Entrega Acumulada A

Tema: Administrador de Películas-Clientes en un videoclub

Nombres: **Luis Romero, Heinz Beckers, Sebastian Villalón**

1.1 Análisis de los datos a utilizar y principales funcionalidades a implementar que dan sentido a la realización del proyecto.

El problema que se nos presenta es lograr la implementación de un sistema que sea capaz de administrar un registro de clientes, así como de películas, el cual nos permita mediante el uso del historial de arriendo de un cliente y el stock de cintas disponibles en la tienda, sugerir a este películas que le podrían interesar arrendar y permitir que las arriende.

Dentro de las principales entidades tenemos a Película la cual almacenará los principales datos de un film, la entidad Cliente la cual contendrá datos sobre el arrendador de películas que serán necesarios para la recomendación de estas, además del manejo del negocio,la entidad Arriendo que contiene datos relacionados al arriendo de una película y por último la entidad VideoClub la cual de una manera u otra contendrá a todas las clases anteriormente mencionadas, además de información relacionada a la tienda física.

En cuanto a las principales funcionalidades que nos gustaría implementar tenemos las siguientes:

1. Administración de películas (agregar, actualizar, mostrar, eliminar).
2. Administración de clientes (agregar, actualizar, mostrar, eliminar).
3. Arrendar películas.
4. Recomendar películas.

1.2 Diseño conceptual de clases del Dominio y su código en Java

- **Clase Videoclub**

Esta clase se encarga de almacenar todos los datos relacionados a una tienda/videoclub. Entre los atributos de esta tienda tenemos los String nombre y dirección, además de arraylist y hashmap de clase Película que almacena todos los films (con su respectiva información) que posee la tienda y otro arraylist y hashmap clase Cliente que contiene todos los clientes que ha tenido la tienda (con su respectiva información).

Posee los métodos setter y getter para cada variable privada.

- **Clase Película**

Contiene las variables que almacenan los datos de la película, además de la cantidad de copias existentes, las disponibles y su id única.

Posee los métodos setter y getter para cada variable privada.

- **Clase Cliente**

Esta clase contiene el nombre y rut del cliente en atributos de tipo Strings, así como una variable llamada deuda que contiene el monto adeudado por el cliente en caso de existir y también dos Arraylist y Hashmap que contienen objetos de clase Arriendo, para almacenar los arriendo actuales y el historial de arriendo de la persona.

Posee los métodos setter y getter para cada variable privada.

- **Clase Arriendo**

Esta clase se encarga de almacenar atributos relacionados al arriendo de una película, como por ejemplo, id de la película, fechas de arriendo y devolución, valoración de la película por parte del cliente y veces arrendada por este.

Posee los métodos setter y getter para cada variable privada.

- **Clase Interface**

Esta clase contiene los métodos necesarios para la implementación y ejecución del menú por consola. No posee atributos.

- **Clase Funciones**

Clase que contiene los métodos necesarios para la implementación y ejecución de las funciones principales del proyecto. No posee atributos.

- **Clase Comprobar**

Clase que contiene los métodos necesarios para la comprobación de algunas entradas de datos. No posee atributos y se incorpora en un futuro a clase Funciones.

1.3 Todos los atributos de todas las clases deben ser privados y poseer sus respectivos métodos de lectura y escritura (getter y setter).

Clase VideoClub

```java
package Clases;
import java.util.*;
import java.io.*;

public class VideoClub {
    private String nombreTienda;
    private String direccion;
    private ArrayList<Pelicula> listaPeliculas;//Coleccion de objetos 1 anidación
    private ArrayList<Cliente> listaClientes;  //Coleccion de objetos 2 anidaciones

    private HashMap<String,Pelicula> pelisXId;   //Key es ID pelicula
    private HashMap<String,Cliente> clientesXRut; //Key es Rut cliente

    public VideoClub(){ //constructor
        listaPeliculas = new ArrayList<>();
        listaClientes = new ArrayList<>();
        pelisXId = new HashMap<>();
        clientesXRut = new HashMap<>();
    }
//------------------SETTER/GETTER-----------------
    public String getNombreTienda() { return nombreTienda; }

    public void setNombreTienda(String nombreTienda) { this.nombreTienda = nombreTienda; }

    public String getDireccion() { return direccion; }

    public void setDireccion(String direccion) { this.direccion = direccion; }
```

Clase Película

```java
package Clases;

import java.util.Arrays;

public class Pelicula {

    private String nombre;
    private String id;
    private short existencias; //Numero total de peliculas de la que es dueña la tienda
    private short disponibles; //Numero de peliculas disponibles para arrendar
    private float valuacion;   //Valoracion de la pelicula de 0 a 5 estrellas
    private short cantValuaciones; //Numero perosnas que han valorado la pelicula
    private short añoEstreno;
    private short duraciónMin;

    private String sinopsis;   //Breve sinopsis de la pelicula
    private String calidad;    //Almacena la calidad de la pelicula
    private String director[]; //Arreglo que contiene los nombres de los directores(Compacto sin plibre)
    private String actores[];  //Arreglo que contiene los nombres de los actores(Compacto sin plibre)
    private String generos[];  //Arreglo que contiene los nombres de los generos(Compacto sin plibre)

    public Pelicula(){}

//------------------SETTER/GETTER-----------------
    public void setNombre(String nombre) { this.nombre = nombre; }

    public String getNombre() { return nombre; }

    public String getId() { return id; }

    public void setId(String id) { this.id = id; }
```

```
public short getExistencias() { return existencias; }

public void setExistencias(short existencias) { this.existencias = existencias; }

public short getDisponibles() { return disponibles; }

public void setDisponibles(short disponibles) { this.disponibles = disponibles; }

public float getValuacion() { return valuacion; }

public void setValuacion(float valuacion) { this.valuacion = valuacion; }

public short getCantValuaciones() { return cantValuaciones; }

public void setCantValuaciones(short cantValuaciones) { this.cantValuaciones = cantValuaciones; }

public short getAñoEstreno() { return añoEstreno; }

public void setAñoEstreno(short añoEstreno) { this.añoEstreno = añoEstreno; }

public short getDuraciónMin() { return duraciónMin; }

public void setDuraciónMin(short duraciónMin) { this.duraciónMin = duraciónMin; }

public String getSinopsis() { return sinopsis; }

public void setSinopsis(String sinopsis) { this.sinopsis = sinopsis; }

public String getCalidad() { return calidad; }
```

```
public void setCalidad(String calidad) { this.calidad = calidad; }

public String[] getDirector() { return Arrays.copyOf(director,director.length); }

public void setDirector(String[] directores) { director = Arrays.copyOf(directores,directores.length); }

public String[] getActores() { return Arrays.copyOf(actores,actores.length); }

public void setActores(String[] actor) { actores = Arrays.copyOf(actor,actor.length); }

public String[] getGeneros() { return Arrays.copyOf(generos,generos.length); }

public void setGeneros(String[] genero) { generos = Arrays.copyOf(genero,genero.length); }
```

Clase Cliente

```java
package Clases;
import java.util.ArrayList;
import java.util.HashMap;

public class Cliente{
    private String nombre;
    private String rut;
    private ArrayList<Arriendo> historialArriendos; //Coleccion de objetos 1 anidación.
    private HashMap<String, Arriendo> historialXid;
    private ArrayList<Arriendo> arriendosActuales; //Contiene las peliculas arrendadas actuales
    private HashMap<String, Arriendo> arriendoXid;
    private int deuda;

    public Cliente() {...}

    public Cliente(String nombre, String rut){...}
```

```java
public String getNombre() { return nombre; }

public void setNombre(String nombre) { this.nombre = nombre; }

public String getRut() { return rut; }

public void setRut(String rut) { this.rut = rut; }

public int getDeuda() { return deuda; }

public void setDeuda(int deuda) { this.deuda = deuda; }

public Arriendo getHistorialXId(String id) { return historialXid.get(id); }

public Arriendo getArriendoXId(String id) { return arriendoXid.get(id); }

public Arriendo getArriendo(int i) { return arriendosActuales.get(i); }
```

Clase Arriendo

```java
package Clases;

import java.time.LocalDate;

public class Arriendo {
    private String id;               //key
    private float valoracion;        //Numero de estrellas con la que el cliente valoro el film de 0 a 5
    private LocalDate fechaArriendo; //yyyy-mm-dd
    private LocalDate fechaEntrega;  //yyyy-mm-dd fecha de devolución
    private boolean entregado;       //TRUE si la pelicula fue devuelta, False en el caso contrario
    private int vecesArrendada;
    //Hacer una coleccion de un objeto que contenga fecha de arriendo, entrega y valoracion, para tener un registro historico.

    public Arriendo(){
        id = null;
        valoracion = 0;
        fechaArriendo = null;
        fechaEntrega = null;
        entregado = true;
        vecesArrendada = 0;
    }
```

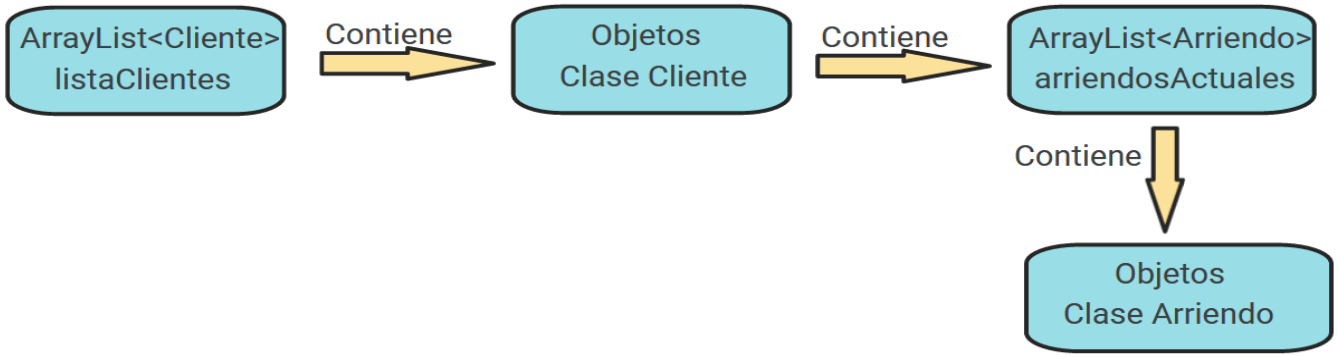
```java
//------------------SETTER/GETTER-----------------
    public String getId() { return id; }

    public void setId(String id) { this.id = id; }

    public float getValoracion() { return valoracion; }

    public void setValoracion(float valoracion) { this.valoracion = valoracion; }

    public LocalDate getFechaArriendo() { return fechaArriendo; }

    public void setFechaArriendo(LocalDate fechaArriendo) { this.fechaArriendo = fechaArriendo; }

    public LocalDate getFechaEntrega() { return fechaEntrega; }

    public void setFechaEntrega(LocalDate fechaEntrega) { this.fechaEntrega = fechaEntrega; }

    public boolean isEntregado() { return entregado; }

    public void setEntregado(boolean entregado) { this.entregado = entregado; }

    public void setVecesArrendada(int vecesArrendada) { this.vecesArrendada = vecesArrendada; }

    public int getVecesArrendada() { return vecesArrendada; }
```

1.4 Se deben incluir datos iniciales dentro del código.

Datos iniciales incluidos como archivos en la carpeta data, bajo los nombres de clientes.tsv y peliculas.tsv.

2.1 Diseño conceptual y codificación de 2 (dos) niveles de anidación de colecciones de objetos.

Se decidió usar la siguiente anidación de colecciones ubicadas en clase VideoClub y Cliente:



Lo anterior también aplica para el Hashmap<Cliente> clientesXRut, el cual contiene clientes, los cuales contienen dos Arraylist.y Hashmap de objetos Arriendo.

2.2 Diseño conceptual y codificación de 2 (dos) clases que utilicen sobrecarga de métodos.

Se tiene la siguiente sobrecarga en la clase VideoClub.

- **mostrarDatosPeliculas()**:muestra todos los datos de las películas dentro del ArrayList listaPeliculas.
- **mostrarDatosPeliculas(String id):**muestra todos los datos de una película en específico dentro del ArrayList listaPeliculas, que tiene como parámetro un String que equivale a la id de la película cuyos datos queremos obtener.

También en clase Funciones tenemos otra sobrecarga.

- **arrendar(VideoClub tienda, String rut):** Permite a un cliente identificado por el rut, arrendar un máximo de 3 películas, siempre y cuando hayan copias disponibles y la persona no tenga deudas con la tienda.
- **arrendar(VideoClub tienda):** Admite el ingreso del rut del cliente por consola, en caso de ser cliente, se le permite, arrendar un máximo de 3 películas, siempre y cuando hayan copias disponibles y la persona no tenga deudas con la tienda.

2.3 Diseño conceptual y codificación de al menos una clase mapa del JCF.

Se implementan dos HashMap en la clase VideoClub:

- **pelisXid**: Almacena las películas que posee el videoclub, a cada película se le asigna un String llamado id, la cual es única y es usada como key.
- **clientesXrut**: Almacena los clientes que posee el videoclub, el rut propio de cada cliente es usado como key.

Estos dos HashMaps acompañan a sus respectivos ArrayList, listaPeliculas y listaClientes , los cuales se usan para poder recorrer a todos los clientes y películas del videoclub.

3.1 Diseño de diagrama de clases UML

Se encuentra adjuntado en el .zip entregado y en el repositorio de github este se encuentra en la rama mismo proyecto se llama “diagrama entrega parte A”

3.2 Implementación de menú del sistema para funcionalidades

El acceso a ciertas funcionalidades de la aplicación se ha implementado mediante la creación de dos menús, uno para el cliente del videoclub y otro para el administrador/trabajador de este establecimiento. La implementación del menú se encuentra en la clase “Interface”.

En interface nos encontramos con 3 métodos principales *inicioMenu(VideoClub)*, *menuCliente()* y *menuAdmin()*. *inicioMenu(VideoClub)* se encarga de dar la opción a que tipo de menú entrar, menú cliente o administrador y en caso que se desee salir del programa también está la opción disponible.

El método login despliega las siguientes opciones:

"1)Cliente": Accede al menú de clientes.
"2)Trabajador": Accede al menú de trabajador
"0)Finalizar programa": Finaliza la ejecución del programa

El método *menúCliente* despliega las siguientes opciones:

"1)Películas": Da acceso al siguiente submenú.

"1)Desplegar catálogo de películas": Muestra el catálogo completo de películas que tiene disponible la tienda.
"2)Buscar Película": Muestra los datos guardados de una película, dado su nombre.
"2)Recomendar película": Recomienda la película mejor valorada en la tienda.
"2)Recomendar película por género": Recomienda la película mejor valorada en la tienda, según género.
"0)Menú anterior": Retorna al menú anterior.

"2)Servicios para cliente": Da acceso al siguiente submenú.

- **"1)Arrendar películas":** Permite al cliente arrendar películas.
- **"2)Devolver películas":** Permite al cliente devolver las películas arrendadas.
- **"3)Pagar deuda":** Permite al cliente pagar su deuda.
- **"0)Menú anterior":** Retorna al menú anterior.

"3)Información usuario": Da acceso al siguiente submenú.

- **"1)Desplegar mi ficha cliente":** Muestra los datos almacenados del cliente.
- **"2)Revisar mi historial":** Muestra el historial completo de películas arrendadas por el cliente.
- **"3)Revisar historial de película":** Muestra el historial de solo una película.
- **"0)Menú anterior":** Retorna al menú anterior.

"0)Menú anterior": Retorna al menú anterior.

El método menúAdmin despliega las siguientes opciones:

"1)Ingresar datos": Da acceso al siguiente submenú.

- **"1)Registrar Cliente":** Registra un nuevo cliente en el sistema.
- **"2)Registrar Película":** Registra una nueva película en el sistema.
- **"3)Registrar Historial":** Se agrega una objeto arriendo a al historial de arriendos del cliente
- **"0)Menú anterior":** Retorna al menú anterior.

"2)Mostrar datos": Da acceso al siguiente submenú.

- **"1)Desplegar lista de clientes":** Muestra una lista con los datos de todos los clientes.
- **"2)Desplegar lista de películas":** Muestra una lista con los datos de todas las películas que tiene la tienda.
- **"3)Desplegar historial de cliente":** Muestra el historial de arriendos de un solo cliente.
- **"0)Menú anterior":** Retorna al menú anterior.

"3)Buscar datos": Da acceso al siguiente submenú.

- **"1)Buscar Película":** Muestra los datos guardados de una película, dado su nombre.
- **"2)Buscar Cliente":** Muestra los datos de un cliente, dado su nombre.
- **"0)Menú anterior":** Retorna al menú anterior.

"4)Editar datos": Da acceso al siguiente submenú.

- **"1)Editar Película":** Permite cambiar los datos de una película.
- **"2)Editar Cliente":** Permite cambiar los datos de un cliente.
- **"3)Eliminar Película":** Elimina la película indicada de las colecciones correspondientes.
- **"4)Eliminar Cliente":** Elimina el cliente indicado desde las colecciones correspondientes.
- **"0)Menú anterior":** Retorna al menú anterior.

"5)Servicios para cliente": Da acceso al siguiente submenú.

"1)Arrendar películas": Permite al cliente arrendar películas.
"2)Devolver películas": Permite al cliente devolver las películas arrendadas.
"3)Pagar deuda": Permite al cliente pagar su deuda.
"0)Menú anterior": Retorna al menú anterior.

"6)Generar reportes": Genera archivos con reportes que se guardan en carpeta Reportes.

"1)Deudores": Crea archivo que contiene el nombre de todos los clientes deudores
"2)Películas en arriendo": Crea archivo que contiene información del usuario y el nombre de las películas que tiene arrendadas actualmente.

"0)Menú anterior": Retorna al menú anterior.

3.2.1 Funcionalidad básica (1): Inserción manual o agregar elemento

Mediante el menú para administradores se permite el registro de nuevos Clientes y Películas a los ArrayList y hashMap de estas entidades en la clase VideoClub:

En Funciones.java:

public static void **registrarCliente**(VideoClub x);
public static void **registraPelicula**(VideoClub x);

En VideoClub.java:

public void **addPeliToListaPelis**(Pelicula peli);
public void **addClientToListaClients**(Cliente cliente);
public void **addClientToClientXRut**(String rut, Cliente cliente);
public void **addPeliToPelisXId**(String id, Pelicula peli);

También en la clase Cliente y mediante el menú de Administrador, se permite el ingreso de un nuevo objeto de la clase Arriendo:

En Funciones.java:

public static void **registrarArriendo**(VideoClub x);

En Cliente.java:

public void **addToHistorialXid**(Arriendo arriendo);
public void **addToHistorial**(Arriendo arriendo)
public void **addToArriendosActuales**(Arriendo arriendo)

public void **addToArriendosXid**(Arriendo arriendo)

3.2.2 Funcionalidad básica (2): Mostrar por pantalla listado de elementos

Para mostrar los elementos de la listaClientes debemos ir al menú de administrador codificado en la clase Interface, donde se accede a la opción **"2)Mostrar datos"** y **"1)Desplegar lista de clientes"** la cual llama al método listaCliente(VideoClub) implementado en la clase Funciones, la cual llama a otro método implementado en la clase VideoClub llamado mostrarDatosPeliculas() la cual se encarga de mostrar por pantalla cada elemento de la lista con su s datos más importantes.

Para mostrar el historial de arriendo de un cliente, se puede acceder de dos formas dependiendo de si es un cliente o un administrador el que desea ver el array historialArriendos.

Caso de cliente:

Mediante la interfaz se accede al menú cliente ingresando el rut correspondiente, y selecciona la opción **"3)Información usuario"** y luego **"2)Revisar mi historial"** , la cual llama la función implementada en Funciones mostrarHistorialCliente(VideoClub x, String rut), la cual invoca a la función mostrarHistorial() la cual se encuentra en la clase Cliente e imprime la lista de arriendos del cliente por pantalla.

Caso administrador:

Desde el menú administrador en la clase Interface, está disponible la opción **"2)Mostrar datos"** y luego **"3)Desplegar historial de cliente"** la cual llama a un método ubicado en la clase Funciones llamado mostrarHistorialCliente(VideoClub x), la cual accede a otro método llamado mostrarHistorial() en la clase Cliente, el cual se encarga de mostrar por pantalla una lista con todos los arriendos del cliente.

A.2 Implementación de las siguientes funcionalidades en el menú:

A.2.1 Funcionalidad básica (1): Edición/modificación de elemento

En menú administrador opción **4** se puede acceder a la edición de ciertos elementos**:**

"1)Editar Película": Permite cambiar los datos de una película.
"2)Editar Cliente": Permite cambiar los datos de un cliente.

En menú administrador opción **5->2** y menú cliente opción **2->2** se puede acceder a la función ***delvolverArriendo()***, esta función al devolverse un arriendo, en caso de que no haya arrendado la película antes se agrega el objeto arriendo a las colecciones correspondientes, de lo contrario se ***editan/modifican*** los objetos ya almacenados en las colecciones correspondientes, las cuales se ubican en el segundo nivel de anidación.

A.2.2 Funcionalidad básica (2): Eliminación del elemento para las 2 colecciones anidadas.

En menú administrador opción **5->2** y menú cliente opción **2->2** se puede acceder a la función ***delvolverArriendo()***, esta función al devolverse un arriendo, elimina el objeto arriendo de las colecciones correspondientes, las cuales se ubican en el segundo nivel de anidación..

En menú administrador opción **4** se puede acceder a la eliminación de ciertos elementos.

"3)Eliminar Película": Elimina la película indicada de las colecciones correspondientes.
"4)Eliminar Cliente": Elimina el cliente indicado desde las colecciones correspondientes.

A.3 Se debe generar un reporte en archivo txt/csv que considere mostrar datos de las 2 colecciones anidadas

Las funciones que se encargan de generar reportes se encuentran en la clase VideoClub y son las siguientes:

escribirArchivoDeudores(): Crea archivo que contiene el nombre de todos los clientes deudores.(Primer nivel de anidación)
escribirArchivoArriendosActuales(): Crea archivo que contiene información del usuario y el nombre de las películas que tiene arrendadas actualmente.(Segundo nivel de anidación)

Se pueden acceder desde menú administrador opción **6**.

A.5 Todas las funcionalidades pueden ser implementadas mediante consola.

Las funcionalidades más importantes para el desarrollo del proyecto se pueden acceder directamente mediante el uso del menú por consola.

A.6 Link repositorio github

https://github.com/luis922/Proyecto-POO-2S-2021.git

La rama principal del proyecto sobre la cual se trabaja es “Mismo-proyecto”

La rama correspondiente al. zip entregado es ***backup_ParteA.***

Opcionales

Implementada la búsqueda de películas y clientes, accesible desde menú administrador opción **3->1 y 3->2** y en menú cliente opción **1->2**

Extras

Se implementa un try-catch en metodo ***escribirArchivoDeudores()*** y ***escribirArchivoArriendosActuales*** ubicados en la clase VideoClub.