Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

**TC1030.328**

Object Oriented Programming

# **Integrative project**

Ernesto Miranda Solís

Delivery date: June 15th, 2022

Teacher:

Sergio Ruiz Loza

Tecnológico de Monterrey

Campus de Ciudad de México

Table of contents

[**Integrative project** 1](#_Toc106232890)

[Introduction 3](#_Toc106232891)

[UML Diagram 3](#_Toc106232892)

[Execution example 4](#_Toc106232893)

[Argumentation 11](#_Toc106232894)

[a) The proper class are identified 11](#_Toc106232895)

[b) Inheritance is implemented properly 11](#_Toc106232896)

[c) Access modifiers are implemented properly 12](#_Toc106232897)

[d) Method overwriting is implemented properly 13](#_Toc106232898)

[e) Polymorphism is implemented properly 13](#_Toc106232899)

[f) Abstract classes are implemented properly 13](#_Toc106232900)

[g) At least one operator is overloaded properly 13](#_Toc106232901)

[h) Pre-defined and user-defined exceptions are handled properly 14](#_Toc106232902)

[Cases that would prevent the program from working 14](#_Toc106232903)

[Conclusion 15](#_Toc106232904)

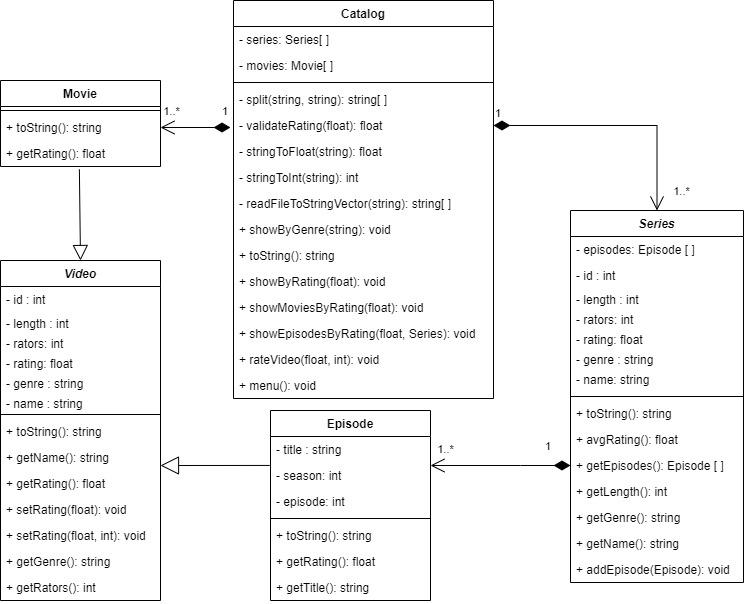
[References 15](#_Toc106232905)

# Introduction

This project has the scope of modelling a system that displays catalog of a contemporary streaming platform, such as Netflix, Amazon Prime Video, HBO and so on. The provider works with two main collections. A collection of movies and a collection of series. Each series has also a collection of episodes. Each video stored in the catalog has an ID, name, length and genre. Episodes have also a title and a season to which they belong. They have also a specific rating and a specific number of raters that vote to give that current rating. The rating average value goes from 1 to 5.

In this project the main concern is to show the collection of videos, the episodes of a specific series, and the movies, along with their rankings. The functionality to rank a video is also mandatory, and the user can choose to show a video according to either its ranking or its genre.

# UML Diagram



Five important classes were identified. The catalog class oversees unifying all the other ones. It handles the functionality of the program overall through the menu function. In this class are also allocated the collections of series and movies. It can read a file to obtain all the information of a catalog.

Moving forward the next important class might be the Video class. It is an abstract class from which Movie and Episode will import their attributes and methods. As their attributes are private, it provides the correct encapsulation to make this class safe. The Episode class implements also encapsulation by making its attributes private. The overridden functions from both classes are the “toString” method and the “getRating” method, therefore, episode also implements a “getTitle” function.

Finally, the Series class makes a composition of episodes and has some repeated attributes from the video class. However, a series is not a video, rather it is a collection of episodes with a different behavior as a Video.

# Execution example

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Argumentation

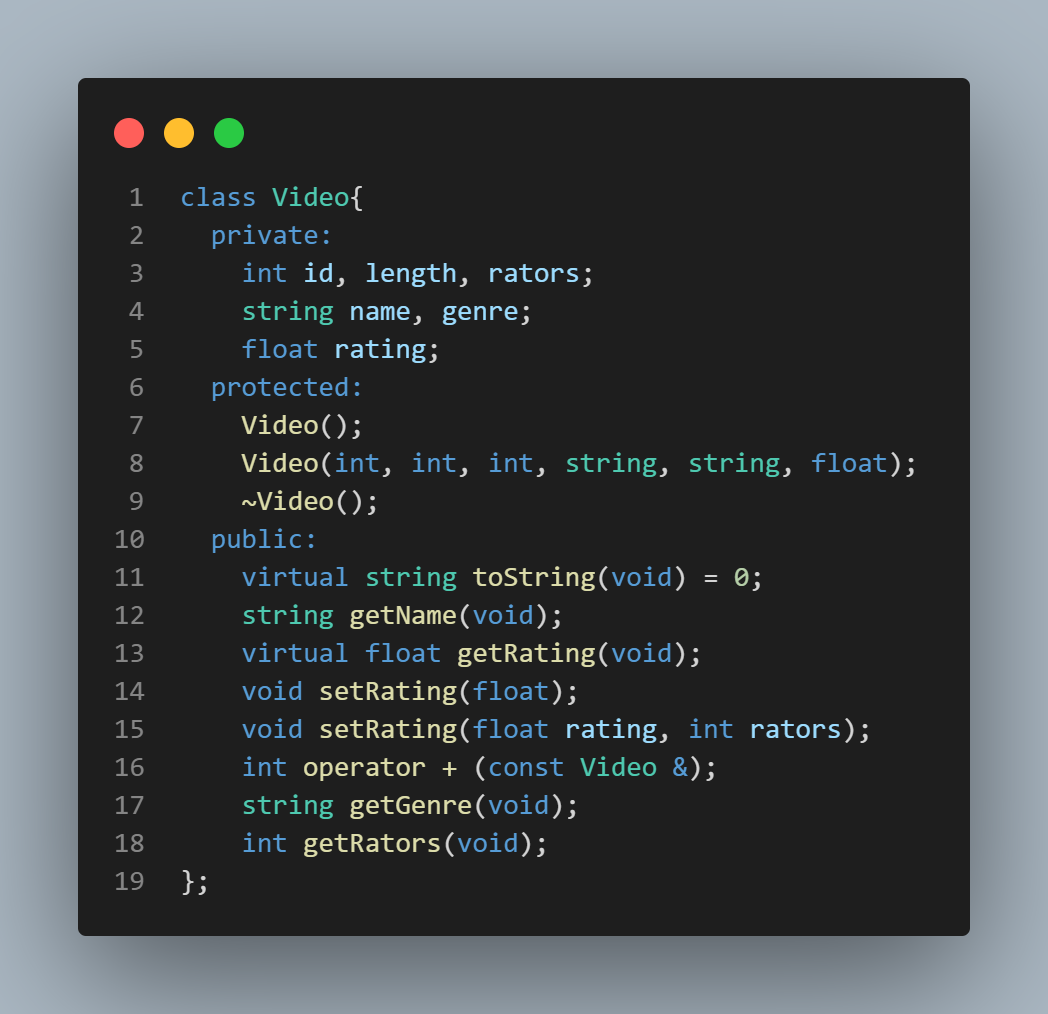
## The proper class are identified

There was one helping class that was implemented in the code but wasn’t expressed in the UML diagram because it can be declared as a struct. This class is the Comparator class, which helps to develop and implement one solution seen in Stack Overflow for finding objects.

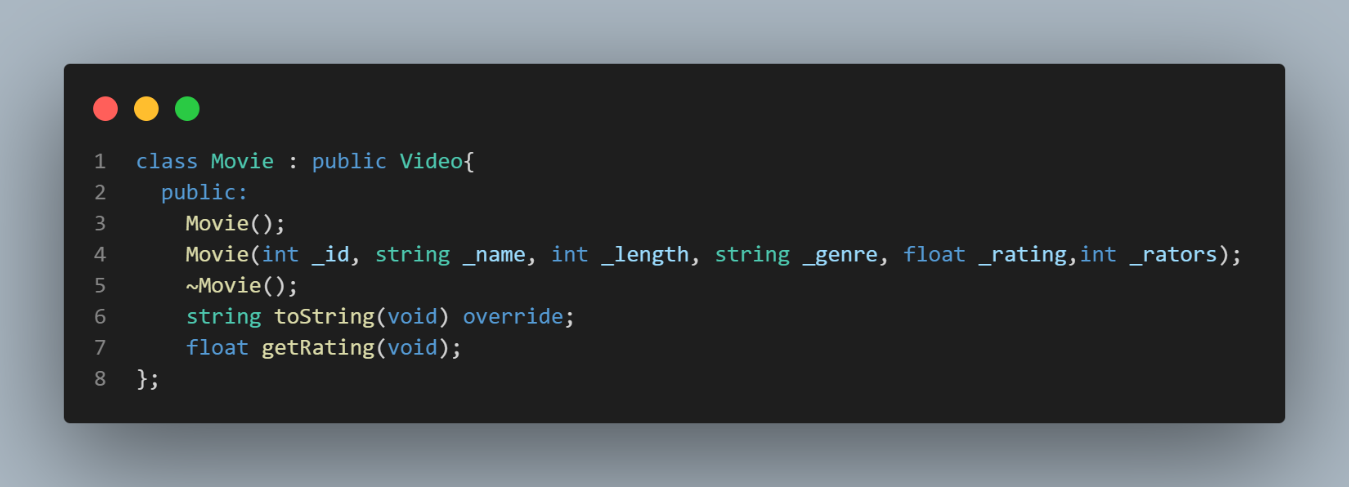
Other than that, the classes identified served not only a functional way, but also a logical implementation of inheritance and polymorphism based on the abstraction of the elements of a streaming catalog.

## Inheritance is implemented properly

As it was already discussed, the project serves as a logical abstraction. That’s why it is better to say that both episode and movie extend video rather than saying that series extend video, because series is not a type of video, rather it has multiple videos, which in this case are episodes. It helped to reuse code that otherwise will create redundancies. Although, not all the attributes from the series class were used in the functionality, they were needed according to the description of the problem.







## Access modifiers are implemented properly

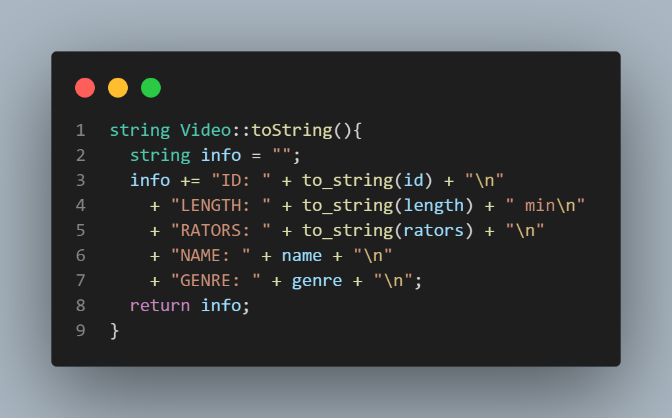
Throughout the code encapsulation played a huge roll in deciding which attributes were going to be public or private. Neither functions nor attributes were protected because they were implemented through inheritance in most cases when needed, and if it wasn’t needed, in order to access or modify an attribute, the getters and setters were implemented. The only protected elements from a class were the video constructors because they were only needed for the inherited classes.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

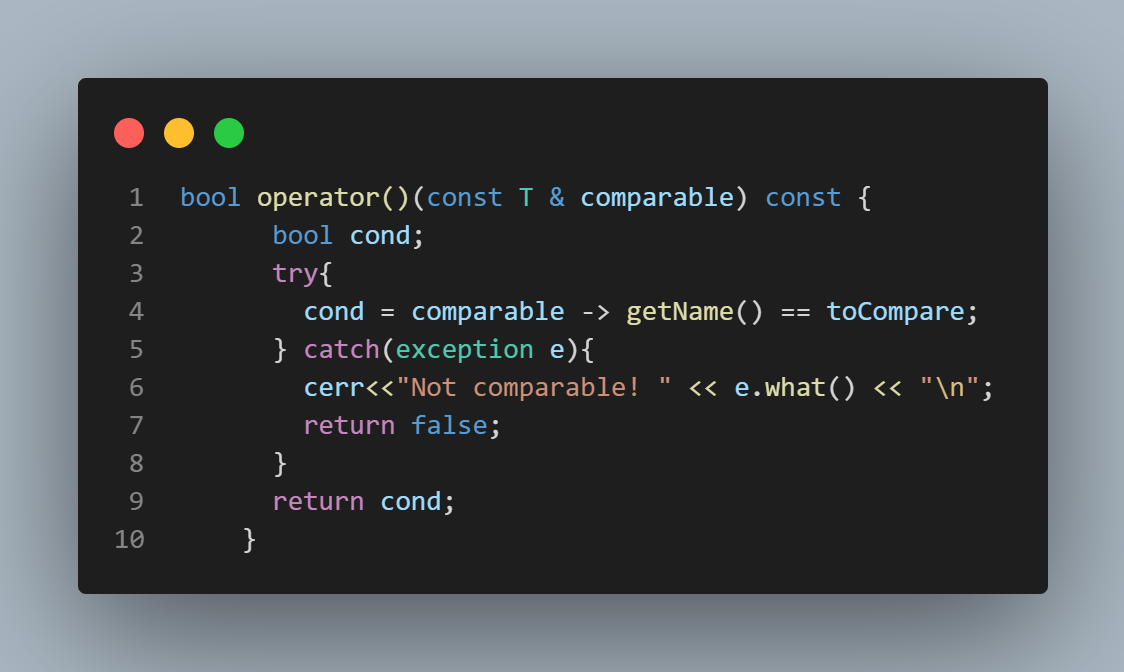
### Method overwriting is implemented properly

Because video was an abstract class, many of the methods were declared as virtual and although they might have already some implemented codes in the source file, most of them must be overridden/overwritten. Particularly the methods to string and get rating must be overwritten in the inherited classes.

## Polymorphism is implemented properly

Both movie and episode share the methods from video, which was key using the comparator because it used exclusively the method get name. Additionally, if it was needed, a vector of video pointers can be populated with instances of both movies and episodes.



## Abstract classes are implemented properly

As mentioned previously, video is a pure virtual class. This means that it is a well-defined abstract class implemented in both episode and movie classes.

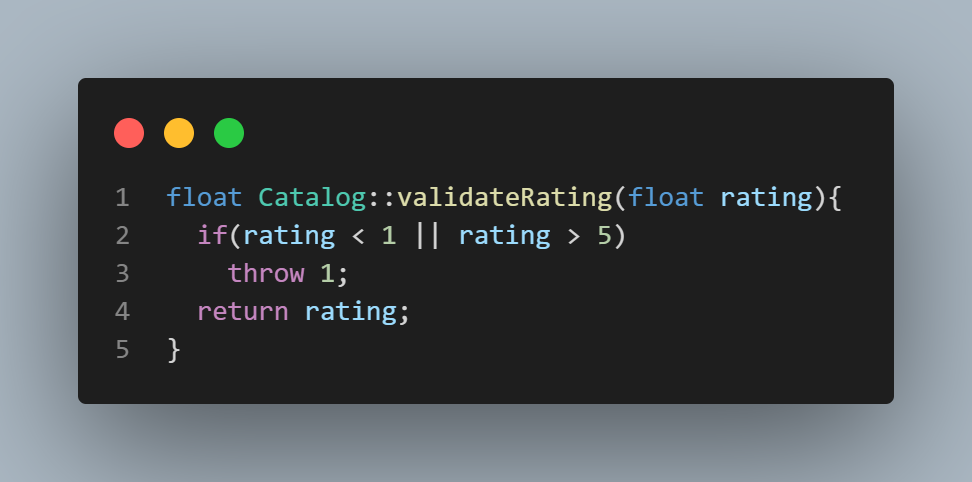
## At least one operator is overloaded properly

In the development of this project, two operators were overloaded. The first one was in the comparator class, while the operator + was overloaded in the video class which helped calculate the average rating from a series.



## Pre-defined and user-defined exceptions are handled properly

The exceptions handle some user introduced data, like the validation of the rating. However, another exception handling is seen in the comparator class when it checks if the classes are comparable.





# Cases that would prevent the program from working

This program doesn’t handle incorrect datatype for input streams. So, if a user introduces a string instead of an integer some errors might be produced. It also depends on the order of the columns on the csv file. That means that it doesn’t handle cases where the rows are changed.

# Conclusion

Throughout this project’s development many object-oriented design competencies were developed. It helped understand a little bit more the structure of different streaming catalogs relying on abstraction and code implementation. For further development it might be interesting to optimize the rating calculation, I’m particularly unsure if the solution implemented was mathematically optimized. Hence, documentation shall be added in the code and not only in an external pdf file, particularly in the source files for further development.

# References

IMDb. (n.d.). IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows. <https://www.imdb.com>

Sevilla, D. (2011, May 26). Searching for string in vector of pointers. Stack Overflow. <https://stackoverflow.com/questions/6145670/searching-for-string-in-vector-of-pointers>

cplusplus. (n.d.). Input/output with files - C++ Tutorials. <https://m.cplusplus.com/doc/tutorial/files/>

cplusplus, (n.d.). Reference. <https://cplusplus.com/reference/>