

Programación orientada a objetos

Situación problema

Instituto Tecnológico y de Estudios Superiores de Monterrey



Grupo 33

Jared Alberto Flores Espinosa	A01655569
Elí Salomón Martínez Hernández	A01653876
Daniel I. Núñez López	A01654137

Profesor:

David Alejandro Escárcega Centeno

Junio 2021

ÍNDICE

Introducción	3
---------------------------	----------

Diagrama de clases	3
---------------------------------	----------

Ejemplo de ejecución	4
-----------------------------------	----------

Ejemplo de ejecución	5
-----------------------------------	----------

Argumentación de las partes del proyecto	5
---	----------

Identificación de clases límite	5
--	----------

Conclusión	5
-------------------------	----------

Referencias.....	6
-------------------------	----------

Introducción

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. Es por eso que nosotros haremos un recomendador de streaming que se encargará de buscar en la base de datos de las tres plataformas más populares, para que el usuario pueda saber en cuál de ellas encontrar el contenido de su preferencia.

Diagrama de clases

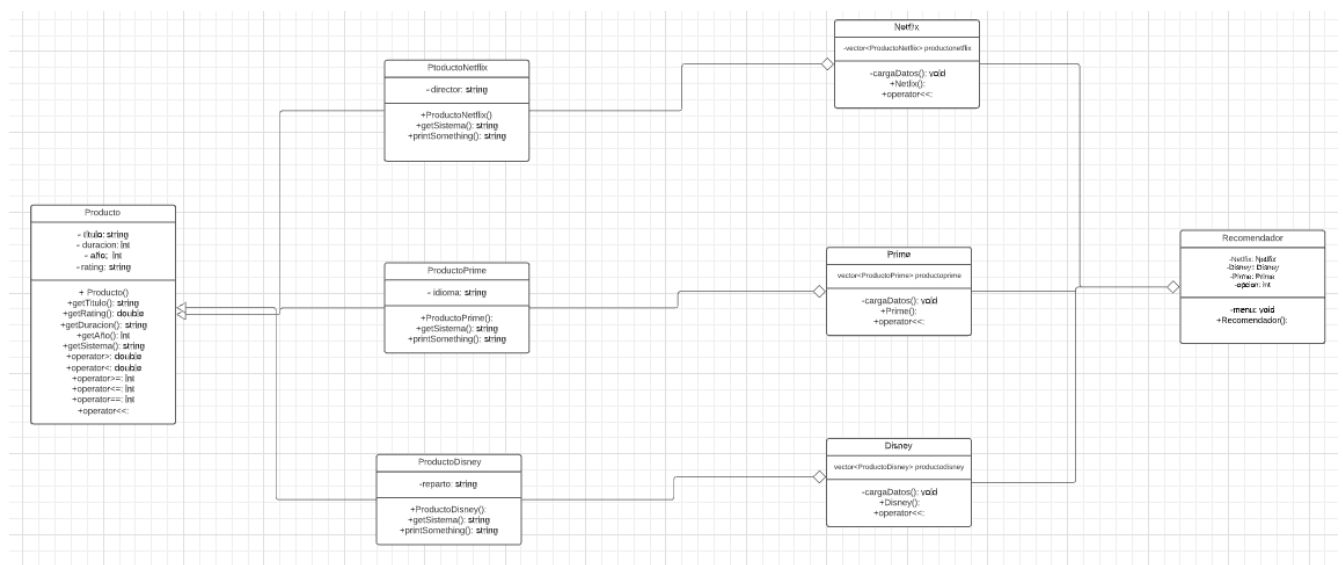
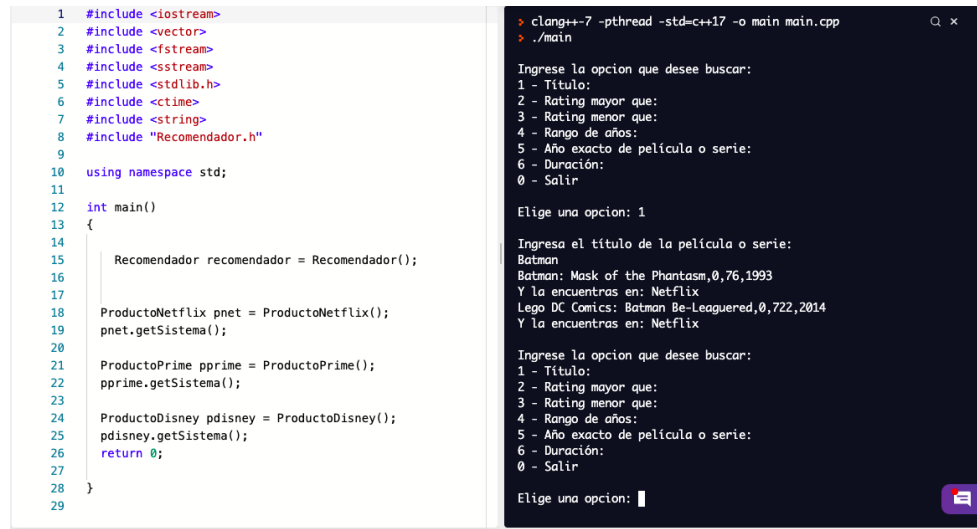


Diagrama 1. UML final https://lucid.app/lucidchart/d9f640bb-a83c-42d1-bb25-0df14269e12b/edit?shared=true&page=0_0#

El diagrama de clases elaborado representa los métodos y atributos utilizados en cada clase en el programa. Además, se expresa que las clases **ProductoNetflix**, **ProductoPrime** y **ProductoDisney** heredan de la clase **Producto**, es decir, son subclases de la clase madre **Productos**. A su vez, estas se relacionan por agregación con la clase de cada plataforma, donde se carga su base de datos correspondiente. Finalmente se tiene la clase del

recomendador que se encarga de desplegar el menú y hacer la búsqueda dentro de la base de datos y por tipo de atributo elegido en el menú.

Ejemplo de ejecución



The image shows a C++ program being compiled and executed. The source code on the left includes headers for iostream, vector, fstream, sstream, stdlib.h, ctime, and string, and defines a Recommendar class. The main function creates instances of ProductoNetflix, ProductoPrime, and ProductoDisney, and calls their getSistema() methods. The terminal output on the right shows the compilation command, the program's menu, and the user selecting option 1.

```
1 #include <iostream>
2 #include <vector>
3 #include <fstream>
4 #include <sstream>
5 #include <stdlib.h>
6 #include <ctime>
7 #include <string>
8 #include "Recomendador.h"
9
10 using namespace std;
11
12 int main()
13 {
14     Recomendador recomendador = Recomendador();
15
16     ProductoNetflix pnet = ProductoNetflix();
17     pnet.getSistema();
18
19     ProductoPrime pprime = ProductoPrime();
20     pprime.getSistema();
21
22     ProductoDisney pdisney = ProductoDisney();
23     pdisney.getSistema();
24     return 0;
25 }
26
27
28
29
```

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main

Ingrese la opcion que desee buscar:
1 - Título:
2 - Rating mayor que:
3 - Rating menor que:
4 - Rango de años:
5 - Año exacto de película o serie:
6 - Duración:
0 - Salir

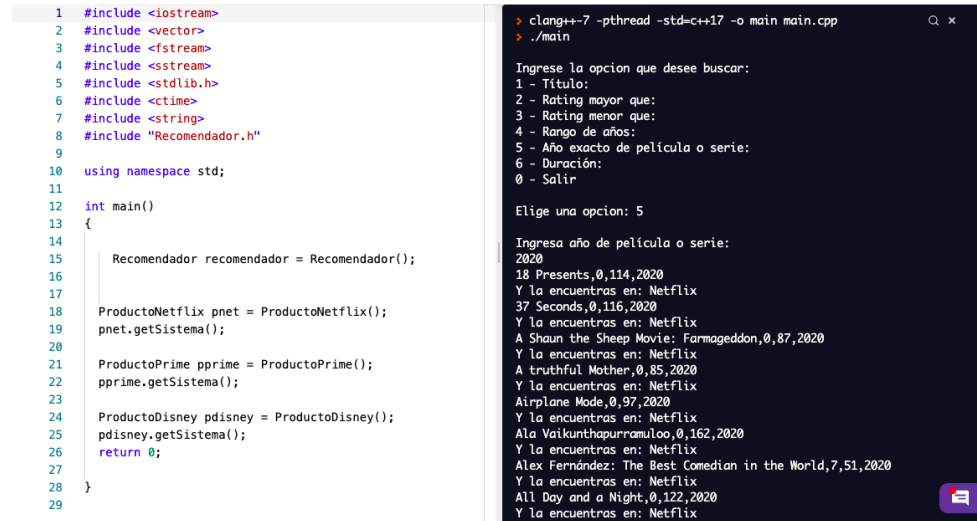
Elige una opcion: 1

Ingresa el título de la película o serie:
Batman
Batman: Mask of the Phantasm,0,76,1993
Y la encuentras en: Netflix
Lego DC Comics: Batman Be-Leaguered,0,722,2014
Y la encuentras en: Netflix

Ingrese la opcion que desee buscar:
1 - Título:
2 - Rating mayor que:
3 - Rating menor que:
4 - Rango de años:
5 - Año exacto de película o serie:
6 - Duración:
0 - Salir

Elige una opcion: 1
```

Imagen 1. Evidencia de funcionamiento del programa



The image shows the same C++ program being executed again. The source code is identical to the first image. The terminal output shows the user selecting option 5, and the program displaying a list of movies and their details.

```
1 #include <iostream>
2 #include <vector>
3 #include <fstream>
4 #include <sstream>
5 #include <stdlib.h>
6 #include <ctime>
7 #include <string>
8 #include "Recomendador.h"
9
10 using namespace std;
11
12 int main()
13 {
14     Recomendador recomendador = Recomendador();
15
16     ProductoNetflix pnet = ProductoNetflix();
17     pnet.getSistema();
18
19     ProductoPrime pprime = ProductoPrime();
20     pprime.getSistema();
21
22     ProductoDisney pdisney = ProductoDisney();
23     pdisney.getSistema();
24     return 0;
25 }
26
27
28
29
```

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main

Ingrese la opcion que desee buscar:
1 - Título:
2 - Rating mayor que:
3 - Rating menor que:
4 - Rango de años:
5 - Año exacto de película o serie:
6 - Duración:
0 - Salir

Elige una opcion: 5

Ingresa año de película o serie:
2020
18 Presents,0,114,2020
Y la encuentras en: Netflix
37 Seconds,0,116,2020
Y la encuentras en: Netflix
A Shaun the Sheep Movie: Farmageddon,0,87,2020
Y la encuentras en: Netflix
A truthful Mother,0,85,2020
Y la encuentras en: Netflix
Airplane Mode,0,97,2020
Y la encuentras en: Netflix
Ala Vaikunthapuramuloo,0,162,2020
Y la encuentras en: Netflix
Alex Fernández: The Best Comedian in the World,7,51,2020
Y la encuentras en: Netflix
All Day and a Night,0,122,2020
Y la encuentras en: Netflix
```

Imagen 2. Evidencia de funcionamiento del programa

```
1 #include <iostream>
2 #include <vector>
3 #include <fstream>
4 #include <sstream>
5 #include <stdlib.h>
6 #include <ctime>
7 #include <string>
8 #include "Recomendador.h"
9
10 using namespace std;
11
12 int main()
13 {
14     Recomendador recomendador = Recomendador();
15
16     ProductoNetflix pnet = ProductoNetflix();
17     pnet.getSistema();
18
19     ProductoPrime pprime = ProductoPrime();
20     pprime.getSistema();
21
22     ProductoDisney pdisney = ProductoDisney();
23     pdisney.getSistema();
24     return 0;
25 }
```

Ingresa año de película o serie:
4

Ingresa la opción que desee buscar:
1 - Título:
2 - Rating mayor que:
3 - Rating menor que:
4 - Rango de años:
5 - Año exacto de película o serie:
6 - Duración:
0 - Salir

Elige una opción: 6

Ingresa duración de película o serie:
125

#Selfie,0,125,2014
Y la encuentras en: Netflix
A Most Violent Year,0,125,2014
Y la encuentras en: Netflix
Air Force One,0,125,1997
Y la encuentras en: Netflix
Amar's Hands,0,125,2011
Y la encuentras en: Netflix
Article 15,0,125,2019
Y la encuentras en: Netflix
Cake,0,125,2018
Y la encuentras en: Netflix
Casino Tycoon,0,125,1992
Y la encuentras en: Netflix
Finally Found Someone,0,125,2017
Y la encuentras en: Netflix

Imagen 3. Evidencia de funcionamiento

Argumentación de las partes del proyecto

a) Se identifican de manera correcta las clases a utilizar

Para desarrollar nuestro Recomendador de películas y series identificamos 8 clases, en la primera clase se contemplan los atributos que tienen en común las 3 bases de datos. Y de esta forma las 3 clases hijas puedan heredar los atributos de la primera clase para crear objetos y añadirlos a las clases principales de cada sistema de streaming. Las clases hijas contienen los atributos heredados y un atributo especial propio de cada sistema de streaming. Posteriormente se tienen las clases de cada servicio (Netflix, Disney y Prime) en donde se cargan las bases de datos con las películas y series. Así como se crearán las funciones de búsqueda y los vectores para almacenar los contenidos de cada una. Finalmente se tiene la clase Recomendador donde se carga el menú con los métodos de búsqueda y arrojará los resultados de la búsqueda del usuario. Junto con excepciones y funciones virtuales.

b) Se emplea de manera correcta el concepto de Herencia

Para poder tener algo congruente con nuestras clases, tuvimos que utilizar los conceptos de herencia, que son: Agregación (cuando dos clases se relacionan pero pueden

existir por sí mismas, es decir, si una desaparece, la otra no) y Composición (cuando dos clases se relacionan pero si una no existe, la otra tampoco). En nuestro proyecto utilizamos la herencia de la en la clase producto hacia las demás clases producto, y entre las clases producto, las clases de los servicios y el recomendador, utilizamos agregación.

c) Se emplea de manera correcta los modificadores de acceso

Para el caso de los atributos de cada clase se decidió hacerlos privados con el fin de seguir una buena práctica de código. Sin embargo los métodos y las funciones virtuales así como la sobrecarga de operadores se realizó con el acceso público al igual que la herencia de clases, esto porque otras clases pueden acceder a los métodos si son públicos, a diferencia de si fueran privados, ya que a esos solo los puede acceder esa clase, para acceder desde otras clases, fue necesario usar getters.

d) Se emplea de manera correcta la sobrecarga y sobreescritura de métodos

Para la sobrecarga y sobreescritura de métodos, utilizamos los conceptos de polimorfismo, poral o primero, sobrecargamos los operadores de mayor que, menor que, igual y mayor que, igual y menor que, y igual igual, esto se implementó para poder realizar búsquedas de acuerdo al rating ingresado por el usuario o para comparar años, y mostrar los productos que cumplieran con lo mencionado. En cuanto a la sobreescritura, utilizamos funciones virtuales y parcialmente virtuales, lo primero para desplegar en qué plataforma de streaming se encontraba el producto y lo segundo para mostrar los atributos específicos de cada servicio de streaming, pero, ya que utilizamos una función que sobrescribe el título, un atributo del producto, como tienen prioridad la del producto, no se imprime la sobrecarga, sin embargo, está implementado correctamente.

Igualmente, sobrecargamos el ostream, que es lo que muestra en la consola, de forma específica según se requiriera, ya sea con ciclos for o no.

e) Se emplea de manera correcta el concepto de Polimorfismo

Se maneja el polimorfismo por lo descrito en el punto anterior, ya que el polimorfismo incluye sobrecarga de variables, sobrecarga de operadores y sobrecarga de funciones. Lo primero se usa porque al ser varios productos, se reemplazan las variables de la clase producto, lo segundo se usa porque simplifica los proceso de comparación de rating y del año de lanzamiento y los tercero se ocupó en las funciones virtuales y parcialmente virtuales.

f) Se emplea de manera correcta el concepto de Clases Abstractas

En este proyecto la única clase abstracta fue la clase Producto ya que no fue utilizada para la creación de objetos y solo para crear otras clases, esto porque nos sirve para simplificar a las demás clases, ya que encontrando todas las variables que tenían en común, pudimos crear la clase producto.

g) Se sobrecarga al menos un operador en el conjunto de clases propuestas

Se decidió sobrecargar 5 operadores en la clase Producto, esto con el fin de poder usar la sobrecarga en las funciones de búsqueda y comparación con lo que el usuario ingrese. Es decir, para evitar usar los métodos de get en cada función de búsqueda, sólo se crea el operador con un argumento entero y de esta forma se compare directamente el atributo con lo que se está buscando.

h) Se utiliza de manera correcta el uso de excepciones tanto predefinidas como definidas por el programador

Al final de nuestro código en la clase Recomendador se hizo uso de excepciones para evitar que el usuario al ingresar un número de opción equivocado, el código no deje de funcionar. Y en cambio solo muestre un mensaje de error y vuelva a correr el código para que el usuario tenga otra oportunidad de ingresar una opción correcta.

Identificación de casos límite

Nuestra propuesta para la resolución de la situación problema tiene algunas limitantes para un funcionamiento óptimo o de forma generalizada. Una de ellas es que las bases de datos utilizadas no tienen contenido actualizado al año presente (2021), solo arroja recomendaciones hasta el año 2020. También la base de datos utilizada para Prime solo contiene películas, no tiene ni shows ni series. La propuesta se limita a recomendar por años, rating, título y duración, por lo que no arroja resultados si se quiere buscar por director, reparto, género o alguna otra característica, además de que se podría programar para hacer un filtro por más de una característica, para ofrecer una recomendación más exacta.

Conclusión

Hoy en día las plataformas de entretenimiento como lo son las de streaming son una parte muy demandante en nuestra sociedad, por lo que diseñar un recomendador personalizado para los usuarios es fundamental para todos estos servicios como Netflix, Prime o incluso para plataformas de venta en línea, en las que de igual forma se aplican filtros para que los usuarios tengan una buena experiencia al hacer uso de los servicios. Es aquí donde entra la importancia de la programación de sistemas de búsqueda dentro de estos servicios de streaming. Lo cual hace más fácil la elección de contenido por los usuarios

basado en sus gustos, a pesar de que los buscadores de estas plataformas contienen elementos más avanzados y sofisticados. Con ayuda de este proyecto pudimos conocer un poco del procedimiento que se llevó a cabo para el desarrollo de motores de filtro y búsqueda en diferentes plataformas y servicios de internet.

Referencias

Bases de datos obtenidas de: <https://www.kaggle.com/>

UML obtenido de:

https://lucid.app/lucidchart/d9f640bb-a83c-42d1-bb25-0df14269e12b/edit?shared=true&page=0_0#