Taller 1 Lab 1

Juan Esteban Aristizabal 160004903
Alex Fabian Cardenas Baron 160004909

26/04/2023

Punto 1:

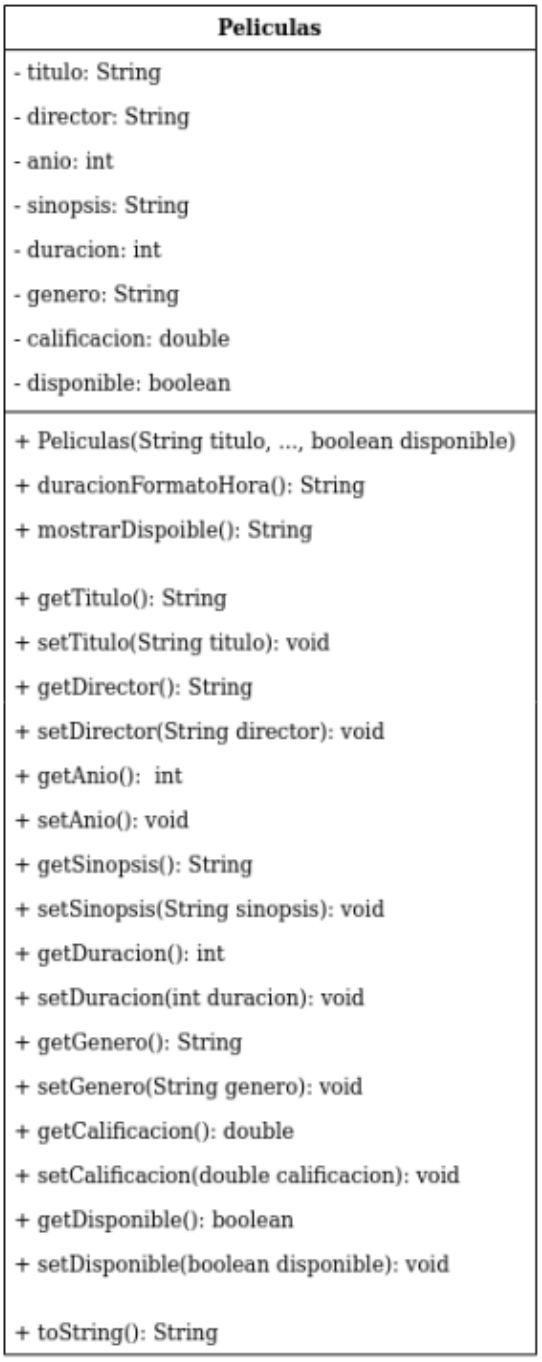
- Enunciado

Realizar un programa en C++ para administrar las películas y documentales disponibles en una videoteca, en esta se encuentran disponibles títulos nacionales e internacionales y busca guardar un histórico de películas y documentales disponibles. Para el desarrollo del programa debe tener en cuenta las siguientes consideraciones:

1. El sistema debe permitir guardar una nueva película (Máximo 5), de cada película se almacena: título, director, sinopsis, duración (minutos), género ('Comedia', 'Terror', 'Documental', 'Acción'), calificación (del 0 al 10) y disponibilidad (si hay en inventario o no).
    a. La duración de la película se debe almacenar la cantidad en minutos, un número entero, ejemplo: 90 minutos,
    b. Al momento de ingresar la calificación se debe validar que el puntaje se encuentre en el intervalo del 0 al 10 (disponible con decimales).
    c. La disponibilidad hace referencia si se encuentra disponible para préstamo o no. Se almacenará un booleano.
    d. Se hace referencia que se debe crear 5 objetos de la clase Películas.
2. El sistema debe permitir modificar la información de alguna película (título, director, sinopsis, duración...) al usuario se le debe preguntar (por consola) qué información desea cambiar y luego realizar/guardar el cambio solicitado.
3. **Método toString():** Se debe visualizar todas las películas almacenadas, con su información correspondiente, además se debe mostrar en pantalla la disponibilidad y la duración de la película en formato horas y minutos, ejemplo: 1 h 30 min.
    a. Para este punto se debe crear un método que muestre la duración de la película en el formato deseado (ej. 1 h 30 min), deben es realizar la conversión de minutos a ese formato.
    b. En el caso de la disponibilidad se debe mostrar un mensaje como: "Disponible" o "Prestado". Para esto, crear un método dentro de la clase que permita consultar el estado de una película (mostrarDisponible()) y ese método debe retornar una variable String con ese mensaje.

    Nota el nombre del método es toString(), no se confundan solo le cambié el nombre al acostumbrado método print(), hacen lo mismo.
4. El sistema debe permitir eliminar una película, el usuario selecciona cual de las 5 películas desea eliminar. Resetear todos los datos en ceros.
5. El sistema debe permitir crear al menos 5 películas (5 objetos).

Este sistema será administrado desde consola por un usuario administrador que podrá ejecutar el programa de videoteca. Para el ejercicio anterior deben tener en cuenta el siguiente diagrama UML de la clase Película.



- Código

```
#include <iostream>
#include <vector> // Uso de vectores
#include <sstream> // Uso del stringstram y los operadores de insercion
#include <iomanip> // Uso del fixed y el setprecision
#include <cstdlib> // Uso del system
#include <limits> // Uso del numeric_limits

using namespace std;

class Peliculas {
   string titulo, director, sinopsis, genero;
   int anio, duracion;
   double calificacion;
   bool disponible;
public:
   Peliculas() {
       titulo = "";
       director = "";
       anio = 0;
       sinopsis = "";
       duracion = 0;
       genero = "";
       calificacion = 0.0;
       disponible = false;
   }
   Peliculas(string titulo, string director, int anio, string sinopsis, int
duracion, string genero,
             double calificacion, bool disponible) {
       this->titulo = titulo;
       this->director = director;
       this->anio = anio;
       this->sinopsis = sinopsis;
       this->duracion = duracion;
       this->genero = genero;
       this->calificacion = calificacion;
       this->disponible = disponible;
   }

   string duracionFormatoHora() const { // const porque no altera los atributos
       int horas = 0, minutos = duracion;
       while (minutos >= 60) {
           minutos -= 60;
           horas += 1;
       }
       return to_string(horas) + "h " + to_string(minutos) + " min";
   }
   string mostrarDisponible() const { // const porque no altera los atributos
       if (disponible) { return "Disponible"; }
       else { return "Prestado"; }
   }
   void resetear() {
       titulo = "";
       director = "";
       anio = 0;
       sinopsis = "";
       duracion = 0;
       genero = "";
```

```
        calificacion = 0.0;
        disponible = false;
    }

    string getTitulo() {
        return titulo;
    }
    void setTitulo(string titulo) {
        this->titulo = titulo;
    }

    string getDirector() {
        return director;
    }
    void setDirector(string director) {
        this->director = director;
    }

    string getSinopsis() {
        return sinopsis;
    }
    void setSinopsis(string sinopsis) {
        this->sinopsis = sinopsis;
    }

    string getGenero() {
        return genero;
    }
    void setGenero(string genero) {
        this->genero = genero;
    }

    int getAnio() {
        return anio;
    }
    void setAnio(int anio) {
        this->anio = anio;
    }

    int getDuracion() {
        return duracion;
    }
    void setDuracion(int duracion) {
        this->duracion = duracion;
    }

    double getCalificacion() {
        return calificacion;
    }
    void setCalificacion(double calificacion) {
        this->calificacion = calificacion;
    }

    bool getDisponible() {
        return disponible;
    }
    void setDisponible(bool disponible) {
        this->disponible = disponible;
    }
```

```
    string toString() {
        stringstream califi; // Creo un objeto stringstream.
        // Con los operadores de insersion, ingreso la calificacion con la presicion
deseada
        // en el objeto califi, luego con la funcion califi.str() lo convierto a
string.
        califi << fixed << setprecision(1) << calificacion;
        return "\nTitulo:         " + titulo + " (" + to_string(anio) + ")"
               + "\nCalificacion:  " + califi.str() + "\nDuracion:       "
               + duracionFormatoHora()
               + "\nDirector:       " + director + "\nGenero:         " + genero
               + "\nDisponible:     " + mostrarDisponible() + "\nSinopsis:\n"
               + sinopsis;
    }
};

void encabezado() {
    cout << "\n################################################" << endl;
    cout << "###       Sistema Gestion de Videoteca       ###" << endl;
    cout << "################################################\n" << endl;
}
int recibirOpcion(vector<int> &opcionesDisponibles) {
    string opc;
    do {
        cin >> opc;
        for (int opcion: opcionesDisponibles){
            if (to_string(opcion) == opc) { return stoi(opc); }
        } // El stoi convierte string a int
        cout << "\n--Digito una opcion incorrecta.--\n" << endl;
    } while (true);
}
double verificarCalificacion(){
    double num;
    do {
        cout << "Digite una entrada valida: ";
        cin >> num;
        if (cin.fail()){ // Pregunta si la entrada fue invalida
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n'); // ignora la
entrada errada
            cout << "\n--Digito una opcion incorrecta.--\n" << endl;
        }
        else if (num >= 0 && num <= 10) { return num; }
        else {cout << "\n--Digito una opcion incorrecta.--\n" << endl;}
    } while (true);
}
bool verificarDisponibilidad(){
    string dispo;
    do {
        cout << "Digite una entrada valida: ";
        cin >> dispo;
        if (dispo == "Y" || dispo == "y") { return true; }
        else if (dispo == "N" || dispo == "n") { return false; }
        else { cout << "\n--Digito una opcion incorrecta.--\n" << endl; }
    } while (true);
}
void subMenuGuardarPeliculas(int &opc, vector<Peliculas> &vectPeliculas){
    string titulo, director, sinopsis, genero;
```

```
    int anio, duracion;
    double calificacion;
    bool disponible;
    cout << "\n-Digite el titulo" << endl;
    cin.ignore();
    getline(cin, titulo);
    vectPeliculas[opc - 1].setTitulo(titulo);
    cout << "\n-Digite el director" << endl;
    cin.ignore();
    getline(cin, director);
    vectPeliculas[opc - 1].setDirector(director);
    cout << "\n-Digite el anio" << endl;
    cin >> anio;
    vectPeliculas[opc - 1].setAnio(anio);
    cout << "\n-Digite la sinopsis" << endl;
    cin.ignore();
    getline(cin, sinopsis);
    vectPeliculas[opc - 1].setSinopsis(sinopsis);
    cout << "\n-Digite la duracion en minutos" << endl;
    cin >> duracion;
    vectPeliculas[opc - 1].setDuracion(duracion);
    cout << "\n-Digite el genero: " << endl;
    cin >> genero;
    vectPeliculas[opc - 1].setGenero(genero);
    cout << "\n-Digite la calificacion" << endl;
    calificacion = verificarCalificacion();
    vectPeliculas[opc - 1].setCalificacion(calificacion);
    cout << "\n-Digite la disponibilidad (Y/N)" << endl;
    disponible = verificarDisponibilidad();
    vectPeliculas[opc - 1].setDisponible(disponible);
}
void subMenuPeliculas(vector<Peliculas> &vectPeliculas) {
    for (int n = 0; n < 5; n++) {
        if (vectPeliculas[n].getTitulo() != "") {
            cout << n + 1 << ". " << vectPeliculas[n].getTitulo() << " ("
                 << vectPeliculas[n].getAnio() << ")" << endl;
        } else { cout << n + 1 << ". Otra pelicula " << n + 1 << endl; }
    }
}
void subMenuModificarPeliculas(int &opc, vector<Peliculas> &vectPeliculas,
vector<int> &opcionesDisponibles){
    string titulo, director, sinopsis, genero;
    int anio, duracion, opc2;
    double calificacion;
    bool disponible;
    system("cls");
    encabezado();
    cout << "\t\tModificar pelicula" << endl;
    cout << "    " << vectPeliculas[opc - 1].getTitulo() << " ("
         << vectPeliculas[opc - 1].getAnio() << ")" << " - Director: "
         << vectPeliculas[opc - 1].getDirector() << endl;
    cout << "\n1. Titulo" << endl;
    cout << "2. Director" << endl;
    cout << "3. Anio" << endl;
    cout << "4. Sinopsis" << endl;
    cout << "5. Duracion" << endl;
    cout << "6. Genero" << endl;
    cout << "7. Calificacion" << endl;
    cout << "8. Disponible" << endl;
```

```
    cout << "\nQue desea cambiar?" << endl;
    opc2 = recibirOpcion(opcionesDisponibles={1,2,3,4,5,6,7,8});
    if (opc2 == 1) {
        cout << "\n-Digite el titulo" << endl;
        cin.ignore();
        getline(cin,titulo);
        vectPeliculas[opc - 1].setTitulo(titulo);
    } else if (opc2 == 2) {
        cout << "\n-Digite el director" << endl;
        cin.ignore();
        getline(cin, director);
        vectPeliculas[opc - 1].setDirector(director);
    } else if (opc2 == 3) {
        cout << "\n-Digite el anio" << endl;
        cin >> anio;
        vectPeliculas[opc - 1].setAnio(anio);
    } else if (opc2 == 4) {
        cout << "\n-Digite la sinopsis" << endl;
        cin.ignore();
        getline(cin, sinopsis);
        vectPeliculas[opc - 1].setSinopsis(sinopsis);
    } else if (opc2 == 5) {
        cout << "\n-Digite la duracion en minutos" << endl;
        cin >> duracion;
        vectPeliculas[opc - 1].setDuracion(duracion);
    } else if (opc2 == 6) {
        cout << "\n-Digite el genero" << endl;
        cin >> genero;
        vectPeliculas[opc - 1].setGenero(genero);
    } else if (opc2 == 7) {
        cout << "\n-Digite la calificacion" << endl;
        calificacion = verificarCalificacion();
        vectPeliculas[opc - 1].setCalificacion(calificacion);
    } else if (opc2 == 8) {
        cout << "\n-Digite la disponibilidad (Y/N)" << endl;
        disponible = verificarDisponibilidad();
        vectPeliculas[opc - 1].setDisponible(disponible);
    }
}
void menuPrincipal(vector<Peliculas> &vectPeliculas) {
    int opc, opc2;
    vector<int> opcionesDisponibles;
    do {
        system("cls");
        encabezado();
        cout << "Menu:" << endl;
        cout << "1. Mostrar todas las peliculas" << endl;
        cout << "2. Guardar una nueva pelicula" << endl;
        cout << "3. Modificar peliculas" << endl;
        cout << "4. Eliminar pelicula" << endl;
        cout << "5. Salir" << endl;
        cout << "\nQue desea realizar?" << endl;
        opc = recibirOpcion(opcionesDisponibles={1,2,3,4,5});
        opcionesDisponibles.clear();

        system("cls");
        if (opc == 1) {
            encabezado();
            cout << "\t\tPeliculas en la agenda\n" << endl;
```

```
            for (int n = 0; n < 5; n++) {
                if (vectPeliculas[n].getTitulo() != "") {
                    cout << vectPeliculas[n].toString() << endl;
                }
            }
            system("pause");
        } else if (opc == 2) {
            encabezado();
            cout << "\t\tGuardar pelicula\n" << endl;
            for (int n = 0; n < 5; n++) {
                if (vectPeliculas[n].getTitulo() == "") {
                    cout << n + 1 << ". Otra pelicula " << n + 1 << endl;
                    opcionesDisponibles.push_back(n + 1);
                }
            }
            cout << "\nDonde desea guardar la pelicula?" << endl;
            opc = recibirOpcion(opcionesDisponibles);
            subMenuGuardarPeliculas(opc, vectPeliculas);
        } else if (opc == 3) {
            encabezado();
            cout << "\t\tModificar peliculas\n" << endl;
            subMenuPeliculas(vectPeliculas);
            cout << "\nCual pelicula desea modificar?" << endl;
            opc = recibirOpcion(opcionesDisponibles={1,2,3,4,5});
            subMenuModificarPeliculas(opc, vectPeliculas, opcionesDisponibles);
        } else if (opc == 4) {
            encabezado();
            cout << "\t\tEliminar pelicula\n" << endl;
            subMenuPeliculas(vectPeliculas);
            cout << "\nCual pelicula desea eliminar?" << endl;
            opc2 = recibirOpcion(opcionesDisponibles={1,2,3,4,5});
            vectPeliculas[opc2 - 1].resetear();
        }
    } while (opc != 5);
}
int main() {
    string sipno1, sipno2;
    sipno1 = "La codicia y la discriminacion de clase amenazan la relacion
simbiotica "
             "recien formada entre la acaudalada familia de Park y el indigente clan
Kim.";
    sipno2 = "En Gotham City, el comediante con problemas mentales Arthur Fleck "
             "es ignorado y maltratado por la sociedad. Luego se embarca en una "
             "espiral descendente de revolucion y crimenes sangrientos. Este "
             "camino lo pone cara a cara con su alter ego: el Joker.";

    vector<Peliculas> vectPeliculas;
    vectPeliculas.push_back(Peliculas("Parasito", "Bong Joon Ho", 2019, sipno1, 132,
"Drama", 8.6, true));
    vectPeliculas.push_back(Peliculas("Guason", "Todd Phillips", 2019, sipno2, 122,
"Drama", 8.5, false));
    vectPeliculas.push_back(Peliculas());
    vectPeliculas.push_back(Peliculas());
    vectPeliculas.push_back(Peliculas());
    menuPrincipal(vectPeliculas);
    return 0;
}
```

- Descripción

Se realizó la creación de la clase Peliculas, con todos sus miembros (atributos y métodos), entre los métodos se encuentran todos los get y los set de cada atributo. Además, se crearon los métodos duracionFormatoHora, mostrarDisponible, resetear y toString. Siendo los dos primeros y el último, de tipo const. También, se crearon sus respectivos constructores, uno por defecto y el otro que recibe parámetros.

Así mismo, se realizó el menú con cada una de las funcionalidades especificadas en el taller, las cuales derivaron en la creación de funciones fuera de la clase, dichas funciones se realizaron con el propósito de hacer que el código sea legible y modular; por ello cada función recibe parámetros por referencia.

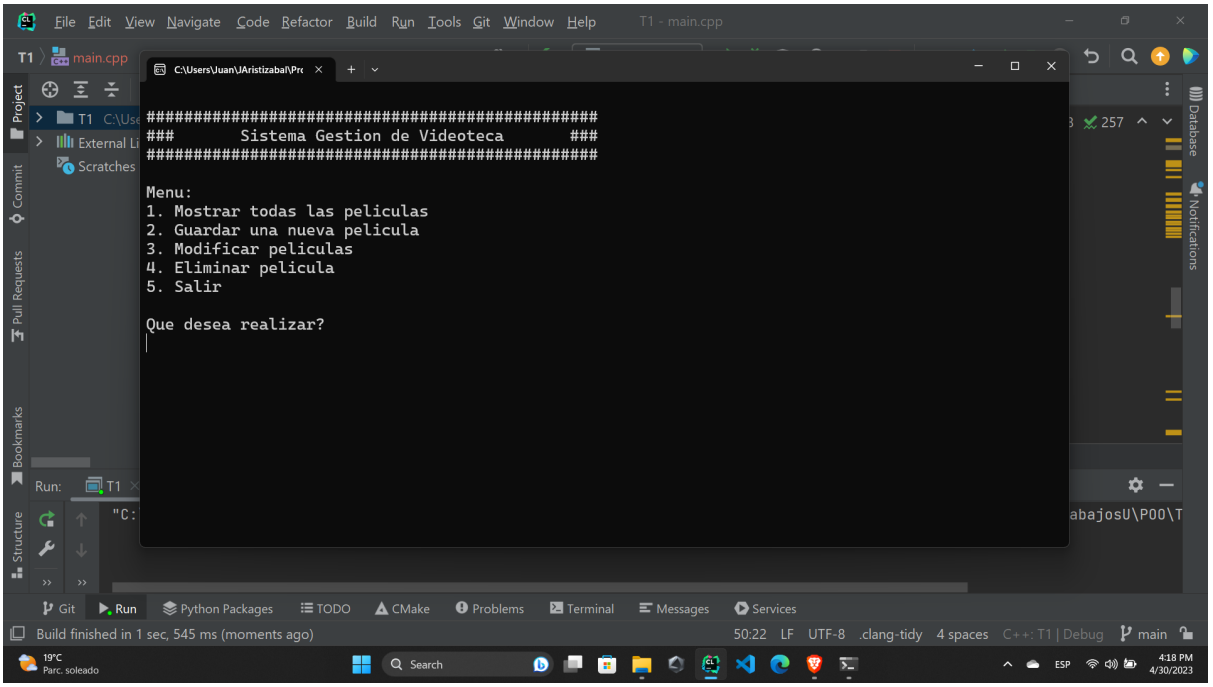
En algunas de las funcionalidades del menú se usaron los métodos set, necesarios para modificar los atributos de cada objeto, ya que estos son privados.

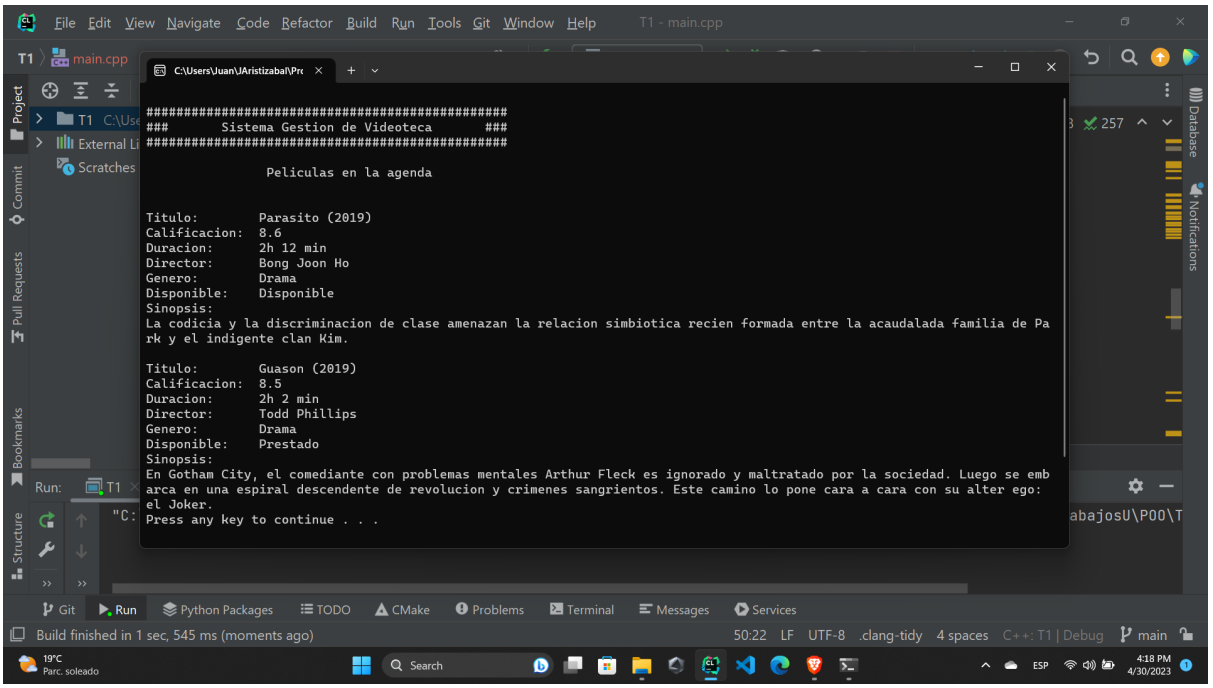
Para este trabajo se realizó el manejo de errores para los momentos en que el usuario debe ingresar información por teclado, específicamente en cada menú, submenú y, en los campos de calificación y disponibilidad.

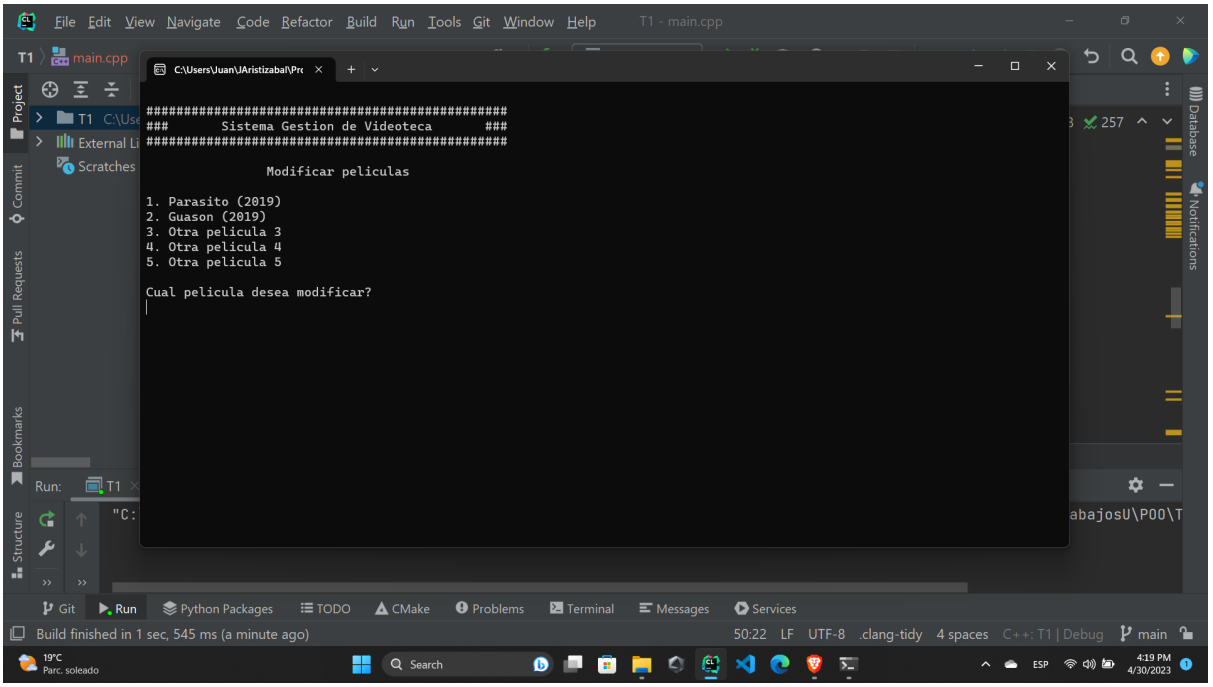
También, se utilizó el conocimiento adquirido en clase, sobre el funcionamiento de los vectores y se realizó la creación de los objetos dentro de un vector llamado vectPeliculas. Tres de los objetos se crearon con ayuda del constructor por defecto y los otros dos se crearon con el constructor que recibe parámetros. Para llenar los campos de estos dos últimos objetos, se usó la información de la guía, sobre las películas que aparecen allí.

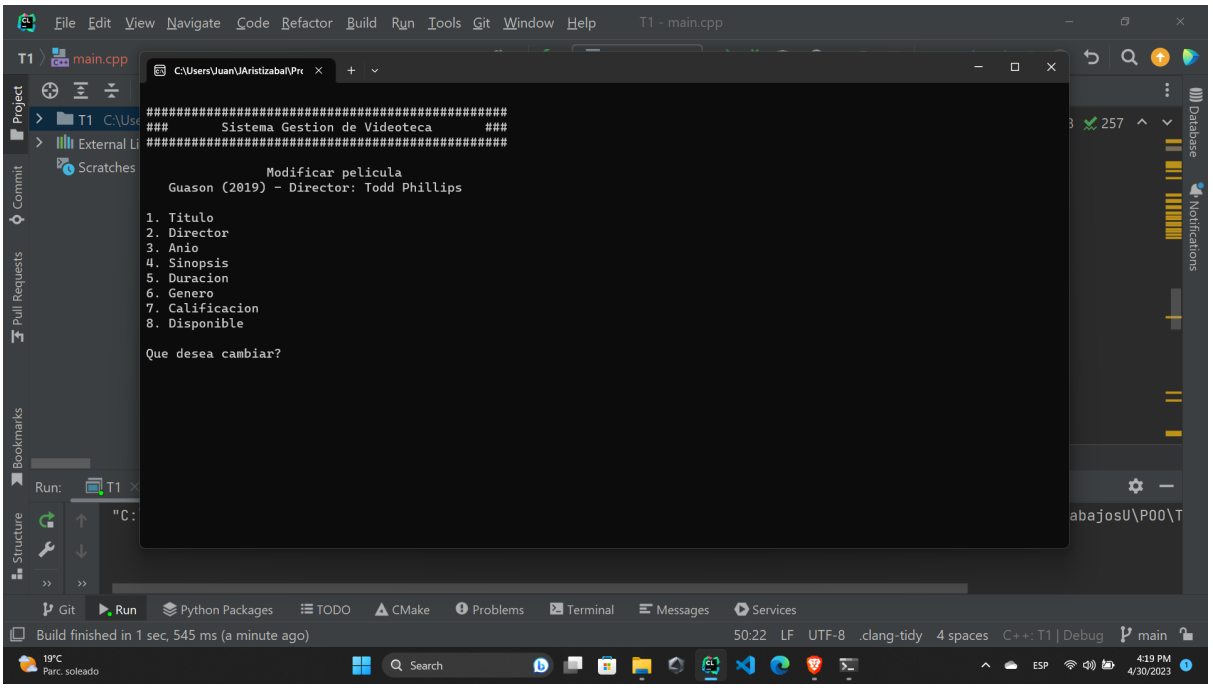
Al momento de pegar el código desde el Clion al word, se trae consigo un formato de texto diferente, además de que desacomoda un poco el código, lo cual intenté arreglar, pero se termina descuadrando más. Por ello decidimos dejarlo así.

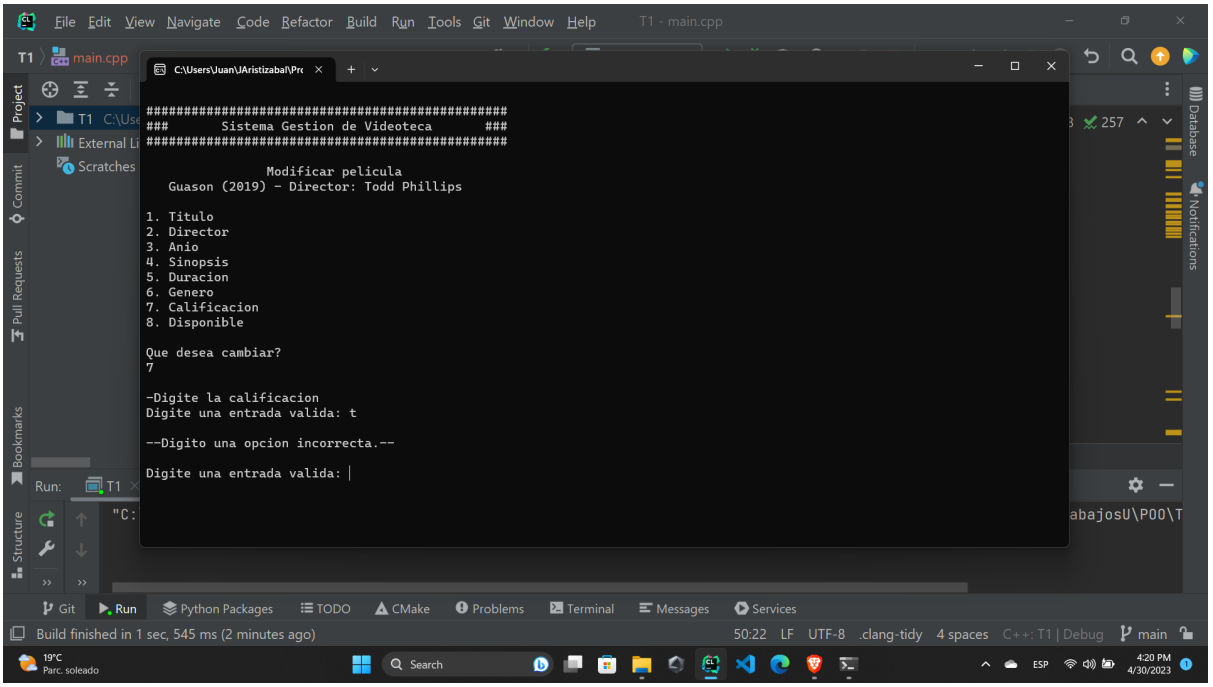
- Captura


C:\Users\Juan\JAristizabal\Pro
##
Sistema Gestion de Videoteca
##
Menu:
1. Mostrar todas las peliculas
2. Guardar una nueva pelicula
3. Modificar peliculas
4. Eliminar pelicula
5. Salir
Que desea realizar?


C:\Users\Juan\JAristizabal\Pro
##
Sistema Gestion de Videoteca
##
Peliculas en la agenda
Titulo: Parasito (2019)
Calificacion: 8.6
Duracion: 2h 12 min
Director: Bong Joon Ho
Genero: Drama
Disponible: Disponible
Sinopsis:
La codicia y la discriminacion de clase amenazan la relacion simbiotica recien formada entre la acaudalada familia de Park y el indigente clan Kim.
Titulo: Guason (2019)
Calificacion: 8.5
Duracion: 2h 2 min
Director: Todd Phillips
Genero: Drama
Disponible: Prestado
Sinopsis:
En Gotham City, el comediante con problemas mentales Arthur Fleck es ignorado y maltratado por la sociedad. Luego se embarca en una espiral descendente de revolucion y crimenes sangrientos. Este camino lo pone cara a cara con su alter ego: el Joker.
Press any key to continue . . .


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
T1 - main.cpp
C:\Users\Juan\JAristizabal\Pro
##
Sistema Gestion de Videoteca
##
Modificar peliculas
1. Parasito (2019)
2. Guason (2019)
3. Otra pelicula 3
4. Otra pelicula 4
5. Otra pelicula 5
Cual pelicula desea modificar?
Run:
Git
Run
Python Packages
TODO
CMake
Problems
Terminal
Messages
Services
Build finished in 1 sec, 545 ms (a minute ago)
50:22 LF UTF-8 .clang-tidy 4 spaces C++: T1 | Debug main
19°C
Parc. soleado
Search
ESP
4:19 PM
4/30/2023


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
T1 - main.cpp
C:\Users\Juan\JAristizabal\Pro
##
Sistema Gestion de Videoteca
##
Modificar pelicula
Guason (2019) - Director: Todd Phillips
1. Titulo
2. Director
3. Anio
4. Sinopsis
5. Duracion
6. Genero
7. Calificacion
8. Disponible
Que desea cambiar?
Run:
Git
Run
Python Packages
TODO
CMake
Problems
Terminal
Messages
Services
Build finished in 1 sec, 545 ms (a minute ago)
50:22 LF UTF-8 .clang-tidy 4 spaces C++: T1 | Debug main
19°C
Parc. soleado
Search
ESP
4:19 PM
4/30/2023


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
T1 - main.cpp
C:\Users\Juan\JAristizabal\Pro
##
Sistema Gestion de Videoteca
##
Modificar pelicula
Guason (2019) - Director: Todd Phillips
1. Titulo
2. Director
3. Anio
4. Sinopsis
5. Duracion
6. Genero
7. Calificacion
8. Disponible
Que desea cambiar?
7
-Digite la calificacion
Digite una entrada valida: t
--Digito una opcion incorrecta.--
Digite una entrada valida:
Run:
Git
Run
Python Packages
TODO
CMake
Problems
Terminal
Messages
Services
Build finished in 1 sec, 545 ms (2 minutes ago)
50:22 LF UTF-8 .clang-tidy 4 spaces C++: T1 | Debug main
19°C
Parc. soleado
Search
ESP
4:20 PM
4/30/2023