Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторная работа № 8**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

**(Семестр** 2)

Выполнил студент гр. ИВТ-21-1б

Гребнев Алексей Дмитриевич

Проверил:

Яруллин Денис Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г.Пермь -2022

**Постановка задачи**

Сформировать двоичный файл из элементов, заданной в варианте структуры, распечатать его содержимое, выполнить удаление и добавление элементов в соответствии со своими 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.

**Исходные данные**

Вариант 10:

1) Носитель

2) Объём

3) Название

4) Автор

Удалить первый элемент с заданным объёмом информации, добавить элемент перед элементом с указанным номером

#include<iostream>

#include <stdio.h>

#include <fstream>

using namespace std;

//Структура для компановки информации

struct info

{

char name[20];

int V;

char autor[15];

char nositel[15];

};

int Index(int DefV, info\* kempl)

{

for (int i = 0; i < 4; i++)

{

if (kempl[i].V == DefV)

{

return i;

}

}

}

int main()

{

int DefV = 35;

int index;

setlocale(LC\_ALL, "Russian");

info kempl[4];

//Создаём блоки с информацияей

kempl[0] = { "Джонс ", 35, "Гендрикс Мл ", "Иван" };

kempl[1] = { "Кельвин По ", 43 , " Толстовин ", "Мотя" };

kempl[2] = { "Солнце ", 43, " Людин ", "Вова" };

kempl[3] = { "Лунный свет ", 54, " Риккенс ", "Дима" };

cout << endl;

ofstream fout;

info\* quert = new info[4];

index = Index(DefV, kempl);

//Кейс для подстановки нового блока по индексу

switch (index)

{

case 3:

quert[0] = kempl[0];

quert[1] = kempl[1];

quert[2] = { "Ночь в точке", 34, " Ригеранн ", "Влад" };

quert[3] = kempl[2];

fout.open("myFile.txt");

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 3; i++)

{

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "Название: " << quert[i].name << "\n";

fout << "Объём: " << quert[i].V << "\n";

fout << "Автор: " << quert[i].autor << "\n";

fout << "Носитель: " << quert[i].nositel << "\n";

fout << "\n";

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout.close();

break;

case 2:

quert[0] = kempl[0];

quert[1] = { "Ночь в точке", 34, " Ригеранн ", "Влад" };

quert[2] = kempl[1];

quert[3] = kempl[3];

fout.open("myFile.txt");

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 4; i++)

{

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "Название: " << quert[i].name << "\n";

fout << "Объём: " << quert[i].V << "\n";

fout << "Автор: " << quert[i].autor << "\n";

fout << "Носитель: " << quert[i].nositel << "\n";

fout << "\n";

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout.close();

break;

case 1:

quert[0] = { "Ночь в точке", 34, " Ригеранн ", "Влад" };

quert[1] = kempl[0];

quert[2] = kempl[2];

quert[3] = kempl[3];

fout.open("myFile.txt");

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 4; i++)

{

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "Название: " << quert[i].name << "\n";

fout << "Объём: " << quert[i].V << "\n";

fout << "Автор: " << quert[i].autor << "\n";

fout << "Носитель: " << quert[i].nositel << "\n";

fout << "\n";

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout.close();

break;

case 0:

quert[0] = { "Ночь в точке", 34, " Ригеранн ", "Влад" };

quert[1] = kempl[1];

quert[2] = kempl[2];

quert[3] = kempl[3];

fout.open("myFile.txt");

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 4; i++)

{

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "Название: " << quert[i].name << "\n";

fout << "Объём: " << quert[i].V << "\n";

fout << "Автор: " << quert[i].autor << "\n";

fout << "Носитель: " << quert[i].nositel << "\n";

fout << "\n";

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout.close();

break;

default:

quert[0] = kempl[0];

quert[1] = kempl[1];

quert[2] = kempl[2];

quert[3] = kempl[3];

fout.open("myFile.txt");

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

for (int i = 0; i < 4; i++)

{

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "Название: " << quert[i].name << "\n";

fout << "Объём: " << quert[i].V << "\n";

fout << "Автор: " << quert[i].autor << "\n";

fout << "Носитель: " << quert[i].nositel << "\n";

fout << "\n";

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout.close();

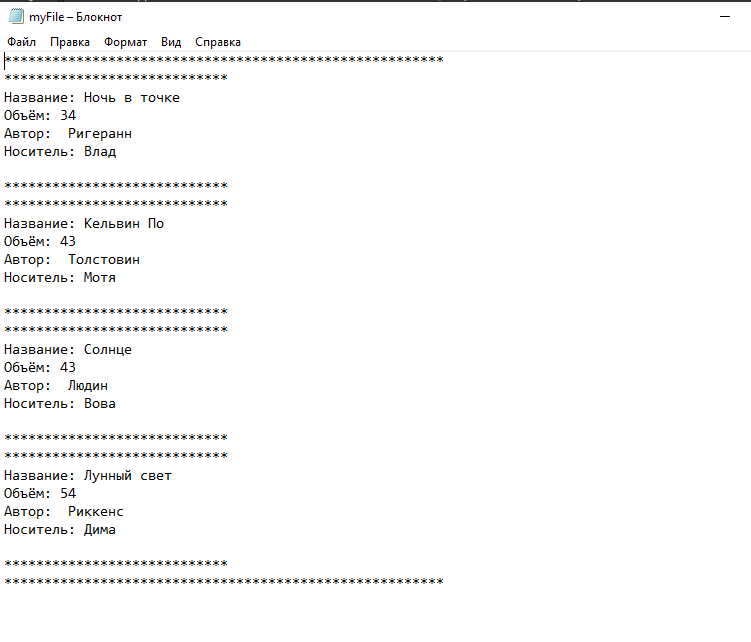
}

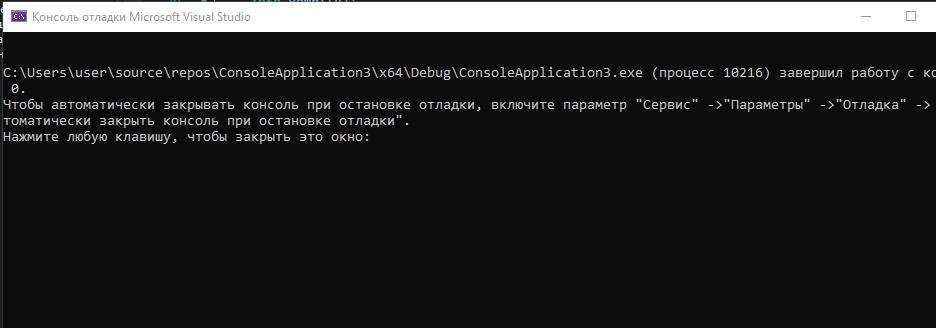
delete[] quert;

return 0;

}

**Результат**





**Блок-схема**

