Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение

высшего образования

«**Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные

системы»

**ОТЧЁТ**

**по лабораторной работе №8**

Дисциплина: «Информатика»

Тема: Структуры данных

Вариант 17

Выполнил:

Студент группы ИВТ-20-2б

Сафронов Владислав Владиславович

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

**Пермь, 2021**

**Цель задачи**

Работа с двоичными файлами, организация ввода-вывода структурированной информации и ее хранение на внешних носителях.

**Постановка задачи**

Сформировать двоичный файл из элементов, заданной в варианте структуры, распечатать его содержимое, выполнить удаление и добавление элементов в соответствии со своим вариантом, используя для поиска удаляемых или добавляемых элементов функцию. Формирование, печать, добавление и удаление элементов оформить в виде функций. Предусмотреть сообщения об ошибках при открытии файла и выполнении операций ввода/вывода.

Структура "Фильм":

- название;

- режиссер;

- год выпуска;

- стоимость.

Удалить все элементы, у которых стоимость превышает заданную, добавить элемент в начало файла.

**Анализ задачи**

**Какие типы данных будут использованы:**

* Целочисленный тип integer для хранения введенных данных для количества элементов в массиве структур, стоимости и полей year и cost в структуре Film;
* Строковый тип данных string;
* Массив структур films\_arr;

**Какие с этими данными выполняются действия:**

* string используется для хранения полей name и director в структуре Film, сравнения (проверка, что строка непустая)
* integer используется для хранения вводимых данных, сравнения (проверка на то, что вводимое число соответствует требованиям)
* Массив структур films\_arr используется для хранения элементов структуры Film удаления и добавления элементов структуры;

**В каком виде эти данные будут представлены:**

1. Структура Films содержит 4 поля: 2 строковых поля (name, director) и 2 целочисленных поля (year, cost).

struct Film

{

string name, director;

int year, cost;

Также структура имеет 2 метода:

* Метод Fill() для заполнения полей структуры;
* Метод Print() для вывода полей структуры в консоль

void Print()

{

cout << "\n";

cout << "Название:\t" << name << "\n";

cout << "Режиссер:\t" << director << "\n";

cout << "Год выпуска:\t" << year << "\n";

cout << "Стоимость:\t" << cost << "\n";

}

1. Функция RemoveExcess для удаления из массива структур элементы, цена которых превышает заданную. В качестве параметров указывается ссылка на указатель на первый элемент массива структур, ссылка на переменную, хранящую размер массива, и переменная cost, которая хранит пороговую стоимость.

void RemoveExcess(Film\* & arr, int& size, int cost)

Сначала в цикле определяется длинна нового массива

int new\_size = size;

for (int i = 0; i < size; i++)

{

if (arr[i].cost > cost) new\_size--;

}

Если длинна нового массива больше нуля, то создется новый массив из элементов, стоимость которых не превышает заданную

if (new\_size == 0)

{

delete[] arr;

arr = nullptr;

size = 0;

}

else

{

Film\* new\_arr = new Film[new\_size];

new\_size = 0;

for (int i = 0; i < size; i++)

{

if (arr[i].cost <= cost)

{

new\_arr[new\_size] = arr[i];

new\_size++;

}

}

delete[] arr;

arr = new\_arr;

size = new\_size;

new\_arr = nullptr;

}

1. Функция AppendToList для добавления элемента структуры в начало массива. В качестве параметров указывается ссылка на указатель на первый элемент массива структур, ссылка на переменную, хранящую размер массива. В теле функции создается новый массив, длинной на 1 единицу больше существующего. Затем с помощью метода Fill() заполняется первый элемент массива. Затем в случае, если старый массив был не пустой, то в новый массив копируются элементы из старого.

void AppendToList(Film\*& arr, int& size)

{

size++;

Film\* new\_arr = new Film[size];

new\_arr[0].Fill();

if (size > 1)

{

for (int i = 1; i < size; i++)

{

new\_arr[i] = arr[i - 1];

}

delete[] arr;

arr = new\_arr;

new\_arr = nullptr;

}

else

{

arr = new\_arr;

new\_arr = nullptr;

}

}

1. Функция PrintList, которая выводит массив структур в консоль

void PrintList(Film\* arr, int size)

{

if (size == 0) cout << "Список пуст\n";

else

{

for (int i = 0; i < size; i++)

{

arr[i].Print();

}

}

}

1. Функция AddToFile для вывода элементов структуры в файл. В качестве параметров функция принимает ссылку на подключенный файл ofstream, указатель на первый элемент массива структур и переменную, хранящую размер массива.

void AddToFile(ofstream& file, Film\* arr, int size)

{

for (int i = 0; i < size; i++)

{

file << "Название:\t" << arr[i].name << endl;

file << "Режиссер:\t" << arr[i].director << endl;

file << "Год выпуска:\t" << arr[i].year << endl;

file << "Стоимость:\t" << arr[i].cost << endl << endl;

}

}

1. Файл “Filmoteka.txt”, в который выводится полученный массив структур.

**Какими операторами будет организован ввод и вывод:**

* Ввод данных осуществляется через консоль с помощью функций cin и getline().
* Вывод данных в консоль осуществляется с помощью функции cout
* Вывод данных в файл используется метод ofstream из библиотеки <fstream>

**Описание функции main():**

Подключение русского языка:

setlocale(LC\_ALL, "Russian");

system("chcp 1251>nul");

Описание переменных num и cost\_del и инициализация строки, хранящей имя файла для вывода:

int num;

int cost\_del;

string filename = "Filmoteka.txt";

Пользователь вводит количество элементов структуры:

cout << "Введите количество фильмов: "; cin >> num;

while (num <= 0)

{

cout << "Неверно введено количество фильмов.\n";

cout << "Введите количество фильмов: "; cin >> num;

}

cin.get();

Создается массив структур с заданной длиной, и пользователь заполняет каждый элемент структуры:

Film\* films\_arr = new Film[num];

for (int i = 0; i < num; i++)

{

films\_arr[i].Fill();

}

Полученный массив печатается в консоль функцией PrintList:

cout << "\nИсходная фильмотека:\n";

PrintList(films\_arr, num);

Пользователь вводит предельную стоимость, затем с помощью функции RemoveExcess из массива удаляются элементы, стоимость которых превышает введенную:

cout << "\nВведите предельную стоимость: ";

cin >> cost\_del;

while (cost\_del <= 0)

{

cout << "Введена некорректная стоимость.\n";

cout << "\nВведите предельную стоимость: ";

cin >> cost\_del;

}

cin.get();

cout << "\nВсе фильмы, превышающие стоймость " << cost\_del << " будут удалены.";

RemoveExcess(films\_arr, num, cost\_del);

Полученный массив выводится на экран:

cout << "\nНовая фильмотека: \n";

PrintList(films\_arr, num);

С помощью функции AppendList в начало массива добавляется новый элемент структуры и полученный массив выводится в консоль:

cout << "\nДобавление фильма в начало списка.\n";

AppendToList(films\_arr, num);

cout << "\nНовая фильмотека:\n";

PrintList(films\_arr, num);

С помощью метода ofstream из библиотеки <fstream> и функции AddToFile полученный массив элементов структуры выводится в файл “Filmoteka.txt”:

ofstream output(filename);

AddToFile(output, films\_arr, num);

output.close();

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

struct Film

{

string name, director;

int year, cost;

void Fill()

{

cout << "\nВведите название фильма: ";

getline(cin, name);

while (name == "" || name[0] == ' ')

{

cout << "Строка не может быть пустой и не может начинаться с пробела.\n";

cout << "Введите название фильма: ";

getline(cin, name);

}

cout << "Введите имя режиссера: ";

getline(cin, director);

while (director == "" || director[0] == ' ')

{

cout << "Строка не может быть пустой и не может начинаться с пробела.\n";

cout << "Введите имя режиссера: ";

getline(cin, director);

}

cout << "Введите год: "; cin >> year;

while (year > 2021 || year < 1895)

{

if (year < 1895)

{

cout << "Первый фильм был снят только в 1895 году.\n";

cout << "Введите год: "; cin >> year;

}

else

{

cout << "Можно добавлять только уже вышедшие фильмы.\n";

cout << "Введите год: "; cin >> year;

}

}

cout << "Введите стоимость: "; cin >> cost;

while (cost <= 0)

{

cout << "Стоимость введена некорректно.\n";

cout << "Введите стоимость: "; cin >> cost;

}

cin.get();

}

void Print()

{

cout << "\n";

cout << "Название:\t" << name << "\n";

cout << "Режиссер:\t" << director << "\n";

cout << "Год выпуска:\t" << year << "\n";

cout << "Стоимость:\t" << cost << "\n";

}

};

void AddToFile(ofstream& file, Film\* arr, int size)

{

for (int i = 0; i < size; i++)

{

file << "Название:\t" << arr[i].name << endl;

file << "Режиссер:\t" << arr[i].director << endl;

file << "Год выпуска:\t" << arr[i].year << endl;

file << "Стоимость:\t" << arr[i].cost << endl << endl;

}

}

void RemoveExcess(Film\*& arr, int& size, int cost)

{

int new\_size = size;

for (int i = 0; i < size; i++)

{

if (arr[i].cost > cost) new\_size--;

}

if (new\_size == 0)

{

delete[] arr;

arr = nullptr;

size = 0;

}

else

{

Film\* new\_arr = new Film[new\_size];

new\_size = 0;

for (int i = 0; i < size; i++)

{

if (arr[i].cost <= cost)

{

new\_arr[new\_size] = arr[i];

new\_size++;

}

}

delete[] arr;

arr = new\_arr;

size = new\_size;

new\_arr = nullptr;

}

}

void AppendToList(Film\*& arr, int& size)

{

size++;

Film\* new\_arr = new Film[size];

new\_arr[0].Fill();

if (size > 1)

{

for (int i = 1; i < size; i++)

{

new\_arr[i] = arr[i - 1];

}

delete[] arr;

arr = new\_arr;

new\_arr = nullptr;

}

else

{

arr = new\_arr;

new\_arr = nullptr;

}

}

void PrintList(Film\* arr, int size)

{

if (size == 0) cout << "Список пуст\n";

else

{

for (int i = 0; i < size; i++)

{

arr[i].Print();

}

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

system("chcp 1251>nul");

int num;

int cost\_del;

string filename = "Filmoteka.txt";

cout << "Введите количество фильмов: "; cin >> num;

while (num <= 0)

{

cout << "Неверно введено количество фильмов.\n";

cout << "Введите количество фильмов: "; cin >> num;

}

cin.get();

Film\* films\_arr = new Film[num];

for (int i = 0; i < num; i++)

{

films\_arr[i].Fill();

}

cout << "\nИсходная фильмотека:\n";

PrintList(films\_arr, num);

cout << "\nВведите предельную стоимость: ";

cin >> cost\_del;

while (cost\_del <= 0)

{

cout << "Введена некорректная стоимость.\n";

cout << "\nВведите предельную стоимость: ";

cin >> cost\_del;

}

cin.get();

cout << "\nВсе фильмы, превышающие стоймость " << cost\_del << " будут удалены.";

RemoveExcess(films\_arr, num, cost\_del);

cout << "\nНовая фильмотека: \n";

PrintList(films\_arr, num);

cout << "\nДобавление фильма в начало списка.\n";

AppendToList(films\_arr, num);

cout << "\nНовая фильмотека:\n";

PrintList(films\_arr, num);

ofstream output(filename);

AddToFile(output, films\_arr, num);

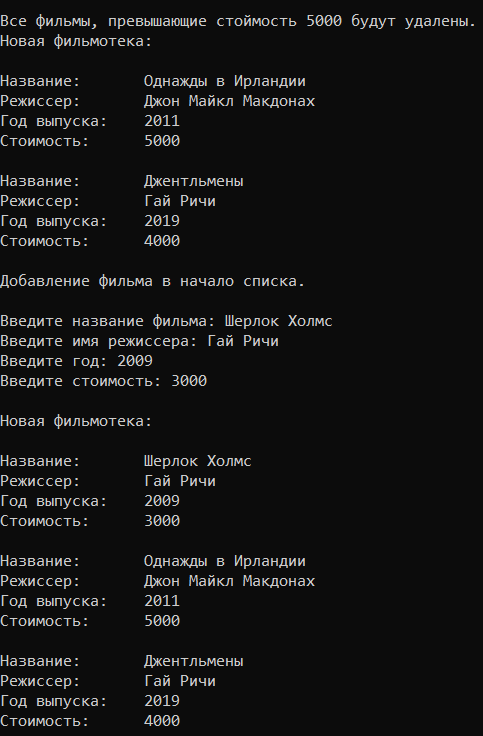
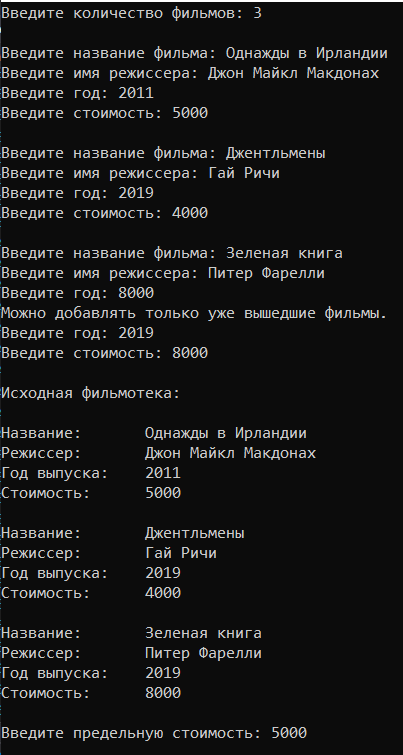
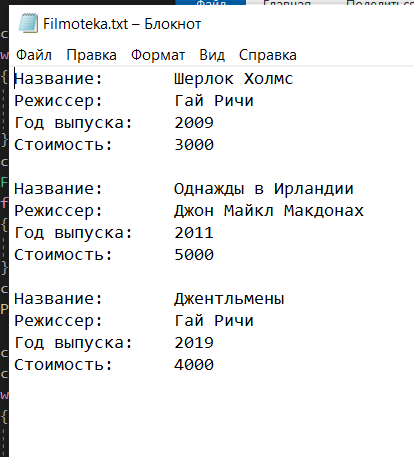
output.close();

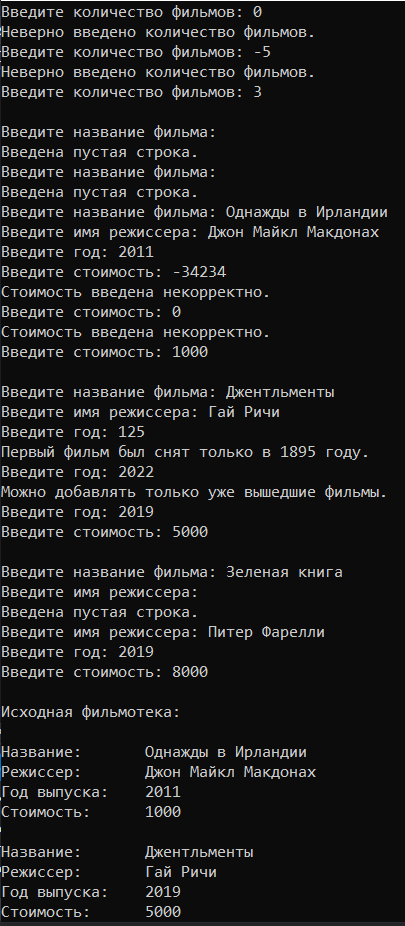
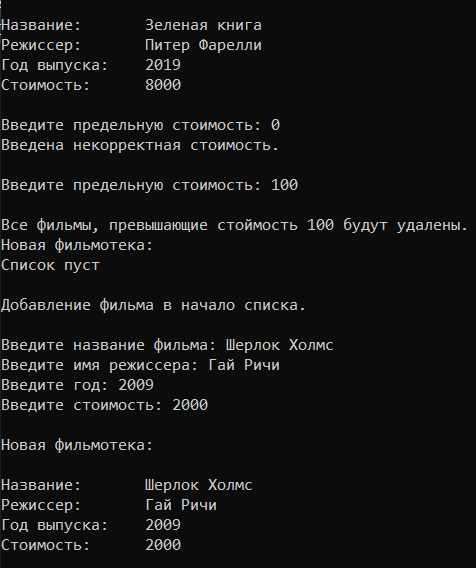
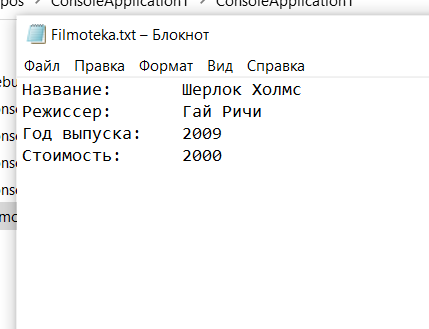
delete[] films\_arr;

return 0;

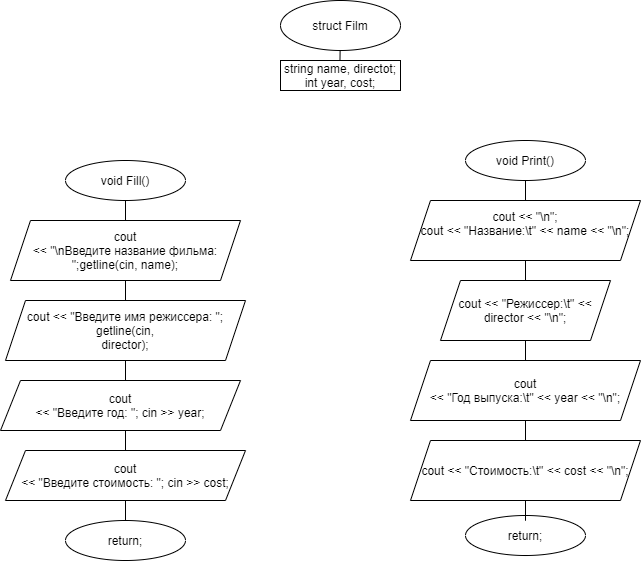
}

**Скриншоты**

****

****

**Блок-схема**

****

