida que se va avanzando en su programación van apareciendo mejoras y funcionalidades adicionales o métodos distintos, y es posible que se deba comenzar desde cero varias veces.

Todo el esfuerzo se ve reflejado cuando el programa logra arrancar y cumple su función (es muy motivador).

BIBLIOGRAFÍA:

Material del ramo “Programación avanzada” UNAB.

Programación en Java, de Joyanes y Zahonero (2011).

Landero. P (2021). Conceptos Principales del Lenguaje de Programación [apunte]. Chile. UNAB

Landero. P (2021). Fundamentos paradigma OO y lenguaje POO [apunte]. Chile. UNAB.

Landero, P (2021). Aplicación de Herencia y Polimorfismo [apunte]. Chile. UNAB

Joyanes, L. y Zahonero, I. (2011). Programación en Java. Algoritmos, programación orientada a objetos e interfaz gráfica de usuario.

Programación en Java || POO || Creación de clases y objetos: LINK

https://www.youtube.com/watch?v=oMWrJwMPd6k&t=147s

Manejo de archivos .txt en java (crear, ingresar, buscar, interactuar): LINK

https://www.youtube.com/watch?v=PhnEJ_Cr65k