Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Электротехнический факультет

Кафедра **«**Информационные технологии и автоматизированные системы**»**

направление подготовки: 09.03.04 – **«**Программная инженерия»

**Лабораторная работа №8.**

**“Функции и массивы”  
Вариант 10**

Выполнил студент гр. РИС-24-2б

Бартов Игорь Сергеевич

Проверил:

Доц. каф. ИТАС

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

г. Пермь, 2025

1. **Постановка задачи:**

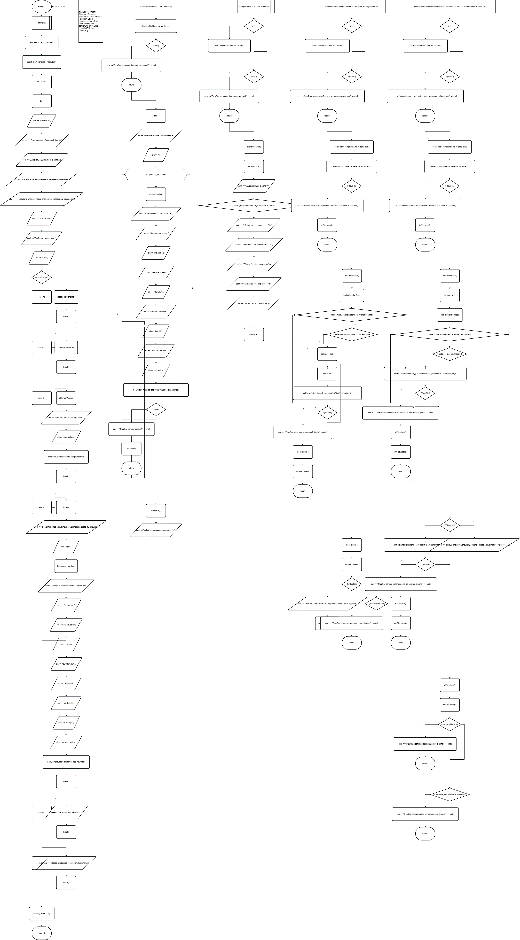
Сформировать двоичный файл из элементов, заданной в варианте структуры, распечатать его содержимое, выполнить удаление и добавление элементов в соответствии со своим вариантом, используя для поиска удаляемых или добавляемых элементов функцию. Формирование, печать, добавление и удаление элементов оформить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

Структура "Информация":

* носитель;
* объем;
* название;
* автор.

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

**2. Блок-схема:**

****

1. **Код:**

#include <iostream>

#include <fstream>

#include <cstdio>

#include <clocale>

using namespace std;

struct Information {

char carrier[21];

int volume;

char title[51];

char author[51];

};

void createFile(const char\* filename) {

ofstream file(filename, ios::binary);

if (!file) {

cerr << "Ошибка открытия файла для записи!" << endl;

return;

}

int n;

cout << "Введите количество элементов: ";

cin >> n;

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

Information info;

cout << "\nЭлемент " << i + 1 << ":\n";

cout << "Введите носитель: ";

cin >> info.carrier;

cout << "Введите объем: ";

cin >> info.volume;

cout << "Введите название: ";

cin >> info.title;

cout << "Введите автора: ";

cin >> info.author;

file.write(reinterpret\_cast<char\*>(&info), sizeof(info));

if (!file) {

cerr << "Ошибка записи в файл!" << endl;

file.close();

return;

}

}

file.close();

cout << "\nФайл сформирован успешно.\n";

}

void printFile(const char\* filename) {

ifstream file(filename, ios::binary);

if (!file) {

cerr << "Ошибка открытия файла для чтения!" << endl;

return;

}

Information info;

int index = 1;

cout << "\nСодержимое файла:\n";

while (file.read(reinterpret\_cast<char\*>(&info), sizeof(info))) {

cout << "\nЭлемент " << index++ << ":\n";

cout << "Носитель: " << info.carrier << "\n";

cout << "Объем: " << info.volume << "\n";

cout << "Название: " << info.title << "\n";

cout << "Автор: " << info.author << "\n";

}

file.close();

}

void deleteByVolume(const char\* filename, int targetVolume) {

ifstream inFile(filename, ios::binary);

if (!inFile) {

cerr << "Ошибка открытия файла для чтения при удалении!" << endl;

return;

}

const char\* tempFilename = "temp.bin";

ofstream outFile(tempFilename, ios::binary);

if (!outFile) {

cerr << "Ошибка открытия временного файла для записи!" << endl;

inFile.close();

return;

}

Information info;

bool deleted = false;

while (inFile.read(reinterpret\_cast<char\*>(&info), sizeof(info))) {

if (!deleted && info.volume == targetVolume) {

deleted = true;

continue;

}

outFile.write(reinterpret\_cast<char\*>(&info), sizeof(info));

if (!outFile) {

cerr << "Ошибка записи во временный файл!" << endl;

inFile.close();

outFile.close();

return;

}

}

inFile.close();

outFile.close();

if (!deleted) {

cout << "Элемент с объемом " << targetVolume << " не найден.\n";

remove(tempFilename);

}

else {

if (remove(filename) != 0) {

cerr << "Ошибка удаления оригинального файла!" << endl;

return;

}

if (rename(tempFilename, filename) != 0) {

cerr << "Ошибка переименования временного файла!" << endl;

return;

}

cout << "Элемент с объемом " << targetVolume << " удален.\n";

}

}

void addElementBefore(const char\* filename, int pos, const Information& newInfo) {

ifstream inFile(filename, ios::binary);

if (!inFile) {

cerr << "Ошибка открытия файла для чтения при добавлении!" << endl;

return;

}

const char\* tempFilename = "temp.bin";

ofstream outFile(tempFilename, ios::binary);

if (!outFile) {

cerr << "Ошибка открытия временного файла для записи!" << endl;

inFile.close();

return;

}

Information info;

int index = 1;

bool inserted = false;

while (inFile.read(reinterpret\_cast<char\*>(&info), sizeof(info))) {

if (index == pos && !inserted) {

outFile.write(reinterpret\_cast<const char\*>(&newInfo), sizeof(newInfo));

if (!outFile) {

cerr << "Ошибка записи нового элемента во временный файл!" << endl;

inFile.close();

outFile.close();

return;

}

inserted = true;

}

outFile.write(reinterpret\_cast<char\*>(&info), sizeof(info));

if (!outFile) {

cerr << "Ошибка записи в временный файл!" << endl;

inFile.close();

outFile.close();

return;

}

index++;

}

if (!inserted) {

outFile.write(reinterpret\_cast<const char\*>(&newInfo), sizeof(newInfo));

if (!outFile) {

cerr << "Ошибка записи нового элемента в конец файла!" << endl;

inFile.close();

outFile.close();

return;

}

cout << "Позиция не найдена. Новый элемент добавлен в конец файла.\n";

}

else {

cout << "Новый элемент успешно добавлен перед элементом " << pos << ".\n";

}

inFile.close();

outFile.close();

if (remove(filename) != 0) {

cerr << "Ошибка удаления оригинального файла!" << endl;

return;

}

if (rename(tempFilename, filename) != 0) {

cerr << "Ошибка переименования временного файла!" << endl;

return;

}

}

int main() {

setlocale(LC\_ALL, "Russian");

const char\* filename = "data.bin";

int choice;

do {

cout << "\nМеню:\n";

cout << "1. Сформировать бинарный файл\n";

cout << "2. Печать содержимого файла\n";

cout << "3. Удалить первый элемент с заданным объемом\n";

cout << "4. Добавить элемент перед элементом с заданным номером\n";

cout << "0. Выход\n";

cout << "Выберите операцию: ";

cin >> choice;

switch (choice) {

case 1:

createFile(filename);

break;

case 2:

printFile(filename);

break;

case 3: {

int targetVolume;

cout << "Введите объем для удаления: ";

cin >> targetVolume;

deleteByVolume(filename, targetVolume);

break;

}

case 4: {

int pos;

cout << "Введите номер элемента, перед которым добавить новый: ";

cin >> pos;

Information newInfo;

cout << "Введите данные нового элемента:\n";

cout << "Носитель: ";

cin >> newInfo.carrier;

cout << "Объем: ";

cin >> newInfo.volume;

cout << "Название: ";

cin >> newInfo.title;

cout << "Автор: ";

cin >> newInfo.author;

addElementBefore(filename, pos, newInfo);

break;

}

case 0:

cout << "Выход из программы.\n";

break;

default:

cout << "Неверный выбор! Повторите попытку.\n";

break;

}

} while (choice != 0);

return 0;

}