МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В. Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа № 2**

по дисциплине: Объектно-ориентированное программирование

тема: «Модульное программирование. Интерфейсы»

Выполнил: ст. группы ПВ-223

Игнатьев Артур Олегович

Проверил:

асс. Черников Сергей Викторович

Белгород 2024г.

**Лабораторная работа №2**

**«Модульное программирование. Интерфейсы»**

**Цель работы:** Получение навыков модульной декомпозиции предметной области, создания модулей. Разработка интерфейсов.

Задание:

1. Разработать программу согласно своего варианта.

**Ход выполнения лабораторной работы:**

**Вариант 3**

Разработать программу «Домашняя фильмотека»

Код программы:

Файл interface.h

#ifndef OOP\_INTERFACE\_H  
#define OOP\_INTERFACE\_H  
  
#include <fstream>  
#include <string>  
#include <iostream>  
  
const int N = 30;  
  
struct info\_film {  
  
 char \*adress;  
 std::string name;  
 FILE \*f;  
 int time;  
 int year;  
  
 info\_film();  
  
 void set\_name(std::string);  
  
 void set\_time(int);  
  
 void set\_year(int);  
  
 void set\_adress(char \*);  
  
 void print\_name();  
  
 void print\_year();  
  
 void print\_time();  
  
 void print\_all();  
  
 void play\_film();  
  
 void off\_film();  
  
 void creatе\_film();  
};  
  
struct film\_collection {  
 film\_collection();  
  
 void add\_film(info\_film);  
  
 void dell\_film(std::string s);  
  
 int search\_film(std::string s);  
  
 void output();  
  
 info\_film film[N];  
 int n;  
};  
  
  
struct sort\_film {  
  
 film\_collection film;  
  
 void name\_sort();  
  
 void year\_sort();  
};  
  
  
struct unity\_film {  
  
 void year\_unity(info\_film t[], int n);  
};  
  
struct play {  
  
 film\_collection film;  
  
 int run\_film(std::string s);  
  
 void run();  
};  
  
#endif //OOP\_INTERFACE\_H

Файл interface.cpp

#include "interface.h"  
#include <iostream>  
#include <string>  
#include <fstream>  
#include <stdlib.h>  
#include <locale.h>  
  
  
using namespace std;  
  
info\_film::info\_film() {  
 name = "";  
 time = 0;  
 year = 0;  
  
}  
  
void info\_film::set\_name(std::string s) {  
 name = s;  
}  
  
  
void info\_film::set\_time(int n) {  
 time = n;  
}  
  
void info\_film::set\_year(int n) {  
 year = n;  
}  
  
void info\_film::print\_name() {  
 std::cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"  
 << endl;  
 std::cout << "Название фильма: " << name << endl;  
}  
  
  
void info\_film::print\_time() {  
 std::cout << "Продолжительность фильма: " << time << " min" << endl;  
}  
  
void info\_film::print\_year() {  
 std::cout << "Год выпуска: " << year << endl;  
}  
  
  
void info\_film::print\_all() {  
 print\_name();  
 cout << " ";  
 print\_year();  
 cout << " ";  
 print\_time();  
 cout << " ";  
}  
  
void info\_film::set\_adress(char \*s) {  
 adress = s;  
}  
  
void info\_film::play\_film() {  
 cout << "======================= Фильм =======================" << endl;  
  
}  
  
void info\_film::off\_film() {  
 system("CLS");  
}  
  
int play::run\_film(std::string name) {  
 int t = film.search\_film(name);  
 cout << t;  
 if (t != -1)  
 film.film[t].play\_film();  
 else  
 cout << "Фильм не найден!" << endl;  
 return t;  
}  
  
film\_collection::film\_collection() {  
 n = 3;  
 char \*s = (char \*) "C:\\movies\\1.mp4";  
 char \*s1 = (char \*) "C:\\movies\\2.mp4";  
 char \*s2 = (char \*) "C:\\movies\\3.mp4";  
  
 film[0].set\_name("Bleach");  
 film[0].set\_adress(s);  
 film[0].set\_year(2004);  
 film[0].set\_time(8784);  
  
  
 film[1].set\_name("Grand Blue");  
 film[1].set\_adress(s1);  
 film[1].set\_year(2018);  
 film[1].set\_time(288);  
  
  
 film[2].set\_name("Hellsing");  
 film[2].set\_adress(s2);  
 film[2].set\_year(2001);  
 film[2].set\_time(299);  
}  
  
void film\_collection::add\_film(info\_film t) {  
 if (search\_film(t.name) == -1) {  
 film[n] = t;  
 n++;  
 } else  
 std::cout << "фильм уже добавлен!" << endl;  
}  
  
void film\_collection::dell\_film(std::string s) {  
 int k = search\_film(s);  
 if (k != -1) {  
 film[k] = film[n - 1];  
 --n;  
 } else  
 std::cout << "фильм не найден!" << endl;  
}  
  
int film\_collection::search\_film(std::string s) {  
 int i = 0;  
 while (i < n) {  
 if (film[i].name == s) {  
 return i;  
 }  
 i++;  
 }  
 return -1;  
}  
  
void info\_film::creatе\_film() {  
 std::string s;  
  
 int n;  
 float n1;  
 cout << "Введите название : ";  
 cin >> s;  
 set\_name(s);  
  
 cout << "Введите год выпуска: ";  
 cin >> n;  
 set\_year(n);  
 cout << "ВВедите продолжительность: ";  
 cin >> n1;  
 set\_time(n1);  
}  
  
void sort\_film::name\_sort() {  
 for (int i = 0; i < film.n - 1; i++)  
 for (int j = i + 1; j < film.n; j++) {  
 if (film.film[i].name > film.film[j].name)  
 std::swap(film.film[i], film.film[j]);  
 }  
}  
  
  
void sort\_film::year\_sort() {  
 for (int i = 0; i < film.n - 1; i++)  
 for (int j = i + 1; j < film.n; j++) {  
 if (film.film[i].year > film.film[j].year)  
 std::swap(film.film[i], film.film[j]);  
 }  
}  
  
void unity\_film::year\_unity(info\_film film[], int n) {  
 int f = 0, k = 1, i = 0, j = 1;  
  
 std::cout << "-------------- 1 папка -------------- " << endl;  
 while (i <= n - 1 && j <= n) {  
 if (film[i].year == film[j].year) {  
 if (!f) film[i].print\_all();  
 f = 1;  
 film[j].print\_all();  
 j++;  
 continue;  
 } else {  
 if (f) {  
 k++;  
 std::cout << "-------------- " << k << " папка -------------- " << endl;  
 }  
 f = 0;  
 i = j;  
 j = i + 1;  
 }  
 }  
  
 i = 0;  
 j = 1;  
 while (i <= n - 1 && j <= n) {  
 if (film[i].year == film[j].year) {  
 i += 2;  
 j += 2;  
 } else {  
 film[i].print\_all();  
 i++;  
 j++;  
 }  
 }  
}  
  
void film\_collection::output() {  
 for (int i = 0; i < n; i++) {  
 film[i].print\_all();  
 }  
}  
  
void play::run() {  
 int n = 0, m, v, f = 0;  
 std::string s, s1, y;  
 sort\_film c;  
 while (n != 6) {  
 if (!f) c.film.output();  
 f = 0;  
 cout << "Включить фильм - 1" << endl;  
 cout << "Добавить фильм в фильмотеку - 2" << endl;  
 cout << "Удалить фильм из фильмотеки - 3" << endl;  
 cout << "Отсортировать фильмотеку - 4" << endl;  
 cout << "Объеденить фильмы по папкам - 5" << endl;  
 cout << "Выйти из домашней фильмотеки - 6" << endl;  
 cout << "Ваш выбор: ";  
 cin >> n;  
 switch (n) {  
 case 1: {  
 cout << "Какой фильм включить? " << endl;  
 cin >> s;  
 cout << endl;  
 int t = run\_film(s);  
 if (t != -1) {  
 cout << endl;  
  
 cout << "Выключить фильм? - 1" << endl;  
  
 c.film.film[t].off\_film();  
 break;  
  
 }  
  
 break;  
 }  
 case 2: {  
 info\_film t;  
 t.creatе\_film();  
 c.film.add\_film(t);  
 break;  
 }  
 case 3: {  
 cout << "Какой фильм удалить? " << endl;  
 cin >> s1;  
 c.film.dell\_film(s1);  
 break;  
 }  
 case 4: {  
 cout << "Отсортировать по названию фильма - 1 " << endl;  
 cout << "Отсортировать по году создания - 2 " << endl;  
 cin >> m;  
 switch (m) {  
 case 1: {  
 c.name\_sort();  
 break;  
 }  
 case 2: {  
 c.year\_sort();  
 break;  
 }  
  
 }  
 break;  
 }  
  
  
 case 5: {  
 cout << "Объеденить фильмы в папку по году их создания - 1" << endl;  
 cin >> m;  
  
 c.year\_sort();  
 unity\_film u;  
 u.year\_unity(c.film.film, c.film.n);  
 f = 1;  
  
 }  
 }  
 }  
}

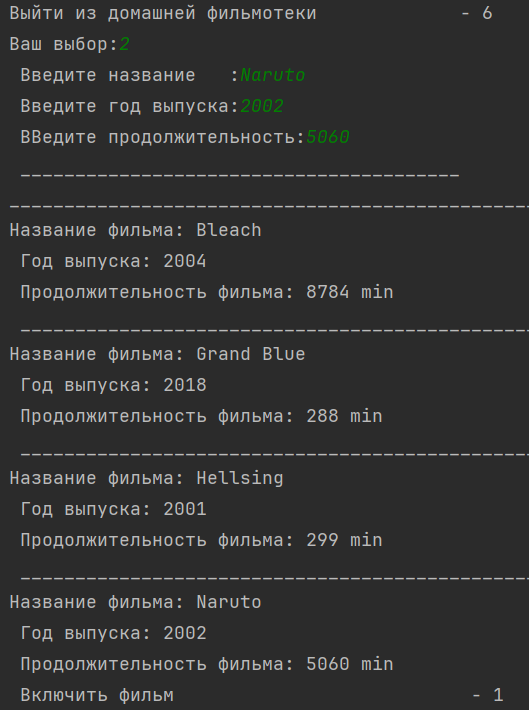
Файл main.cpp

#include <iostream>  
#include "../../libs/alg/labs/lab2/interface.h"  
#include<stdio.h>  
#include<string.h>  
#include<stdlib.h>  
#include <iostream>  
#include<windows.h>  
#include<locale.h>  
  
  
using namespace std;  
  
int main() {  
 SetConsoleCP(1251);  
 SetConsoleOutputCP(CP\_UTF8);  
 setlocale(LC\_ALL, "Russian");  
  
 play r;  
 r.run();  
}

Результат работы программы:

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание



**Вывод:** в ходе выполнения лабораторной работы получил навыки модульной декомпозиции предметной области, создания модулей, разработки интерфейсов.