José Carlos Esquerra Rivas – A00827879

12/06/2020

Modelado de servicio de streaming  
En los últimos años, han proliferado los servicios de streaming de video bajo demanda por ejemplo Netflix, Disney, DC entre otros. Algunos de ellos se especializan por el volumen de videos que proporcionan a sus usuarios mientras que otros se han puesto el reto de mostrar solamente videos de su propia marca. Una versión limitada para apoyar a un futuro proveedor de este tipo de servicios es la que se describe a continuación.

Proyecto Integrador

Situación Problema – TC1030.1

Contenido

[Introducción 2](#_Toc42897505)

[Diagrama UML 3](#_Toc42897506)

[Ejemplos de Ejecución 4](#_Toc42897507)

[Identificación de errores 9](#_Toc42897508)

[Conclusión 9](#_Toc42897509)

# Introducción

Esta situación problema plantea un servicio de streaming, en específico uno de manejo de video, para apoyar a un futuro proveedor de este tipo de servicios. El servicio debe manejar dos tipos de videos: películas y series. Cada video tiene un ID, un nombre, una duración y un género.

Las series tienen episodios y los episodios tienen título, id y temporada propios.

El servicio debe ser capaz de permitir al usuario calificar cada video y mostrar los videos. Así como filtrarlos y mostrarlos por calificación, tipo o género.

A continuación, se presentan las bases de este proyecto mediante un diagrama UML de clases, capturas del programa y la argumentación detrás del uso de conocimientos de programación orientada a objetos para realizar el proyecto.

# Diagrama UML

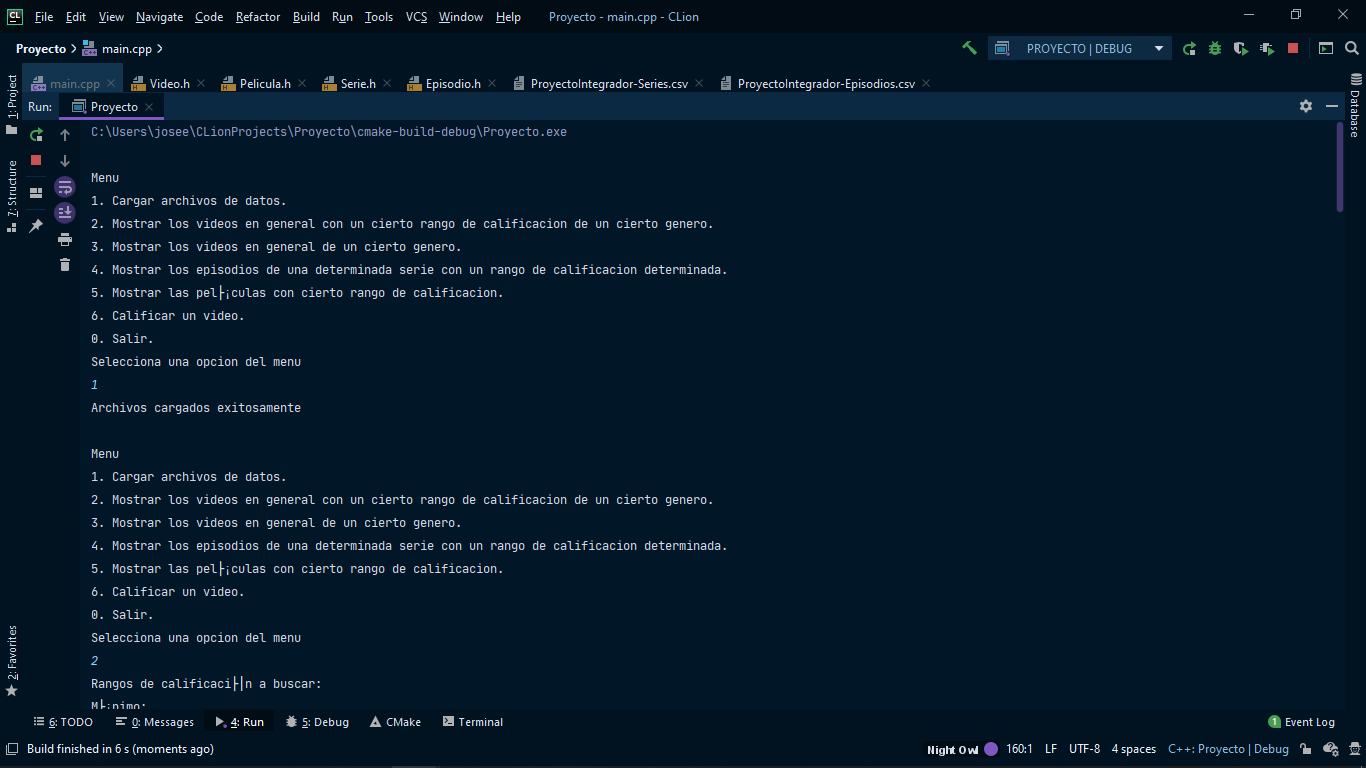
Se propuso la creación de 4 clases: Video, Pelicula, Serie y Episodio. Siendo Video una clase padre abstracta para Pelicula y Episodio. Serie siendo unida a Episodio por su inclusión y la creación de un atributo de un vector tipo Episodio en la clase Serie. Esto se muestra mas a profundidad en el diagrama UML a continuación.

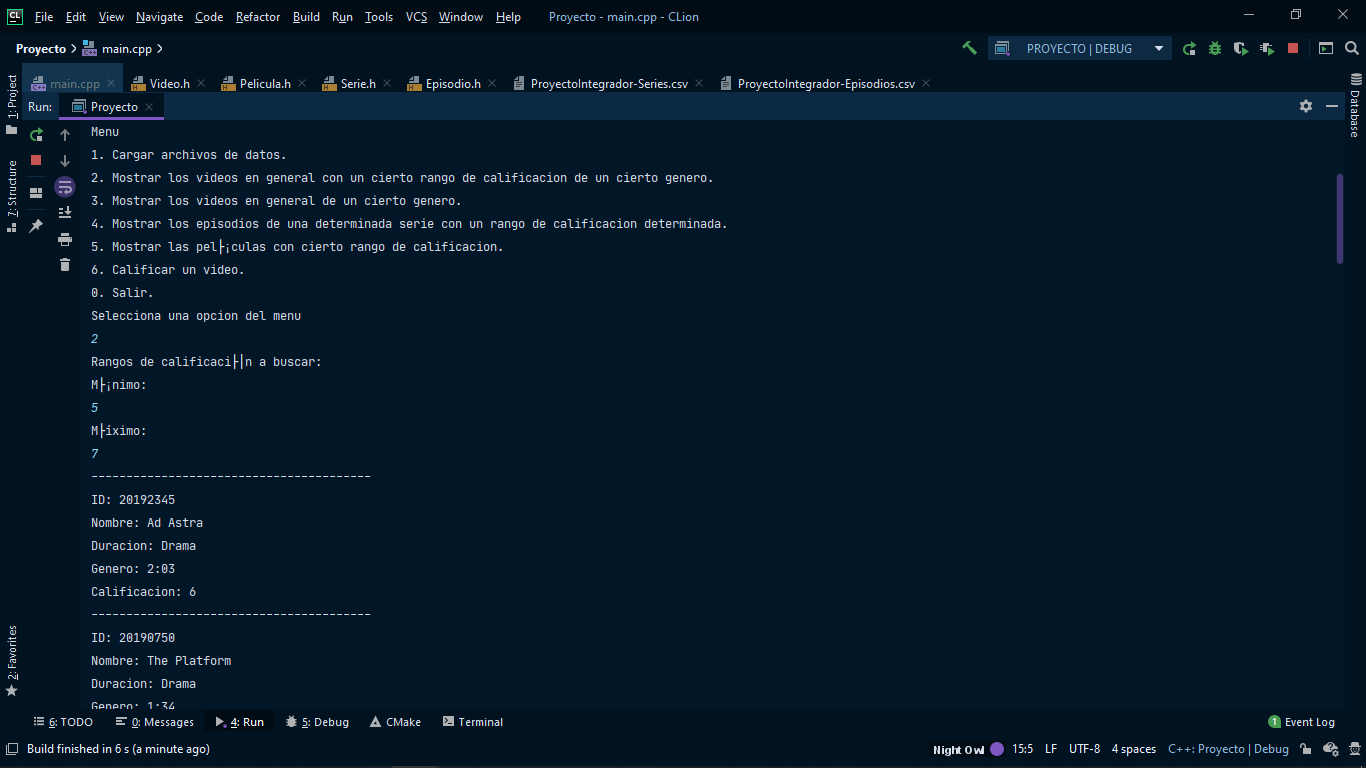
Captura de pantalla de un celular

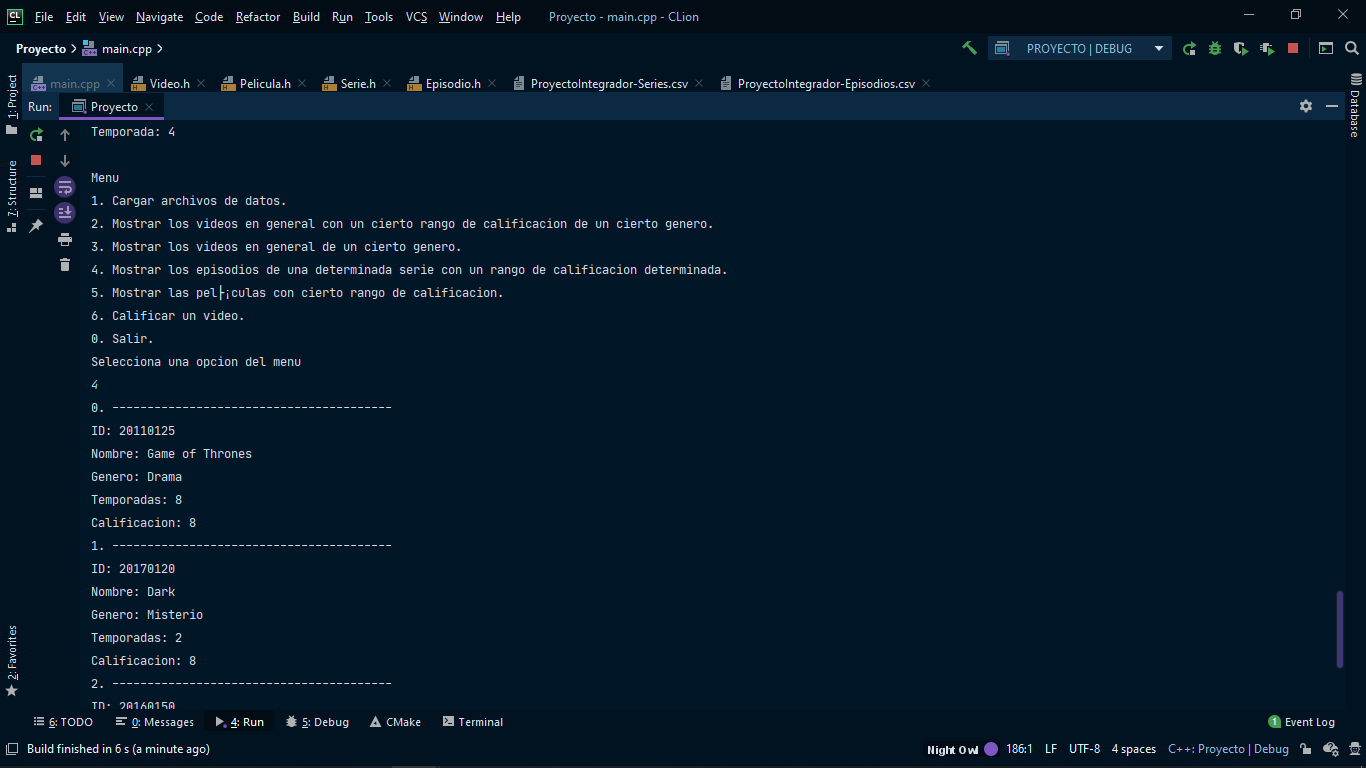
Descripción generada automáticamente

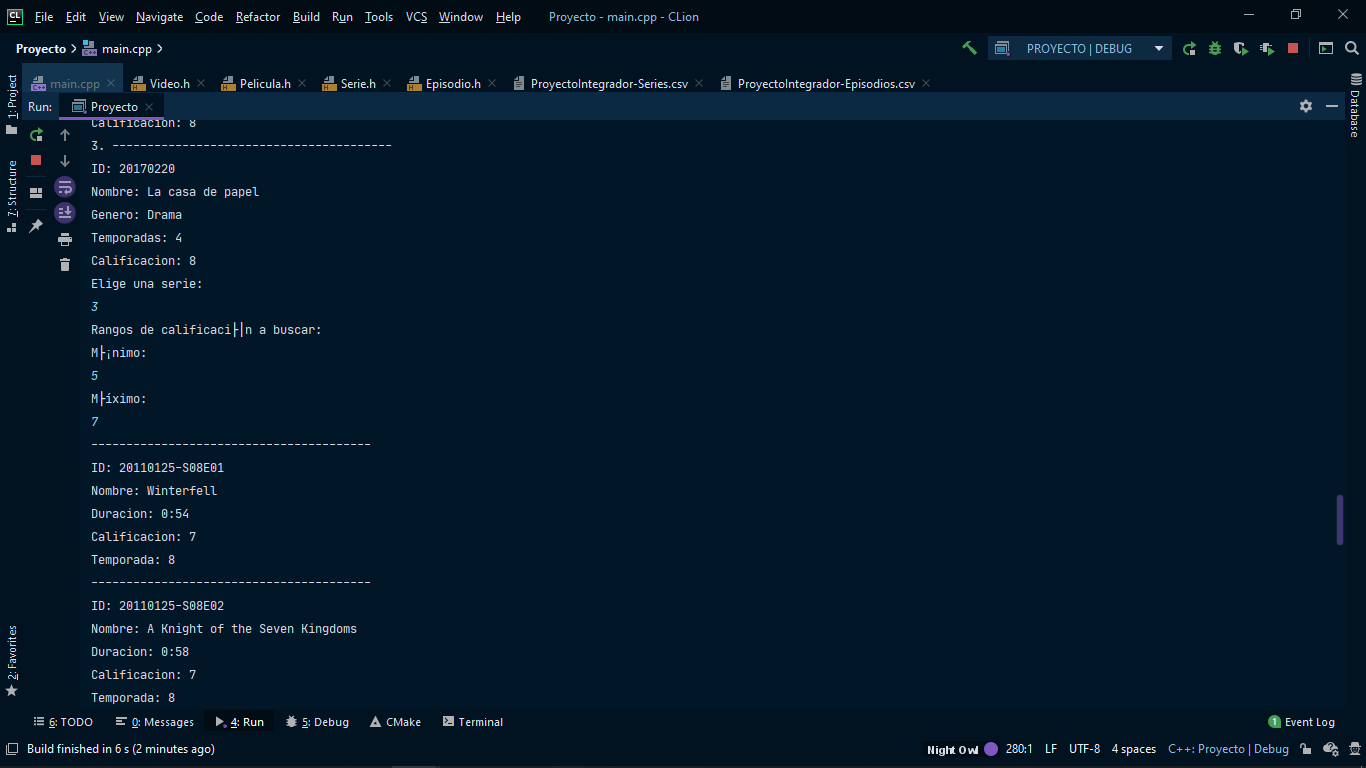
* El método mostrar() se cambió por operator <<(int cero), pero dadas cuestiones de tiempo no se pudo actualizar el diagrama UML, el resto permanece.

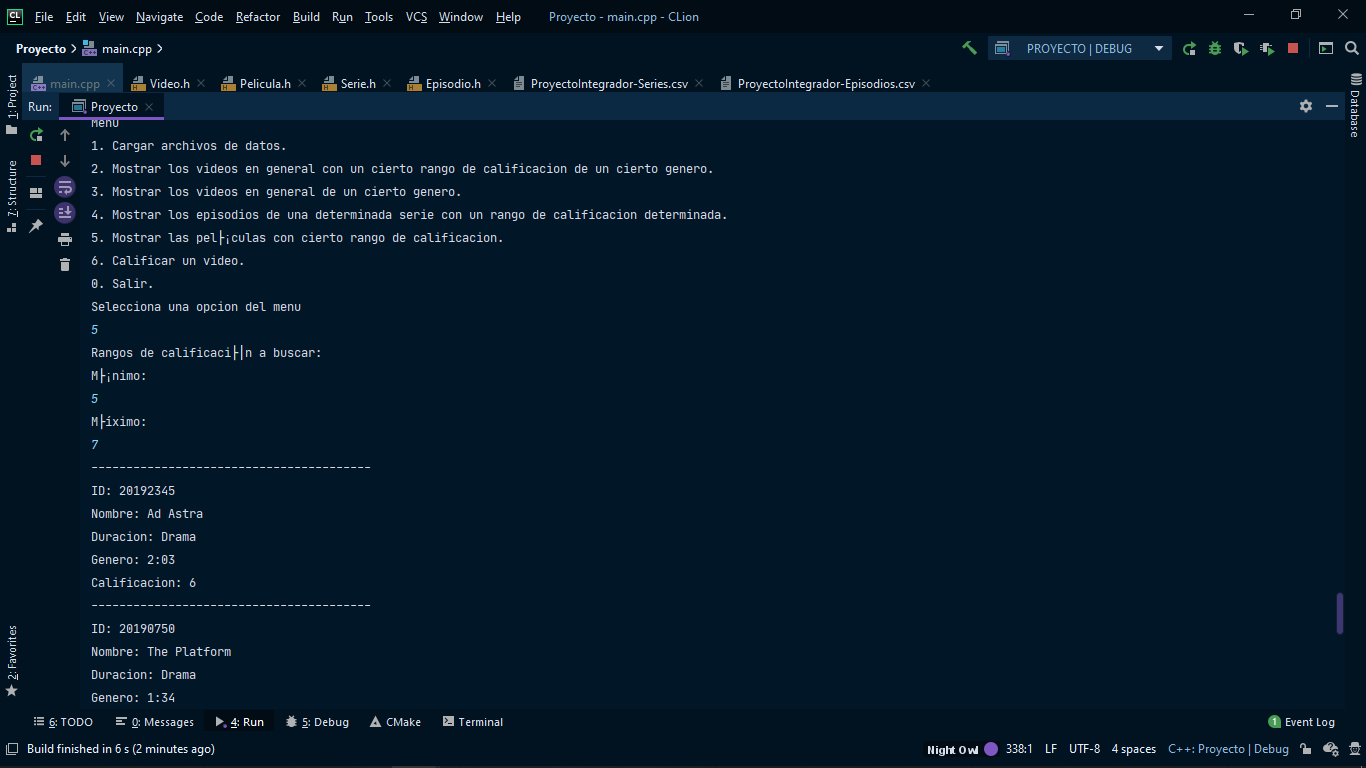
# Ejemplos de Ejecución

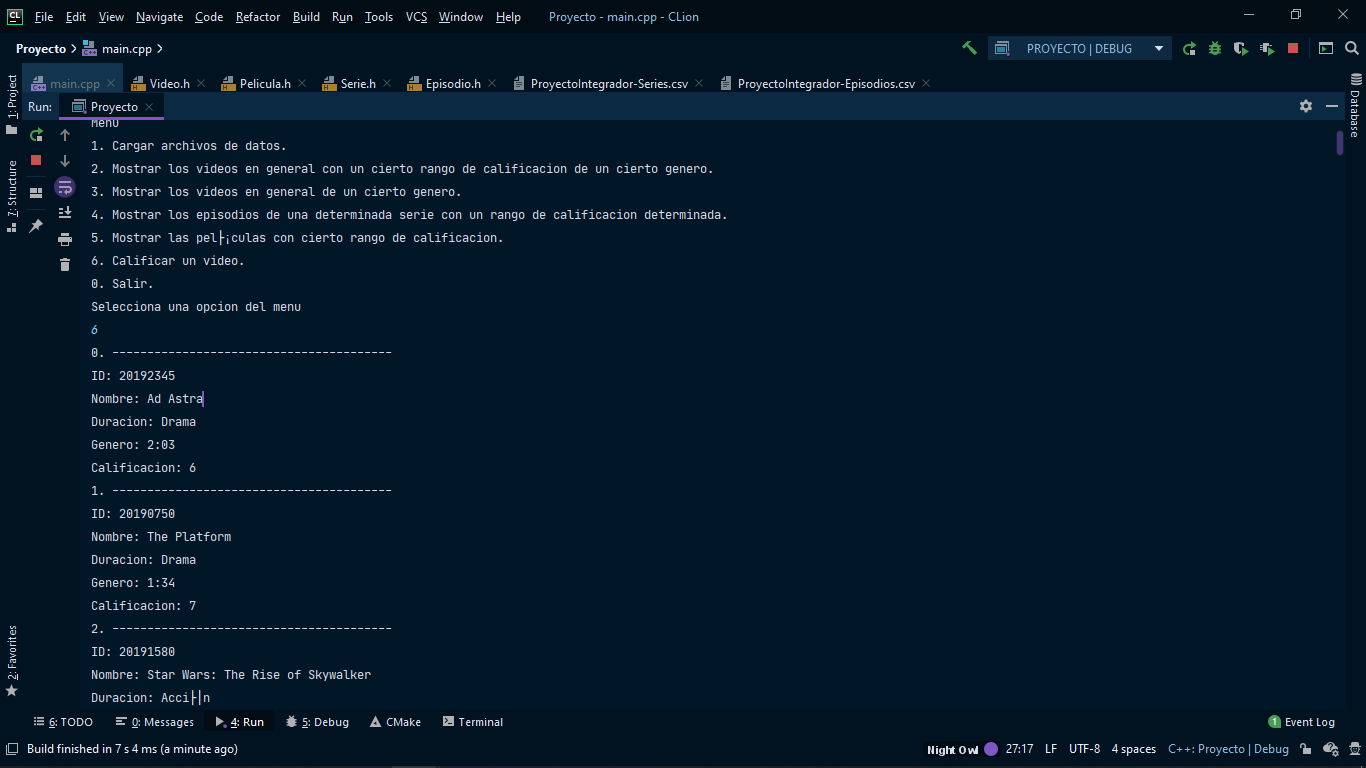


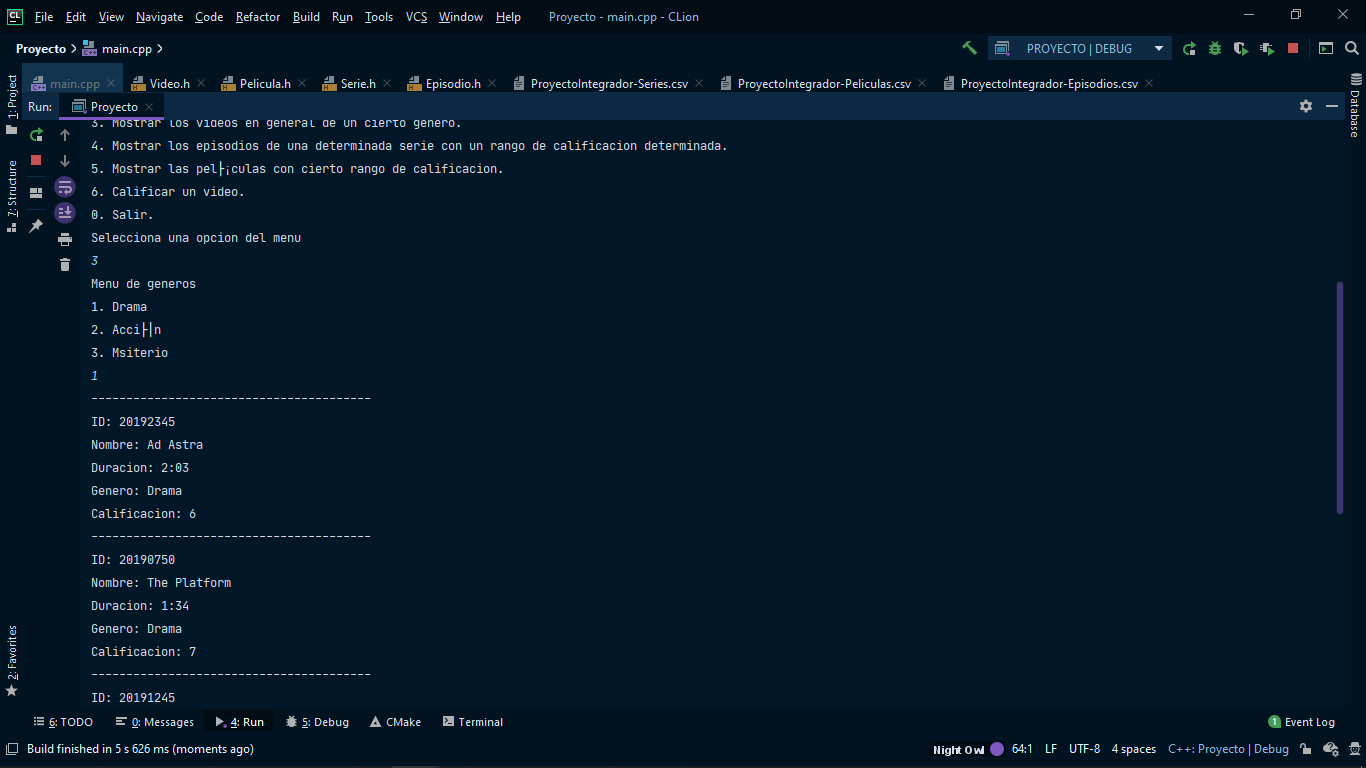






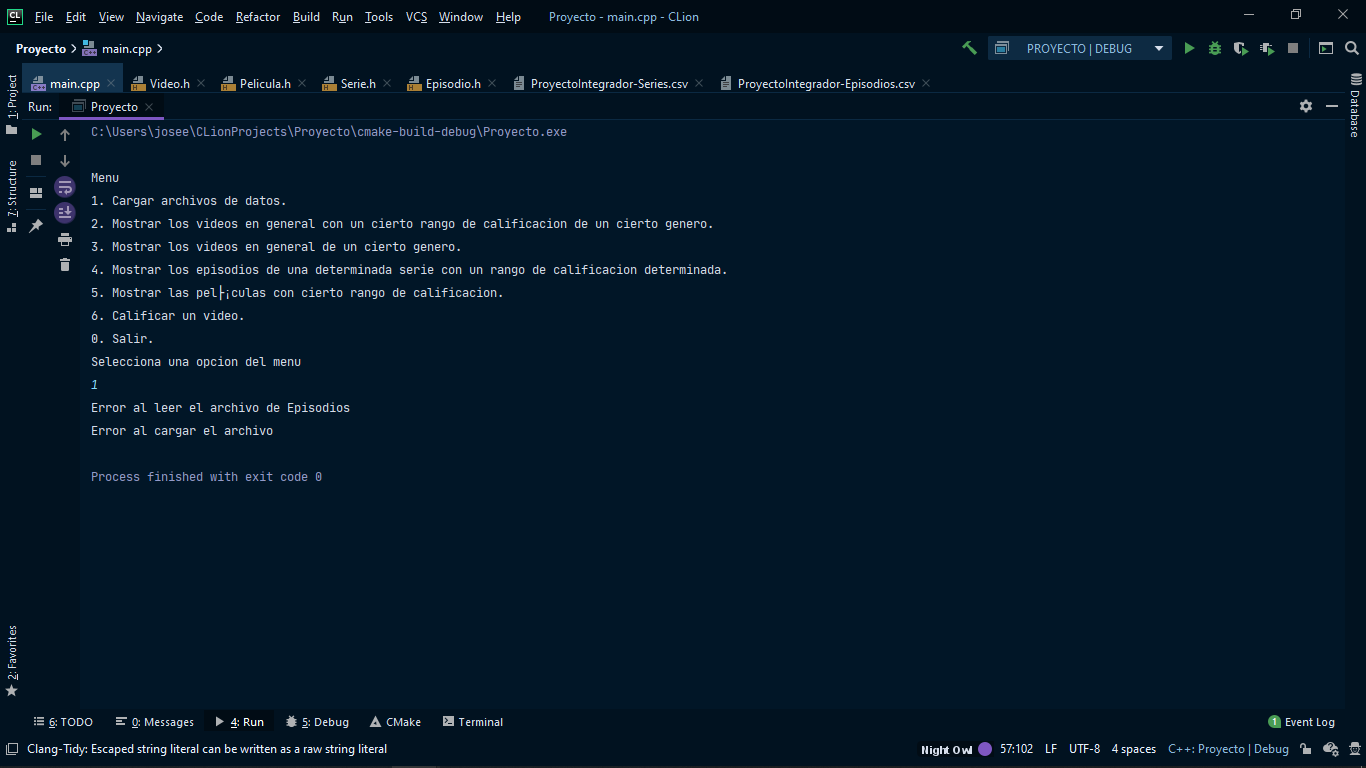




Argumentación

Se utilizó Herencia y Polimorfismo en las clases, tomando a Video como clase padre y a Episodio y Pelicula como hijas. Así como se unió a Serie a Episodio para manejarlas juntas. Se utilizó sobrecarga de operadores para mostrar los videos. Y se usó excepciones para comprobar que los archivos cargaran correctamente.

# Identificación de errores



Conclusión

En este reto se utilizaron nuevos conocimientos de Programación Orientada a Objetos como Herencia, Polimorfismo, Sobrecarga de Operadores y Excepciones. En este proyecto se utilizaron para unir las distintas clases sin fricción entre ellas y para mejorar el diseño del programa.

Se busco emplear todos los conocimientos adquiridos en la clase, aunque el código no sea el más eficiente, estoy consciente de mis errores y tengo los conocimientos para corregirlos. Este proyecto fue un buen reto para hacer el uso de los conocimientos de la clase y me resultó de gran aprendizaje.