**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ-ВАРНА**



КАТЕДРА :"КОМПЮТЪРНИ НАУКИ И ТЕХНОЛОГИИ"

Курсов Проект №11 по ООП

Програма за обработка на филми

Изготвил: Даниел Костов Проверил:доц. М. Карова

ФН: 18621439

Специалност: КСТ Курс: 2-ри Група: 4“а“

**Условие на задачата**

I. Да се състави клас за песни CFilms с член променливи:

Заглавие на филма, продуцент, тираж, година на производство, жанр.

Съставете функции за:

1. за установяване на член променливите
2. функция за извеждане в изходен поток (отделно файл, отделно конзола)

Съставете конструктори:

1. подразбиращ се конструктор
2. копиращ конструктор
3. експлицитен конструктор

Съставете функции за:

1. За валидизиране на йанра
2. Изчисляване възрастта на филма
3. Извеждане данните за филма на принтер
4. Оператор за присвояване =
5. Логически оператор, който проверява дали филма е създаден през текущата година
6. Оператор за сравнение > (по тираж)
7. Оператор за сравнение (по продуцент)

II. Да се състави клас CRazpredelitel, който съдържа CFilms с член променливи име на разпространителя, брой филми, брой приходи за една година.

Съставете следните функции:

1. Създаване на обект от друг обект
2. Създаване на обект чрез експлицитно зададени параметри
3. Функция, определяща дали филма е прдлаган повече от 1 път

III. Съставете главна програма, илюстрираща използването на създадените класове. Да се създаде контейнер вектор от 10 обекта от класа CRazpredelitel и да се направят следните справки, като:

1. извейдане в отделен списък (list) на филмите с тираж над 50 000 копия, от жанр комедия
2. да се изведат в отделни файлове всички разпространители, които са разпространяват филми на едни и същи продуценти
3. да се изведат заглавията на филмите, които са над 20 за продуцент, чието име е подадено като параметър
4. да се изведат всички данни за разпространителите с брой приходи получени като караметър от текстов файл
5. при подаден като параметър година на роизводство, да се изведат разпространителите на такива филми, които са от жанр екшън или драма

IV. Документиране на курсовия проект: .doc файл със заглавна страница, условие на задачата, кратко описание на класовете и функциите и листинг на програмата с коментари.

**Кратко описание на Програмата**

**Предназначение:**

Прогамата обработва данните подадени от Потребителя, и ги записва във файлове. При използване на Програмата, Потребителя създава филми, по свой избор, и задава разпространители който ще предлагат тези филми, като избора кой разпространител кой филм да предлага е от Потребителя.

Също, Потребителя ще може да създава файлове с данни за всички филми и ще може да създава файлове с името на продуцента, съдържащи разпределителите на филмите му.

**Данни с които работи Програмата:**

Програмата работи само с целочислени данни ( от тип int ) и текстови данни ( от тип string ). Програмата ще може да работи със всякакви променливи без проблем, при леко коригиране на кода. А данните за момента се записват в текстови файлове.

**Възможни грешки и защита включена в програмата:**

Възможните грешки в програмата са спрени с редица защити които да предотвратят нежелани ефекти върху Програмата, като:

* Не създаден файл, преди използвана на функция използваща файлове
* Ненамерени данни поради грешка в правописа

Също, са въведени редица проверки за потвърждение и улесняване на използването на Програмата.

Възможни са грешки при непредвидено използване на Програмата.

**Възможности за развитие на програмата:**

Тъй като програмата е част от университетски проект, заданието е доста специфично и всяко разширение ще изисква допълнителен труд, и може би изцяло ново проектиране на структурата й.

Въпреки това, могат да се добавят допълнителни функции, като:

* Сортиране на данни във файл по избран начин
* Модифициране и Изтриване на вече записани данни

**Обща структура на Програмата**

Програмата е разделена на два класа и една главна функция, с допълнителни функции за по-добра работа.

**Клас CFilms**

**Описание:**

CFilms е клас който ще се използва в член променливата на класа **CRazprostranitel** от тип **vector<CFilms>**. Също чрез този клас ще се създават файлове.

**Член-Променливи:**

string **fname** - заглавие на филма;

string **producer** - име на продуцента

int **tiraj** - тирайж, колко пъти е предлаган филма

int **year** - година на производство

string **genre** - жанр на филма

**Член-Функции:**

**CFilms()** - подразбиращ конструктор

**CFilms**(const CFilms& fil) - копиращ конструктор

**CFilms**(string nam, string pro, string gen, int tir, int yer) - експлицитен конструктор

**Аксесори:**

string **getFilmName() const** - връща заглавието на фълма

string **getProducer() const** - връща името на продуцента на фълма

string **getGenre() const** - връща жанра на фълма

int **getTiraj() const** - връща тиража на фълма

int **getYear() const** - връща годината на производство на фълма

**Мутатори:**

void **setFilmName(const string s)** - променя заглавието на фълма

void **setProducer(const string s)** - променя името на продуцента на фълма

void **setGenre(const string s)** - променя жанра на фълма

void **setTiraj(const int i)** - променя тиража на фълма

void **setYear(const int i)** - променя годината на производство на фълма

**Други функции:**

**istream& input(istream& in)** - функция за въвеждане във филма

**ostream& outputConsolet(ostream& out)** - функция за извеждане в изходен поток конзола

**void outputFile()** - четене от файл и извеждане на конзолата всички филми от главния файл

**int calculateAge()** - презсмята възрастта на филма като извайда текущата година с годината му на производство

**void sendToPrint()** - извеждат се данните от филма една по една към принтера

**bool validateGenre(string s)** - сравнява низа подаден като параметър с жанра на филма, и връща истина или лъжа в зависимост от ресултата

**Оператори:**

**CFilms operator=(const CFilms& fil)** - операторза присвояване на член променливите с на подадения клас

**bool operator!()** - логически оператор, който проверява дали филма е от тази година

**bool operator>(const CFilms& fil)** - оператор за сравнение по тираж

**bool operator==(const CFilms& fil)** - оператор за сравнение по продуценти

**friend istream& operator>>(istream& in, CFilms& fil)** - оператор за вход който вика функцията за въвеждане във файла

**friend ostream& operator<<(ostream& out, CFilms& fil)** - оператор за изход който вика функцият за извеждане в изхоен поток конзола

**Клас CRazprostranitel**

**Описание:**

класа CRazprostranitel събира във вектор обекти от клас CFilms, чрез които ще бъде записан във файл

**Член-Променливи:**

vector<CFilms> **vfilm** - вектор с филмите които разпространителя разпространява

string **rname** - името на разпространителя

double **revenue** - печалба за година, от филмите

**Член-Функции:**

**CRazprostranitel()** - подразбиращ конструктор

**CRazprostranitel**(const CRazprostranitel& raz) - копиращ конструктор

**CRazprostranitel**(vector<CFilms>& fil, string rnm, double rev) - експлицитен конструктор

**Аксесори:**

vector<CFilms> **getCFilmVector() const** - връща вектора с филми

string **getRazName() const** - връща името на разпространителя

int **getRevenue() const** - връща печалбата за година, на разпространителя

**Други функции:**

**istream& RInput(istream& in)** - функция за въвеждане във класа

**ostream& ROutput(ostream& out)** - функция за извеждане в изходен поток

**Оператори:**

**friend istream& operator>>(istream& in, CRazprostranitel& raz)** - оператор за вход който вика функцията за въвеждане във файл

**friend ostream& operator<<(ostream& out, CRazprostranitel& raz)** - оператор за изход който вика функцият за извеждане в изхоен поток

**Допълнителни функции:**

void **makeCFilm**(string& nm, string& pro, int& tir, int& yer, string& gen) - функция за създаване на обект от клас CFilms, който се показва и записва във файл film.txt

void **makeCRazprostranitel**(vector<CRazprostranitel>& vr , string &rnm, double& rv, string vs[]) - функция за създаване на обект от клас CRazprostranitel, който се показва и записва във вектора, подаден като параметър

void **listKomediq50k**(list<CFilms>& k50k, string& kom) - Функция за извеждане на филми с тираж над 20 за даден продуцент.

Функцията получава лист от клас CFilms и низ със въведен от потребителя дума за жанр комедия. Файлът се отваря два пъти, първия път за да преброи колко обекта има в него, а вторя път за да вземе обектите без грешка. След това сравнява жанра на обекта с подадения низ, и ако тиража на обекта е повече от 50 000, се вкарват в листа.

bool **check**(vector<string>& vs, string& s) - Помощна функция за **separateFilesByProducer.** Функцията получава низ с име и вектор с имена на продуценти, и проверява дали името вечего има във вектора и съответно връща истина или лъжа.

void **separateFilesByProducer**(vector<CRazprostranitel>& vr) - Функция която създава файлове с имената на продуцентите и вкарва данните за разпространителите, които предлагат негови филми, тях.

Алгоритъма отваря един for цикъл за да вземе всеки разпространител поотделно. След това се отваря втори for цикъл който да мине през вектора с филми на разпространителите, откъдето се взима името на продуцента. Създава се файл с негово име и се вика функция **check,** и при истина се записват данните на разпространителите във файла.

void **showFilmNames20**(vector<CRazprostranitel>& vr, string& prod) - Функция за извеждане на филми с тираж над 20 за даден продуцент.

Функцията получава вектора от клас CRazprostranitel с всички разпространители и низ с име на продуцент. Алгоритъма започва с отваряне на един for цикъл цикъл за да се вземе всеки разпространител поотделно. След това се отваря втори for цикъл който да мине през вектора с филми на разпространителите. Ако продуцента на филма съответства с подадения низ, и тиража е по-гоям от 20, ще се изведе

void **showRevFromFileName**(string& filename) - Функция за извеждане на печалбата за година на разпространител, от даден файл.

Алгоритъма отваря дадения файл два пъти, първия път за да преброи колко обекта има в себе си, а вторя път за да изведе данните за разпределителите ( име, печалба за годината и брой филми ).

void **showRazprosNaEkshanDrama**(vector<CRazprostranitel>& vr, int& yearch, string& ek, string& dr) - Функция за извеждане на разпространители с филми от жанр екшън или драма за дадна година.

Функцията получава вектора от клас CRazprostranitel с всички разпространители, годината за която ще проверяваме, и два низа със въведени от потребителя думи за жанр екшън и драма. Създава се един флаг който ще използваме да разберем дали има филми които отговарят на изискванията ни. Отваря се един for цикъл за да се вземе всеки разпространител поотделно. След това се отваря втори for цикъл който да мине през вектора с филми на разпространителите, откъдето се проверява дали годината съответства с подадената, и дали жанра съответва с единия низ. Алгоритъма се повтаря и за другия низ. Ако и двете условия се изпълнят, се изкарва името на разпространителя, филма и жанра му, и флага се сменя на истина. Ако флага не се е променил влиза в проверка на лъжа и излиза надпис за ненамерен филм.

void **menu**() - Функция за меню, чрез тази функция се използват останалите допълнителни функции

**Тестови Резултати**

**Меню:**



**Функция едно:** Създаване на филми във файл film.txt



**Функция две:** Създаване на разпространители и записване във вектор



**Функция три:** Извеждане на филмите от жанр комедия и тираж над 50 000 от файл



**Функция четири:** Създаване на файлове с името на продуцентите и записване информацията за разпространителите които предлагат техни филми в тях



файл pro1.txt файл pro2.txt



**Функция пет:** Извеждане на филмите с тираж над 20 за продуцент, по име

продуцент pro1 продуцент pro2



**Функция шест:** Извеждане от файл, с подадено име като параметър, на всички данни за разпространителите

файл pro1.txt файл pro2.txt



**Функция седем:** Извеждане на разпространителите които предлагат филми от жанр екшън или драма, по година дадена като параметър



**Сорс Код**

#include<iostream>

#include<fstream>

#include<string>

#include<vector>

#include<list>

#include<algorithm>

#include<ctime>

#include<stdlib.h>

#pragma warning(disable: 4996);

using namespace std;

fstream fp;

time\_t t = time(NULL);

tm\* timePtr = localtime(&t);

// klas edno

class CFilms {

private:

string fname;

string producer;

int tiraj;

int year;

string genre;

public:

// Ustanovqvane na chlen promenlivite 1.

void setFilmName(const string s) { fname = s; }

void setProducer(const string s) { producer = s; }

void setGenre(const string s) { genre = s; }

void setTiraj(const int i) { tiraj = i; }

void setYear(const int i) { year = i; }

string getFilmName() const { return fname; }

string getProducer() const { return producer; }

string getGenre() const { return genre; }

int getTiraj() const { return tiraj; }

int getYear() const { return year; }

istream& input(istream& in) {

in >> fname;

in >> producer;

in >> genre;

in >> tiraj;

in >> year;

return in;

}

friend istream& operator>>(istream& in, CFilms& fil) { return fil.input(in); }

// funkciq za izvejdane v izhoden potok konzola 2.

ostream& outputConsole(ostream& out) {

out << fname << endl;

out << producer << endl;

out << genre << endl;

out << tiraj << endl;

out << year << endl;

return out;

}

friend ostream& operator<<(ostream& out, CFilms& fil) { return fil.outputConsole(out); }

// funkciq za izvejdane v izhoden potok fail 2.

void outputFile() {

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string line;

int count = 0;

if (fp.is\_open())

{

int inside\_count = 0;

while (getline(fp, line))

{

inside\_count++;

if (inside\_count == 5) {

count++;

inside\_count = 0;

}

}

fp.close();

}

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string fs1, fs2, fs3;

int fi1, fi2;

if (fp.is\_open()) {

int inside\_count = 0;

while (count != inside\_count) {

getline(fp, fs1);

getline(fp, fs2);

getline(fp, fs3);

getline(fp, line);

fi1 = stoi(line);

getline(fp, line);

fi2 = stoi(line);

CFilms fil(fs1, fs2, fs3, fi1, fi2);

cout << fil;

inside\_count++;

}

fp.close();

}

}

// podrazbirasht se konstruktor 1.

CFilms() {

fname = " ";

producer = " ";

genre = " ";

tiraj = 0;

year = 0;

}

// kopirasht konstruktor 2.

CFilms(const CFilms& fil) {

fname = fil.getFilmName();

producer = fil.getProducer();

genre = fil.getGenre();

tiraj = fil.getTiraj();

year = fil.getYear();

}

// ekspliciten konstruktor 3.

CFilms(string nam, string pro, string gen, int tir, int yer) {

fname = nam;

producer = pro;

genre = gen;

tiraj = tir;

year = yer;

}

// funkciq za validizirane na janra 1.

bool validateGenre(string s) {

return (genre == s) ? true : false;

}

// funkciq za izchislqvane na vyzrastta na filma 2.

int calculateAge() {

return timePtr->tm\_year - year;

}

// izvejdane na dannite za filma na printer 3.

void sendToPrint() {

ofstream print;

print.open("LPT1");

print << fname;

print << producer;

print << genre;

print << tiraj;

print << year;

print.close();

}

// operator za prizvoqvane = 4.

CFilms operator=(const CFilms& fil) {

fname = fil.fname;

producer = fil.producer;

genre = fil.genre;

tiraj = fil.tiraj;

year = fil.year;

return \*this;

}

// logicheski operator, koito proverqva dali filma e ot tazi godina 5.

bool operator!() { return (timePtr->tm\_year != year) ? true : false; }

// operator za sravnenie > (po tiraj) 6.

bool operator>(const CFilms& fil) { return (tiraj > fil.getTiraj()) ? true : false; }

// operator za syvpadenie == (po producenti) 7.

bool operator==(const CFilms& fil) { return (producer == fil.getProducer()) ? true : false; }

};

// klas dve

class CRazprostranitel {

private:

vector<CFilms> vfilm;

string rname;

double revenue;

public:

vector<CFilms> getCFilmVector() const { return vfilm; }

string getRazName() const { return rname; }

double getRevenue() const { return revenue; }

// syzdavane na obekt ot drug obekt 1.

CRazprostranitel(const CRazprostranitel& raz) {

vector<CFilms> vf = raz.getCFilmVector();

copy(vf.begin(), vf.end(), back\_inserter(vfilm));

rname = raz.getRazName();

revenue = raz.getRevenue();

}

// syzdavane na obekt chrez eksplicitno zadadeni parametri 2.

CRazprostranitel(vector<CFilms>& fil, string rnm, double rev) {

copy(fil.begin(), fil.end(), back\_inserter(vfilm));

rname = rnm;

revenue = rev;

}

CRazprostranitel() {

rname = " ";

revenue = 0.0;

}

// funkciq opredelqshta dali filma e predlagan poveche ot 1 pyt 3.

bool sheckIfMoreThenOne(CFilms& fil) { return (fil.getTiraj() > 1) ? true : false; }

istream& RInput(istream& in) {

in >> rname;

in >> revenue;

int siz = 0;

in >> siz;

for (int i = 0; i < siz; i++) {

CFilms fil;

in >> fil;

vfilm.push\_back(fil);

}

return in;

}

friend istream& operator>>(istream& in, CRazprostranitel& raz) { return raz.RInput(in); }

ostream& ROutput(ostream& out) {

out << rname << endl;

out << revenue << endl;

out << vfilm.size() << endl;

for (auto it = vfilm.begin(); it != vfilm.end(); it++) {

out << (\*it);

}

return out;

}

friend ostream& operator<<(ostream& out, CRazprostranitel& raz) { return raz.ROutput(out); }

};

// CASE 1

void makeCFilm(string& nm, string& pro, int& tir, int& yer, string& gen) {

system("cls");

fp.open("film.txt", ios::app);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

CFilms fil(nm, pro, gen, tir, yer);

fp << fil;

fp.close();

}

// CASE 2

void makeCRazprostranitel(vector<CRazprostranitel>& vr, string& rnm, double& rv, string vs[]) {

system("cls");

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string line;

int count = 0;

if (fp.is\_open()){

int inside\_count = 0;

while (getline(fp, line)){

inside\_count++;

if (inside\_count == 5) {

count++;

inside\_count = 0;

}

}

fp.close();

}

vector<CFilms> filmvector;

string fs1, fs2, fs3;

int fi1, fi2;

for (int i = 0; i < sizeof(vs); i++) {

string fnm = vs[i];

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

if (fp.is\_open()) {

int inside\_count = 0;

while (count != inside\_count) {

getline(fp, fs1);

getline(fp, fs2);

getline(fp, fs3);

getline(fp, line);

fi1 = stoi(line);

getline(fp, line);

fi2 = stoi(line);

CFilms fil(fs1, fs2, fs3, fi1, fi2);

if (fil.getFilmName() == fnm) {

filmvector.push\_back(fil);

break;

}

inside\_count++;

}

fp.close();

}

}

CRazprostranitel raz(filmvector, rnm, rv);

cout << raz;

vr.push\_back(raz);

system("pause");

}

// CASE 3

void listKomediq50k(list<CFilms>& k50k, string& kom) {

system("cls");

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string line;

int count = 0;

if (fp.is\_open()){

int inside\_count = 0;

while (getline(fp, line)){

inside\_count++;

if (inside\_count == 5) {

count++;

inside\_count = 0;

}

}

fp.close();

}

cout << "Filmi s tiraj nad 50 000 ot janr komediq: " << endl << endl;

fp.open("film.txt", ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string fs1, fs2, fs3;

int fi1, fi2;

if (fp.is\_open()) {

int inside\_count = 0;

while (count != inside\_count) {

getline(fp, fs1);

getline(fp, fs2);

getline(fp, fs3);

getline(fp, line);

fi1 = stoi(line);

getline(fp, line);

fi2 = stoi(line);

CFilms fil(fs1, fs2, fs3, fi1, fi2);

if (fil.getTiraj() > 50000 && fil.getGenre() == kom) {

cout << fil;

k50k.push\_back(fil);

}

inside\_count++;

}

fp.close();

}

system("pause");

}

//funkciq za CASE 4, za proverka na povtarqshti se imena na producenti

//s cel predotvratqvane na dublirashti s danni vyv faila

bool check(vector<string>& vs, string& s) {

bool flag = false;

for (auto i = vs.begin(); i != vs.end(); i++) {

if ((\*i) == s) {

return false;

}

flag = true;

}

return flag;

}

// CASE 4

void separateFilesByProducer(vector<CRazprostranitel>& vr) {

system("cls");

vector<string> vProd;

vProd.push\_back("empty");

string filename;

for (auto it = vr.begin(); it != vr.end(); it++) {

vector<CFilms> filmVector = (\*it).getCFilmVector();

for (auto itf = filmVector.begin(); itf != filmVector.end(); itf++) {

string comp = (\*itf).getProducer();

filename = comp + ".txt";

fp.open(filename, ios::out | ios::app);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

if (fp.is\_open()) {

CRazprostranitel raz((\*it));

string sPr = (\*itf).getProducer();

if (check(vProd, sPr) == true) {

fp << raz;

vProd.push\_back(sPr);

}

fp.close();

}

}

vProd.clear();

vProd.push\_back("Empty");

}

cout << "Failovete sa gotovi" << endl;

system("pause");

}

// CASE 5

void showFilmNames20(vector<CRazprostranitel>& vr, string& prod) {

system("cls");

cout << "Filmite s nad 20 broiki: " << endl;

for (auto it = vr.begin(); it != vr.end(); it++) {

vector<CFilms> filmVector = (\*it).getCFilmVector();

for (auto itf = filmVector.begin(); itf != filmVector.end(); itf++) {

if (prod == (\*itf).getProducer() && (\*itf).getTiraj() > 20) {

cout << "Film: " << (\*itf).getFilmName() << endl;

cout << "Razprostranitel: " << (\*it).getRazName() << endl << endl;

}

}

}

system("pause");

}

// CASE 6

void showRevFromFileName(string& filename) {

system("cls");

fp.open(filename, ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

string line;

int rCount = 0;

if (fp.is\_open()){

int inside\_count = 0;

int vSize = 0;

while (getline(fp, line)){

inside\_count++;

if (inside\_count == 3) {

vSize = stoi(line);

inside\_count = 0;

rCount++;

for (int i = 0; i < vSize \* 5; i++) {

getline(fp, line);

}

}

}

fp.close();

}

fp.open(filename, ios::in);

if (!fp) {

cout << endl << "\*\*\*\*\*\*\* ERROR \*\*\*\*\*\*\*";

exit(1);

}

if (fp.is\_open()) {

int inside\_count = 0;

while (rCount != inside\_count) {

int vSize = 0;

getline(fp, line);

cout << "Razprostranitel: " << line << endl;

getline(fp, line);

cout << "Pechalba: " << line << endl;

getline(fp, line);

cout << "Broi filmi: " << line << endl << endl;

vSize = stoi(line);

for (int i = 0; i < vSize \* 5; i++) {

getline(fp, line);

}

inside\_count++;

}

fp.close();

}

system("pause");

}

// CASE 7

void showRazprosNaEkshanDrama(vector<CRazprostranitel>& vr, int& yearch, string& ek, string& dr) {

system("cls");

bool flag = false;

cout << "Razprostranitelite s filmi za godina " << yearch << " ot janr ekshan ili drama sa: " << endl << endl;

for (auto it = vr.begin(); it != vr.end(); it++) {

vector<CFilms> filmVector = (\*it).getCFilmVector();

for (auto itf = filmVector.begin(); itf != filmVector.end(); itf++) {

if (yearch == (\*itf).getYear()) {

if (ek == (\*itf).getGenre()) {

cout << endl << "Razprostranitel:" << (\*it).getRazName() << endl;

cout << "Film: " << (\*itf).getFilmName() << endl;

cout << "Janr: " << (\*itf).getGenre() << endl << endl;

flag = true;

}

}

}

}

for (auto it = vr.begin(); it != vr.end(); it++) {

vector<CFilms> filmVector = (\*it).getCFilmVector();

for (auto itf = filmVector.begin(); itf != filmVector.end(); itf++) {

if (yearch == (\*itf).getYear()) {

if (dr == (\*itf).getGenre()) {

cout << endl << "Razprostranitel:" << (\*it).getRazName() << endl;

cout << "Film: " << (\*itf).getFilmName() << endl;

cout << "Janr: " << (\*itf).getGenre() << endl << endl;

flag = true;

}

}

}

}

if (flag == false) {

cout << "Nqma filmi za tazi godina" << endl;

}

system("pause");

}

// MAIN

void menu();

int main() {

menu();

return 0;

}

// MENU

void menu() {

vector<CRazprostranitel> vraz;

list<CFilms> komediq50k;

// CASE 1

string nm, pro, gen;

int tir, yer;

// CASE 2

string fnm, rnm;

double rv;

int broi;

string vs[100];

// CASE 3

string kom;

// CASE 4

// bez >>

// CASE 5

string prod;

// CASE 6

string filename;

// CASE 7

int yearch;

string ek, dr;

int ch;

do {

do {

system("cls");

cout << "==========MENU==========" << endl;

cout << "1. Suzdai Film" << endl;

cout << "2. Suzdai Razprostranitel" << endl;

cout << "3. Izvejdane v otdelen spisyk na filmite s tiraj nad 50 000 kopiq, ot janr komediq" << endl;

cout << "4. Da se izvedat v otdelni failove vsichki razprostraniteli, koito razprostranqvat filmi na edin i sysht producent" << endl;

cout << "5. Da se izvedat zaglaviqta na filmite, koito sa nad 20 za producent, po ime" << endl;

cout << "6. Da se izvedat vsichki danni za razprostranitelite s broi prihodi, kato parametyr, ot tekstov fail" << endl;

cout << "7. Pri daden kato parametyr godina da se izveat razprostranitelite na takiva filmi, ot janr ekshan ili drama" << endl;

cout << "8. Exit" << endl;

cout << endl << "Vyvedete izbora si: ";

cin >> ch;

} while (ch < 1 || ch>8);

switch (ch) {

case 1: system("cls");

cout << "Ime na filma: ";

cin.ignore(1, '\n');

getline(cin, nm);

cout << endl << "Producent: ";

getline(cin, pro);

cout << endl << "Tiraj: ";

cin >> tir;

cout << endl << "Godina: ";

cin >> yer;

cout << endl << "Janr: ";

cin.ignore(1, '\n');

getline(cin, gen);

makeCFilm(nm, pro, tir, yer, gen); break;

case 2: system("cls");

cout << "Ime na razprostranitelq: ";

cin.ignore(1, '\n');

getline(cin, rnm);

cout << endl << "Pechalba za godinata: ";

cin >> rv;

cout << endl << "Kolko filma iskate: ";

cin >> broi;

for (int i = 0; i < broi; i++) {

cout << endl << "Ime na film " << i + 1 << " koito shte vzeme: ";

cin >> fnm;

vs[i] = fnm;

}

makeCRazprostranitel(vraz, rnm, rv, vs);

for (int i = 0; i < broi; i++) {

vs[i] = "";

}

break;

case 3: system("cls");

cout << "Napishete kak ste vyveli janra komediq: ";

cin.ignore(1, '\n');

getline(cin, kom);

listKomediq50k(komediq50k, kom); break;

case 4: system("cls");

separateFilesByProducer(vraz); break;

case 5: system("cls");

cout << "Napishete imeto na producenta za proverka: ";

cin.ignore(1, '\n');

getline(cin, prod);

showFilmNames20(vraz, prod); break;

case 6: system("cls");

cout << "Napishete imeto na faila koito iskate da otvorite: ";

cin.ignore(1, '\n');

getline(cin, filename);

showRevFromFileName(filename); break;

case 7: system("cls");

cout << "Vyvedete godina na film: ";

cin >> yearch;

cout << "Napishete kak ste vyveli janra ekshan: ";

cin.ignore(1, '\n');

getline(cin, ek);

cout << "Napishete kak ste vyveli janra drama: ";

getline(cin, dr);

showRazprosNaEkshanDrama(vraz, yearch, ek, dr); break;

default: system("cls"); cout << "==== END ====";

}

} while (ch != 8);

}