# Diagrama de clases

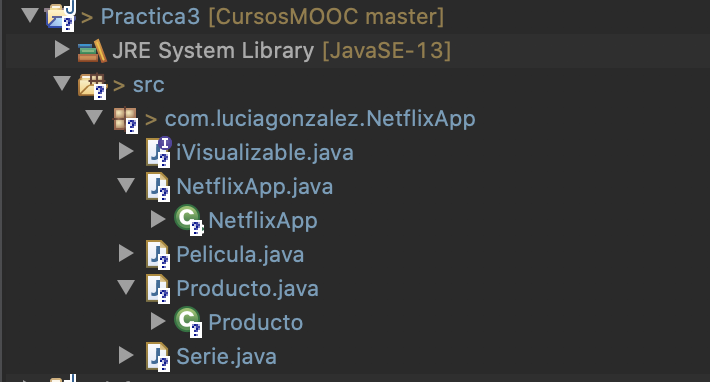


# Cambios respecto al ejercicio anterior

No hace falta implementar interfaz de esVisto() ni tiempoVisto(), ya que de esta forma es mucho más fácil obtener los datos directamente, tal como se podrá observar en el código del programa principal.

# Código implementado

La jerarquía de archivos es la siguiente:



El código de la clase principal es el siguiente:

**package** com.luciagonzalez.NetflixApp;

**import** **java.util.ArrayList**;

**import** **java.util.Collections**;

**import** **java.util.Comparator**;

**import** **com.luciagonzalez.NetflixApp.Serie**;

**import** **com.luciagonzalez.NetflixApp.Pelicula**;

**public** **class** **NetflixApp** {

**public** **static** **void** **main**(String[] args) {

// Creación de arrayList de objetos de tipo Serie y Pelicula

ArrayList <Serie> series = **new** ArrayList <Serie> ();

ArrayList <Pelicula> peliculas = **new** ArrayList <Pelicula> ();

// CREACIÓN DE SERIES Y PELICULAS: Este código se entiende que puede estar sujeto a errores, ya que los parámetros introducidos puede que no cumplan los requisitos para crear un nuevo registro correcto en la lista de objetos Series y Peliculas:

**try** {

// Series:

series.add(**new** Serie());

series.add(**new** Serie("Game of Thrones", "Ciencia Ficción"));

series.add(**new** Serie("Lost", "Ciencia Ficción"));

series.add(**new** Serie("Black Mirror","Charlie Brooker","Ciencia Ficción", **60.43**, **5**));

series.add(**new** Serie("Merlí", "Héctor Lozano", "Comedia", **45.9**, **3**));

// Peliculas

peliculas.add(**new** Pelicula());

peliculas.add(**new** Pelicula("Titanic","Drama"));

peliculas.add(**new** Pelicula("El diario de Noah", "Drama"));

peliculas.add(**new** Pelicula("Bandersnatch","Ciencia Ficción","Charlie Brooker", **185.3**, **2018**));

peliculas.add(**new** Pelicula("Memento","Suspense","Cristopher Nolan", **155.9**, **2000**));

}

// En caso de que haya algún error de tipo en los números de temporadas o duración

**catch** (NumberFormatException numberExcepcion) {

System.out.println("Error en el formato del número");

}

**catch** (Exception e) {

// Este tipo de forma de enseñar el error no es adecuada para mostrar al usuario, porque

// muestra la ruta entera (da información del programa y además es dificil de entender)

System.out.println("Ocurrió el siguiente error: ");

e.printStackTrace(); // Ocurre toda la lista de errores que han ocurrido

}

**finally** {

System.out.println("El total de peliculas introducidas es de " + peliculas.size());

System.out.println("Se han introducido las siguientes peliculas: \n" + peliculas);

System.out.println("El total de series introducidas es de: " + series.size());

System.out.println("Se han introducido las siguientes series: \n" + series);

}

// Eliminar peliculas y series de la lista.

// Marcarlas como vistas

**try** {

series.get(**1**).marcarVisto();

series.get(**4**).marcarVisto();

peliculas.get(**3**).marcarVisto();

peliculas.get(**2**).marcarVisto();

series.remove(**3**);

peliculas.remove(**7**); // Fuera de limites, se fuerza el error.

}

// En caso de que haya un exceso de números de la matriz, sale el error:

**catch** (IndexOutOfBoundsException arrayE) {

System.out.println();

System.out.println("Error: No se encuentra índice para proceder a la eliminación");

}

**catch** (Exception e) {

// Este tipo de forma de enseñar el error no es adecuada para mostrar al usuario, porque

// muestra la ruta entera (da información del programa y además es dificil de entender)

System.out.println("Ocurrió el siguiente error: ");

e.printStackTrace(); // Ocurre toda la lista de errores que han ocurrido

}

**finally** {

// Datos de series:

System.out.println();

System.out.println("El listado de series vistas es:");

// Imprimir nombre y duración de cada serie si tiene el atributo Vista = true

// \*\*\* Haciéndolo de esta forma nos ahorramos usar los métodos esVisto() y tiempoVisto()

series.forEach(Serie ->

{

**if** (Serie.isVisto() == **true**)

{

System.out.println(Serie.getTitulo() + " Duración: " + Serie.getDuracion());

}

});

// Datos de series

System.out.println();

System.out.println("El listado de peliculas vistas es:");

// Imprimir nombre y duración de cada serie si tiene el atributo Vista = true

peliculas.forEach(Pelicula ->

{

**if** (Pelicula.isVisto() == **true**)

{

System.out.println(Pelicula.getTitulo() + " Duración: " + Pelicula.getDuracion());

}

});

// Mostrar resultados máximos:

System.out.println();

Serie maxTemp = Collections.max(series, Comparator.comparing(s -> s.getnTemporadas()));

System.out.println("La serie con más temporadas es " + maxTemp.getTitulo() + " con " + maxTemp.getnTemporadas());

System.out.println();

Pelicula reciente = Collections.max(peliculas, Comparator.comparing(s -> s.getAño()));

System.out.println("La serie más reciente es " + reciente.getTitulo() + " del año " + reciente.getAño());

}

}

}

# Resultados obtenidos

Los resultados obtenidos en la consola son los siguientes:

