Texto

Descripción generada automáticamente con confianza baja

**Situación problema modelado de servicio de streaming**

Pablo Heredia Sahagún a01637103

Programación orientada a objetos

Edgar Gerardo Salinas Gurrión

15 de junio de 2023

**Situación problema modelado de servicio de streaming**

**INDICE**

1. Introducción

1.1 Planteamiento del problema

1. Diseño del diagrama de clases UML

2.1 Argumentación del diseño

1. Ejemplo de ejecución

3.1 Explicación de la ejecución del codigo

3.2 Capturas de pantalla del ejemplo de ejecución

1. Argumentación de la elección de la solución y descarte de otras opciones

4.1 Explicación de la solución elegida

4.2 Comparativa con otras posibles soluciones descartadas

1. Identificación de casos que harían que el proyecto deje de funcionar

5.1 Casos que podrían afectar al funcionamiento del proyecto

1. Conclusión personal
2. Referencias consultadas

1.

En los últimos años, hemos sido testigos de un aumento significativo en la popularidad de los servicios de streaming de video bajo demanda, como Netflix, Disney y DC, entre otros. Estos servicios ofrecen una amplia variedad de contenido, desde películas hasta series, cada uno con su propio catálogo exclusivo. En este proyecto, nos centraremos en desarrollar una versión limitada de un servicio de streaming que nos permitirá trabajar con dos tipos de videos: películas y capítulos (o episodios).

Cada video en nuestro sistema estará identificado por un ID único y contendrá información como el nombre, la duración y el género al que pertenece, como drama, acción o misterio. Los capítulos tendrán el nombre de su serie además de su número de capitulo.

Una característica importante que queremos implementar en nuestro sistema es la capacidad de calcular y mostrar la calificación promedio que ha recibido cada video. Esta calificación será otorgada por los usuarios y se representará en una escala del 1 al 5, donde 5 es la mejor calificación posible.

1.1

Para lograr una solución eficiente y bien estructurada, haremos uso de los conceptos fundamentales de la Programación Orientada a Objetos, como la herencia, el polimorfismo y la sobrecarga de operadores. Mediante la herencia, podremos establecer una jerarquía de clases que nos permita reutilizar comportamientos comunes entre las películas y los capítulos. Asimismo, el polimorfismo nos brindará la flexibilidad necesaria para tratar tanto a películas como a capítulos como objetos genéricos de video, permitiéndonos trabajar con ellos de manera uniforme

Además, como parte de este proyecto, nos planteamos el desafío de construir una aplicación que pueda generar reportes sobre los diferentes tipos de videos. Esto incluye la capacidad de filtrar y mostrar películas o capítulos de un determinado género, así como películas o capítulos con su respectiva calificación. Esta información nos será otorgada a través de un archivo txt con un formato predeterminado, que tendremos que leer y usar en nuestra ejecución para poder interactuar con la información de los videos.

2

2.1

Como se puede ver en el diagrama uml, en mi implementación decidí utilizar la clase video como padre de película y capitulo. Básicamente aquí inicializamos los atributos importantes como el id, el nombre de video, la duración y el género. Además, tenemos el atributo calificación, el cual viene acompañado de sumacalis y numcalis, y hablaremos de su implementación más adelante. Tenemos implementado dos constructores, uno por default y uno que recibe los atributos id, nombre, duración y género, ya que la calificación la inicializa en 0. El método calificar toma como parámetro un entero, y lo suma a la variable sumacalis asi como suma +1 a numcalis, y despues llama el método calcCali para poder calcular la calificación en base al promedio de calificaciones previamente ingresadas por el usuario (divide sumacalis entre numcalis). El método imprimir es de tipo virtual ya que se instancia en las clases Película y Capitulo, esto ya que imprime toda la información de dicho video, y al capitulo tener dos atributos extra (episodio y serie), imprimirá cosas un poco diferentes. Los métodos imprimirCalif e imprimirGenero se usan dentro del menú ciclado para imprimir ciertas características de los videos. CalcSC fue mi solución para poder incluir “SC” si el video no tenía calificación, evitando reutilizar el codigo en todos los métodos que impriman alguna información del video. Finalmente, implementamos los getters que fui necesitando para poder acceder a los atributos.

3

3.1

Para poder interactuar con el servicio, utilizamos un menú ciclado con las siguientes opciones:

1. Mostrar todo el catálogo con calificaciones.

2. Calificar un video

3. Mostrar películas o capítulos con una calificación mínima determinada

4. Mostrar películas o capítulos de un cierto género.

9. Salir

La primera opción muestra todos los videos que tenemos guardados y su respectiva información. SI no tiene una calificación se muestra SC. Además, aquí es donde usamos el polimorfismo en e método imprimir ya que depende de qué tipo de video sea. Para calificar un video usamos el método calificar, previamente obteniendo el id del video que se quiere calificar asi como la calificación misma. Para la opción 3 y 4 se hace básicamente lo mismo, lo cual es desplegar un submenú con la opción de ver solo películas, capítulos o ambos, y posterior a eso preguntar por una calificación mínima o un género deseado. Finalmente le damos la opción al usuario de salir del programa.

3.2

Capturas de pantalla

4

4.1

En general, decidí implementar mi codigo de esta manera por que traté de aplicar la mayoría de los conceptos vistos en clase. Para poder guardar los videos, utilice un vector de punteros de tipo video, el cual por concepto de herencia puede almacenar Películas y Capítulos. En el caso de guardar y calcular la calificación de un video, opte por sencillamente tener una variable que guarde la suma de todas las calificaciones ingresadas a ese video por el usuario, y un contador de cuantas calificaciones se han ingresado, para eventualmente sacar el promedio. La herencia del método calSC, el cual verifica si el video tiene alguna calificacion, fue muy importante para la lógica del codigo asi como el no repetirlo.

4.2

Un ejemplo de soluciones que había pensado, pero finalmente no implemente fue para obtener la calificación de un video. Había pensado en guardar las calificaciones ingresadas por el usuario en un vector y sacar el promedio de ese vector, pero aparte que era un poco mas tedioso para largos volúmenes de datos como e un servicio de streaming gastaría muchos recursos. Otra puede ser el guardar todos los videos en un mismo vector, y no hacer dos vectores separados, uno para películas y otro para episodios. Esto ya que además de usar punteros y conceptos más avanzados, tenía algunas desventajas a la hora de recorren el vector en ciertas partes del menú, como en la 1er opción, además que me impediría usar la sobrecarga del cout de video, cosa que si me permite el vector de punteros de video.

5

Aunque traté todos los casos de prueba que se me ocurrieron para poder tronar el programa, creo haber validado los inputs para que el usuario no pueda ingresar algo que truene el programa, ya que hice verificación de los cin y el tipo esperado, además de validaciones de rango de las entradas Enel caso del menú y la calificación. Creo que donde si pudiera tener más errores es al leer y guardar los datos del archivo txt, ya que es un tema que no vimos y una coma o espacio en un lugar no formateado rompería el codigo, ya que ahí si no tiene mucho manejo de errores mas que checar si se pudo abrir bien el archivo.

6

Este proyecto me ayudo para reforzar los conocimientos que adquirí durante mi clase, además de poder aplicarlos a una situación un poco mas apegada a la realidad y un producto que usamos continuamente. El poder usar y entender conceptos de herencia y polimorfismo me da a entender lo útil del paradigma orientado a objetos, dándome creatividad para aplicarlos a otros casos de uso. En general, creo que fue un buen acercamiento a la aplicación y uso real de un programa que use POO para su correcto y eficaz funcionamiento.

7

Referencias

una. (2017, March 15). *¿Cómo leer una línea en especifico de un archivo txt?* Stack Overflow En Español. https://es.stackoverflow.com/questions/55617/c%C3%B3mo-leer-una-l%C3%ADnea-en-especifico-de-un-archivo-txt

‌

*getline (string) en C++ – Barcelona Geeks*. (2022, July 5). Barcelonageeks.com. https://barcelonageeks.com/getline-string-en-c/

‌

*how to use cin.ignore - C++ Forum*. (2017). Cplusplus.com. https://cplusplus.com/forum/beginner/217812/

‌