

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА - Российский технологический университет"**

**РТУ МИРЭА**

Институт информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2**

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

Тема. Внешние структуры данных: текстовый и двоичный файлы

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы ИКБО-04-22 |  | Кликушин В.И. |
| Принял старший преподаватель |  | Скворцова Л.А. |

Москва 2023

ОГЛАВЛЕНИЕ

[1 ЗАДАНИЕ №1 3](#_Toc147422365)

[1.1 УСЛОВИЕ ЗАДАЧИ И ЗАДАНИЕ ВАРИАНТА 3](#_Toc147422366)

[1.1.1 Формулировка 3](#_Toc147422367)

[1.1.2 Требования 3](#_Toc147422368)

[1.2 ТЕСТОВЫЙ ПРИМЕР 4](#_Toc147422369)

[1.3 РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ 4](#_Toc147422370)

[1.3.1 Функционал приложения 4](#_Toc147422371)

[1.3.2 Код основной программы 5](#_Toc147422372)

[1.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 13](#_Toc147422373)

[2 ЗАДАНИЕ №2 17](#_Toc147422374)

[2.1 УСЛОВИЕ ЗАДАЧИ 17](#_Toc147422375)

[2.1.1 Формулировка условия 17](#_Toc147422376)

[2.1.2 Требования задания 17](#_Toc147422377)

[2.1.3 Формулировка задания варианта 18](#_Toc147422378)

[2.2 ТЕСТОВЫЙ ПРИМЕР 19](#_Toc147422379)

[2.3 РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ 20](#_Toc147422380)

[2.3.1 Структура записи двоичного файла (из кода) и ее размер в байтах (ручной расчет и системный). 20](#_Toc147422381)

[2.3.2 Изображение структуры двоичного файла 21](#_Toc147422382)

[2.3.3 Функционал приложения 21](#_Toc147422383)

[2.3.4 Код основной программы 22](#_Toc147422384)

[2.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 33](#_Toc147422385)

[3 ВЫВОДЫ 38](#_Toc147422386)

[4 ИСПОЛЬЗУЕМЫЕ ИСТОЧНИКИ 39](#_Toc147422387)

# 1 ЗАДАНИЕ №1

# 1.1 УСЛОВИЕ ЗАДАЧИ И ЗАДАНИЕ ВАРИАНТА

## 1.1.1 Формулировка

* Разработать программу управления текстовым файлом, используя файловые потоки ifstream и ofstream. Создать новый файл из значений исходного, поделив каждое число на наибольший общий делитель чисел файла.

## 1.1.2 Требования

1. Реализация ввода-вывода на основе файловых потоков С++: ofstream, ifstream.

2. Имя физического файла вводится пользователем и передается в функции обработки через параметр.

3. При открытии файла выполнять контроль его существования и открытия.

4. Разработать функции для выполнения операций над текстовым файлом. 1) создание текстового файла средствами текстового редактора кодировки ASCII, содержащего десятичные числа по несколько чисел на строке; 2) вывод содержимого текстового файла; 3) добавление новой записи в конец файла; 4) прочитать значение числа, указав его порядковый номер в файле, и вернуть его значение; 5) определить количество чисел в файле.

5. Разработать приложение и выполнить тестирование всех функций. Приложение должно содержать диалоговый интерфейс на основе текстового меню.

6. Контроль открытия и существования файла выполнить в основной программе перед вызовом функции. Перед закрытием файла, проверить отсутствие ошибок ввода.

7. Создать модуль и перенести в него все отлаженные функции. Исключить функции из приложения. Отладить приложение, подключив к нему модуль с функциями.

8. Разработать функции для реализации дополнительных операций, определенных вариантом и сохранить их в модуле с остальными функциями.

9. Выполнить тестирование приложения в полном объеме.

# 1.2 ТЕСТОВЫЙ ПРИМЕР

Содержимое файла, который является тестовым примером представлено на рисунке №1.1.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 1.1 – Тестовый пример для задания №1

# 1.3 РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ

## 1.3.1 Функционал приложения

Операции

1. Создание и заполнение текстового файла

Предусловие. file\_name – строка, название файла, объект класса ofstream, передаваемый по ссылке.

Постусловие. Булево значение – код работы метода.

**bool create\_file(string file\_name, ofstream &input);**

1. Вывод содержимого текстового файла

Предусловие. file\_name – строка, название файла, объект класса ifstream, передаваемый по ссылке.

Постусловие. Булево значение – код работы метода.

**bool output\_file(string file\_name, ifstream& output);**

1. Добавление новой записи в конец текстового файла

Предусловие. file\_name – строка, название файла, объект класса ofstream, передаваемый по ссылке, целое число, добавляемое в файл.

Постусловие. Нет возвращаемого значения.

**void add\_to\_file(string file\_name, ofstream& input, int x);**

1. Получение значения числа по индексу

Предусловие. file\_name – строка, название файла, объект класса ifstream, передаваемый по ссылке, целое число, индекс.

Постусловие. Целочисленное значение по индексу, -1, если ошибочный иднекс.

**int get\_number(string file\_name, ifstream& output, int index);**

1. Определение количества чисел в файле

Предусловие. file\_name – строка, название файла, объект класса ifstream, передаваемый по ссылке.

Постусловие. Целочисленное значение, соответствующее количеству чисел в файле.

**int get\_total(string file\_name, ifstream& output);**

1. Создание нового файла из значений исходного, поделив каждое число на наибольший общий делитель чисел файла.

Предусловие. file\_name – строка, название файла, new\_file – строка, название нового файла, объект класса ifstream, передаваемый по ссылке, целочисленное значение – количество чисел, целочисленное значение - НОД.

Постусловие. Булево значение – код работы метода.

**bool create\_new\_file(ifstream& output, string file\_name, string new\_file,int nod,int n);**

1. Определение НОД чисел в файле

Предусловие. file\_name – строка, название файла.

Постусловие. Целочисленное значение, соответствующее НОД чисел в файле.

**int get\_nod(string file\_name);**

## 1.3.2 Код основной программы

Код файла FileFuncs.h:

#pragma once

#include <iostream>

#include <string>

#include <fstream>

#include <vector>

using namespace std;

bool create\_file(string file\_name,ofstream &input);

bool output\_file(string file\_name, ifstream& output);

void add\_to\_file(string file\_name, ofstream& input, int x);

int get\_number(string file\_name, ifstream& output, int index);

int get\_total(string file\_name, ifstream& output);

bool create\_new\_file(ifstream& output, string file\_name, string new\_file,int nod,int n);

int get\_nod(string file\_name);

Код файла FileFuncs.cpp:

#include "FileFuncs.h"

#include <string>

bool create\_file(string file\_name, ofstream &input)

{

int n,m,x;

cout << " Введите количество строк для записи чисел в файл: ";

cin >> n;

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

{

cout << " Введите количество чисел в " << i+1 << "-ой строке: ";

cin >> m;

for (int j = 0; j < m; j++)

{

cin >> x;

input << x;

if (j!=m-1)

input << " ";

}

if (i!=n-1)

input << '\n';

}

return 0;

}

bool output\_file(string file\_name, ifstream& output)

{

int x;

cout << " Содержимое файла:";

while (!output.eof())

{

output >> x;

cout << " " << x;

}

cout << endl;

return 0;

}

void add\_to\_file(string file\_name, ofstream& input, int x)

{

input << " ";

input << x;

}

int get\_number(string file\_name, ifstream& output, int index)

{

int x;

int i = 0;

int res = -1;

while (!output.eof() || !(i == index))

{

output >> x;

output.get();

if (i==index)

{

res = x;

return res;

}

i++;

}

return res;

}

int get\_total(string file\_name, ifstream& output)

{

int total = 0;

int x;

while (!output.eof())

{

output >> x;

total++;

}

return total;

}

bool create\_new\_file(ifstream& output, string file\_name,string new\_file,int nod,int n)

{

ofstream newf;

newf.open(new\_file);

int x;

int i = 1;

while (!output.eof())

{

output >> x;

newf << (double)x/nod;

if (i < n)

newf << "\n";

i++;

}

return 0;

}

int get\_nod(string file\_name)

{

ifstream output(file\_name);

if (!output)

{

return -1;

}

else

{

vector <int> data;

int x;

while (!output.eof())

{

output >> x;

data.push\_back(x);

}

if (data.size() == 0)

{

return -1;

}

if (data.size() == 1)

return data[0];

while (data[0] && data[1])

if (data[0] > data[1]) data[0] %= data[1];

else data[1] %= data[0];

int res = data[1] + data[0];

for (auto i = 2; i < data.size(); i++)

{

while (res && data[i])

if (res > data[i]) res %= data[i];

else data[i] %= res;

res = data[i] + res;

}

output.close();

return res;

}

}

Код файла Main.cpp:

#include "FileFuncs.h";

int main()

{

int num;

string file\_name;

string new\_file;

ofstream input;

ifstream output;

system("chcp 1251");

while (true)

{

cout << "-----------------------------------------------------------------------------------------------------------" << endl;

cout << " Меню\n";

cout << " 1. Создать и заполнить текстовый файл\n";

cout << " 2. Вывести содержимое текствого файла\n";

cout << " 3. Добавить новую запись в конец файла\n";

cout << " 4. Прочитать значение числа, указав его порядковый номер в файле, и вернуть его значение\n";

cout << " 5. Определить количество чисел в файле\n";

cout << " 6. Создать новый файл из значений исходного, поделив каждое число на наибольший общий делитель чисел файла\n";

cout << " 7. Завершение работы программы\n";

cin >> num;

switch (num)

{

case 1:

cout << " Введите название файла: ";

cin >> file\_name;

input.open(file\_name);

if (!input)

{

cout << " Файл не открыт\n";

break;

}

else

{

create\_file(file\_name,input);

if (!input)

{

cout << " Ошибка при работе с файлом!\n";

return -1;

}

input.close();

}

break;

case 2:

cout << " Введите название файла: ";

cin >> file\_name;

output.open(file\_name);

if (!output)

{

cout << " Файл не открыт или файла не существует\n";

break;

}

else

{

output\_file(file\_name,output);

cout << endl;

if (!output)

{

cout << " Ошибка при работе с файлом!\n";

}

output.close();

}

break;

case 3:

cout << " Введите название файла: ";

cin >> file\_name;

input.open(file\_name, ios::out | ios::app);

if (!input)

{

cout << " Файл не открыт\n";

break;

}

else

{

cout << " Введите число для добавления в конец файла: ";

cin >> num;

add\_to\_file(file\_name, input,num);

if (!input)

{

cout << " Ошибка при работе с файлом!\n";

}

input.close();

}

break;

case 4:

cout << " Введите название файла: ";

cin >> file\_name;

output.open(file\_name);

if (!output)

{

cout << " Файл не открыт или файла не существует\n";

break;

}

else

{

cout << " Введите индекс, по которому нужно вернуть значение: ";

cin >> num;

int res = get\_number(file\_name, output, num);

if (res == -1)

{

cout << " IndexError\n";

}

else

{

cout << " Значение по заданному индексу: " << res << endl;

}

if (!output)

{

cout << " Ошибка при работе с файлом!\n";

}

output.close();

}

break;

case 5:

cout << " Введите название файла: ";

cin >> file\_name;

output.open(file\_name);

if (!output)

{

cout << " Файл не открыт или файла не существует\n";

break;

}

else

{

cout << " Всего чисел: " << get\_total(file\_name, output) << endl;

if (!output)

{

cout << " Ошибка при работе с файлом!\n";

}

output.close();

}

break;

case 6:

cout << " Введите название файла, на основе которого будет сгенерирован новый файл: ";

cin >> file\_name;

output.open(file\_name);

if (!output)

{

cout << " Файл не открыт или файла не существует\n";

break;

}

else

{

cout << " Введите название нового файла, куда будет записан результат: ";

cin >> new\_file;

int nod = get\_nod(file\_name);

if (nod != -1)

{

ifstream sss;

sss.open(file\_name);

create\_new\_file(output, file\_name, new\_file,nod,get\_total(file\_name,sss));

if (!sss)

{

cout << " Ошибка при работе с файлом!\n";

}

sss.close();

}

else

{

cout << " Пустой файл\n";

}

if (!output)

{

cout << " Ошибка при работе с файлом!\n";

}

output.close();

}

break;

case 7:

return 0;

default:

cout << " Нет такого пункта\n";

}

}

return 0;

}

# 1.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Результаты тестирования представлены на рисунках ниже.

1. Тестирование функции вывода чисел из файла

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание

Рисунок 1.2 – Результат тестирования функции вывода из файла

1. Заполнение нового текстового файла

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.3 – Результат тестирования функции создания и заполнения нового файла

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.4 – Результат тестирования функции создания и заполнения нового файла: созданный файл

1. Добавление новой записи в конец файла

Входные данные файл nums.txt, содержание которого показано на рисунке 1.4

Выходные данные: файл, состоящий из чисел 1 2 3.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.5 – Результат тестирования функции добавления новой записи

1. Возврат значения элемента по его индексу

Входные данные: рисунок 1.1

Выходные данные: 23

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.6 – Результат тестирования функции возврата значения числа по индексу

Входные данные: рисунок 1.1

Выходные данные: -1 (ошибка)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.7 – Результат тестирования функции возврата значения числа по индексу

1. Определить количество чисел в файле

Входные данные: рисунок 1.1

Выходные данные: 20

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 1.8 – Результат тестирования функции подсчёта количества чисел в файле

1. Создать новый файл из значений исходного, поделив каждое число на наибольший общий делитель чисел файла

Входные данные: рисунок 1.1

Выходные данные: файл без изменений (рисунок 1.1)

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание

Рисунок 1.9 – Результат тестирования функции задания варианта

Входные данные: 100 25 500 75

Выходные данные: 4 1 20 3

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Рисунок 1.10 - Результат тестирования функции задания варианта

# 2 ЗАДАНИЕ №2

# 2.1 УСЛОВИЕ ЗАДАЧИ

## 2.1.1 Формулировка условия

* Разработать программу управления двоичными файлами с записями фиксированный длины.

## 2.1.2 Требования задания

* Файл состоит из записей определённой структуры согласно варианту. Записи имеют ключ, уникальный в пределах файла.
* 1. Разработать структуру записи двоичного файла согласно варианту задания.
* 2. Подготовить тестовые данные в текстовом файле с кодировкой ASCII, в соответствии со структурой записи варианта. При открытии файла выполнить контроль его существования и открытия. Примечание. Реализация операций по чтению данных из файла будет проще, если значение для каждого поля записи размещать на отдельной строке текстового редактора.
* 3. Имя файла вводит пользователь.
* 4. При открытии файла обеспечить контроль существования и открытия файла.
* 5. При применении механизма прямого доступа к записи файла выполнить контроль присутствия записи с заданным номером в файле.
* 6. Разработать функции для выполнения операций: − преобразование тестовых данных из текстового файла в двоичный файл; − сохранение данных двоичного файла в текстовом, так, чтобы используя их можно было восстановить двоичный файл; − вывод всех записей двоичного файла; − доступ к записи по ее порядковому номеру в файле, используя механизм прямого доступа к записи в двоичном файле; − удаление записи с заданным значением ключа, выполнить путем замены на последнюю запись. − манипулирование записями в двоичном файле согласно дополнительным операциям, определенным в варианте;
* 7. Сохраните функции в новом модуле.
* 8. Разработать приложение, демонстрирующее выполнение всех операций, подключив к нему модуль с функциями.
* 9. Выполнить тестирование приложения, продемонстрировав выполнение всех операций.

## 2.1.3 Формулировка задания варианта

Структура записи: студент: номер зачетной книжки, номер группы, ФИО.

Дополнительные операции:

1. Сформировать список студентов одной группы в двоичном файле той же структуры, что и исходный.

2. Перевести некоторых студентов в другую группу. Номера зачетных книжек переводимых студентов находятся в текстовом файле, каждый номер на отдельной строке. Номер новой группы известен и передается в операцию.

# 2.2 ТЕСТОВЫЙ ПРИМЕР

Содержимое файла, который является тестовым примером представлено на рисунке №2.1.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 2.1 – Тестовый пример для задания №2

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.2 – Вывод содержимого двоичного файла в консоль

# 2.3 РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ

## 2.3.1 Структура записи двоичного файла (из кода) и ее размер в байтах (ручной расчет и системный).

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 2.3 – Структура записи

Произведем расчет размера одной структуры в байтах: 40 \* 1 + 40 \* 1 + 80 \* 1 = 160 байт.

Системный расчет размера одной структуры в байтах представлен на рисунке № 2.4.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.4 – Системный расчет размера одной структуры

## 2.3.2 Изображение структуры двоичного файла

Изображение выглядит как диаграмма, линия, График, План

Автоматически созданное описание

Рисунок 2.5 – Структура двоичного файла

## 2.3.3 Функционал приложения

Операции

1. Создание бинарного файла из текстового

Предусловие. Объект класса ifstream, передаваемый по ссылке, объект класса ofstream, передаваемый по ссылке.

Постусловие. Нет возвращаемого значения.

**void create\_bin\_file(ifstream& output, ofstream& input);**

1. Сохранение данных двоичного файла в текстовом

Предусловие. Объект класса ifstream, передаваемый по ссылке, объект класса ofstream, передаваемый по ссылке.

Постусловие. Заполняет текстовый файл, нет возвращаемого значения.

**void write\_from\_bin\_to\_text(ifstream& output, ofstream& input);**

1. Вывод всех записей двоичного файла

Предусловие. Объект класса ifstream, передаваемый по ссылке.

Постусловие. Вывод записей файла в консоль. Нет возвращаемого значения.

**void output\_bin(ifstream& output);**

1. Возвращение записи по индексу

Предусловие. Объект класса ifstream, передаваемый по ссылке, объект структуры Student, целочисленное значение n – индекс.

Постусловие. Булево значение – код работы метода.

**bool get\_student(ifstream& output, int n, Student& stud);**

1. Удаление записи по ключу

Предусловие. Объект класса fstream, передаваемый по ссылке, строковая переменная id, строковая переменная bin\_file.

Постусловие. Удаляет запись из двоичного файла. Нет возвращаемого значения.

**void delete\_student(fstream& f, string id, string bin\_file);**

1. Сформировать список студентов одной группы в двоичном файле той же структуры, что и исходный.

Предусловие. Объект класса ifstream, передаваемый по ссылке, объект класса ofstream, передаваемый по ссылке, строка – новая группа.

Постусловие. Формирует новый двоичный файл. Нет возвращаемого значения.

**void one\_group(ifstream& output, ofstream& input, string new\_group);**

1. Перевести некоторых студентов в другую группу.

Предусловие. Объект класса ifstream, передаваемый по ссылке, объект класса fstream, передаваемый по ссылке, строковая переменная new\_group.

Постусловие. Изменяет содержимое двоичного файла. Нет возвращаемого значения.

**void move\_students(fstream& f, ifstream& output, string new\_group);**

## 2.3.4 Код основной программы

Код файла Student.h:

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <io.h>

#include <windows.h>

#include <fcntl.h>

#include <share.h>

#include <sys/stat.h>

#include <string>

#include <iostream>

#include <fstream>

#include <vector>

#include <algorithm>

#include <string.h>

using namespace std;

struct Student

{

char studentID[40];

char group[40];

char full\_name[80];

Student();

~Student();

};

void create\_bin\_file(ifstream& output, ofstream& input);

void write\_from\_bin\_to\_text(ifstream& output, ofstream& input);

void output\_bin(ifstream& output);

bool get\_student(ifstream& output, int n, Student& stud);

void delete\_student(fstream& f, string id, string bin\_file);

void one\_group(ifstream& output, ofstream& input, string new\_group);

void move\_students(fstream& f, ifstream& output, string new\_group);

Код файла Student.cpp:

#include "Student.h"

Student::Student()

{

this->studentID[0] = '\0';

this->group[0] = '\0';

this->full\_name[0] = '\0';

}

Student::~Student()

{

this->studentID[0] = '\0';

this->group[0] = '\0';

this->full\_name[0] = '\0';

}

void create\_bin\_file(ifstream& output, ofstream& input)

{

while (!output.eof())

{

Student Vlad;

char symbol;

int i = 0;

output.get(Vlad.studentID[i]);

while (Vlad.studentID[i] != '\n')

{

i++;

output.get(Vlad.studentID[i]);

}

Vlad.studentID[i] = '\0';

i = 0;

output.get(Vlad.group[i]);

while (Vlad.group[i] != '\n')

{

i++;

output.get(Vlad.group[i]);

}

Vlad.group[i] = '\0';

i = 0;

output.get(Vlad.full\_name[i]);

while (Vlad.full\_name[i] != '\n' and !output.eof())

{

i++;

output.get(Vlad.full\_name[i]);

}

Vlad.full\_name[i] = '\0';

input.write((char\*)&Vlad, sizeof(Vlad));

}

}

void write\_from\_bin\_to\_text(ifstream& output, ofstream& input)

{

Student Vlad;

output.read((char\*)&Vlad, sizeof(Vlad));

while (!output.eof())

{

input << Vlad.studentID << '\n'

<< Vlad.group << '\n'

<< Vlad.full\_name;

output.read((char\*)&Vlad, sizeof(Vlad));

if (!output.eof())

{

input << '\n';

}

}

}

void output\_bin(ifstream& output)

{

Student Vlad;

output.read((char\*)&Vlad, sizeof(Student));

while (!output.eof())

{

cout << Vlad.studentID << " | " << Vlad.group << " | " << Vlad.full\_name << "\n";

output.read((char\*)&Vlad, sizeof(Vlad));

}

}

bool get\_student(ifstream& output,int n,Student& stud)

{

output.seekg(n \* sizeof(Student), ios::beg);

if (!output.bad())

{

output.read((char\*)&stud, sizeof(Student));

return 0;

}

return 1;

}

void delete\_student(fstream& f, string id, string bin\_file)

{

f.seekg((-1) \* sizeof(Student), std::ios::end);

Student Vlad;

f.read(reinterpret\_cast<char\*>(&Vlad), sizeof(Student));

bool ok = false;

f.seekg(0, std::ios::beg);

while (true)

{

Student V;

f.read(reinterpret\_cast<char\*>(&V), sizeof(Student));

if (f.eof())

{

break;

}

if (V.studentID == id)

{

f.seekg((-1)\*sizeof(Student), std::ios::cur);

f.write(reinterpret\_cast<char\*>(&Vlad), sizeof(Student));

ok = true;

break;

}

}

f.seekg(0, std::ios::end);

int size = f.tellg();

f.close();

if (!ok)

{

return;

}

int x;

errno\_t r = \_sopen\_s(&x, bin\_file.c\_str(), \_O\_RDWR, \_SH\_DENYNO, \_S\_IREAD | \_S\_IWRITE);

if (!r)

{

\_chsize(x, size - sizeof(Student));

\_close(x);

}

}

void one\_group(ifstream& output, ofstream& input, string new\_group)

{

Student Vlad;

output.read((char\*)&Vlad, sizeof(Student));

while (!output.eof())

{

if (Vlad.group == new\_group)

{

input << Vlad.studentID << '\n'

<< Vlad.group << '\n'

<< Vlad.full\_name;

output.read((char\*)&Vlad, sizeof(Student));

if (!output.eof())

{

input << '\n';

}

}

else

output.read((char\*)&Vlad, sizeof(Student));

}

}

void move\_students(fstream& f, ifstream& output, string new\_group)

{

vector <string> data;

string z;

while (!output.eof())

{

output >> z;

data.push\_back(z);

}

while (!f.eof())

{

Student Vlad;

f.read((char\*)&Vlad, sizeof(Student));

if (f.eof())

{

break;

}

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

{

if (data[i] == Vlad.studentID)

{

strcpy(Vlad.group, new\_group.c\_str());

f.seekg((-1) \* sizeof(Student), ios::cur);

f.write((char\*)&Vlad, sizeof(Student));

}

}

}

}

Код файла main.cpp:

#include "Student.h"

int main()

{

SetConsoleOutputCP(CP\_UTF8);

setlocale(LC\_CTYPE, "Russian");

string text\_file;

string bin\_file;

string new\_file;

string new\_group;

string id;

ifstream output;

ofstream input;

ofstream newfile;

fstream f;

int num;

int n;

int key\_work;

Student stud;

system("chcp 1251");

while (true)

{

cout << "-----------------------------------------------------------------------------------------------------------" << endl;

cout << " Меню\n";

cout << " 1. Преобразовать текстовый файл в двоичный\n";

cout << " 2. Вывести содержимое двоичного файла в текстовый\n";

cout << " 3. Вывести содержимое двоичного файла в консоль\n";

cout << " 4. Получение записи из двоичного файла по порядковому номеру\n";

cout << " 5. Удаление записи по ключу\n";

cout << " 6. Сформировать список студентов одной группы\n";

cout << " 7. Перевести некоторых студентов в другую группу\n";

cout << " 8. Завершение работы программы\n";

cin >> num;

switch (num)

{

case 1:

cout << " Введите название текстового файла: ";

cin >> text\_file;

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

output.open(text\_file);

if (!output.is\_open())

{

cout << " Текстовый файл не открыт\n";

}

else

{

input.open(bin\_file, ios::binary);

create\_bin\_file(output, input);

if (!input.good())

{

cout << " Ошибка записи в двоичный файл\n";

return -1;

}

else

{

output.close();

input.close();

}

}

break;

case 2:

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

cout << " Введите название текстового файла: ";

cin >> text\_file;

output.open(bin\_file, ios::binary);

if (!output.is\_open())

{

cout << " Двоичный файл не открыт\n";

}

else

{

input.open(text\_file);

write\_from\_bin\_to\_text(output, input);

if (!input)

{

cout << " Ошибка записи в текстовый файл\n";

return -1;

}

else

{

output.close();

input.close();

}

}

break;

case 3:

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

output.open(bin\_file, ios::binary);

if (!output.is\_open())

{

cout << " Двоичный файл не открыт\n";

}

else

{

output\_bin(output);

if (output.bad())

{

cout << " Ошибка при чтении из двоичного файла\n";

}

else

{

output.close();

}

}

break;

case 4:

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

output.open(bin\_file, ios::binary);

if (output.is\_open())

{

cout << " Введите номер записи: ";

cin >> n;

key\_work = get\_student(output, n, stud);

if (key\_work == 1 or stud.studentID[0] == '\0')

{

cout << " Записи с заданным номером не существует\n";

}

else

{

cout << " " << stud.studentID << " | " << stud.group << " | " << stud.full\_name << endl;

stud.~Student();

}

output.close();

}

else

{

cout << " Двоичный файл не существует\n";

}

break;

case 5:

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

f.open(bin\_file);

if (f.is\_open())

{

cout << " Введите номер зачетной книжки: ";

cin >> id;

delete\_student(f,id,bin\_file);

f.close();

}

else

{

cout << " Двоичный файл не существует\n";

}

break;

case 6:

cout << " Введите название двоичного файла, где записана информация о студентах: ";

cin >> bin\_file;

output.open(bin\_file, ios::binary);

if (output.is\_open())

{

cout << " Введите название нового файла, где будет сформирован список студентов одной группы: ";

cin >> new\_file;

new\_file += ".txt";

input.open(new\_file);

if (!input.is\_open())

{

cout << " Двоичный файл для записи не открыт\n";

}

else

{

cout << " Введите номер группы для создания нового файла со студентами этой группы: ";

cin >> new\_group;

one\_group(output, input, new\_group);

if (input.bad())

{

cout << " Ошибка при записи из двоичного файла в двоичный\n";

}

else

{

input.close();

}

}

output.close();

output.open(new\_file);

string t = new\_file.substr(0, new\_file.size() - 4);

input.open(t, ios::binary);

create\_bin\_file(output, input);

output.close();

input.close();

}

else

{

cout << " Двоичный файл не существует\n";

}

break;

case 7:

cout << " Введите название двоичного файла: ";

cin >> bin\_file;

f.open(bin\_file);

if (f.is\_open())

{

cout << " Введите название текстового файла, где храянтся номера зачеток: ";

cin >> text\_file;

output.open(text\_file);

if (output.is\_open())

{

cout << " Введите номер новой группы: ";

cin >> new\_group;

move\_students(f, output, new\_group);

output.close();

}

else

{

cout << " Текстовый файл не существует\n";

}

f.close();

}

else

{

cout << " Двоичный файл не существует\n";

}

break;

case 8:

return 0;

default:

cout << " Нет такого пункта\n";

}

}

return 0;

}

# 2.4 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

1) Преобразование текстового файла в двоичный и вывод содержимого двоичного файла в текстовый:

Входные данные: рисунок 2.1

Выходные данные: рисунок 2.6 и рисунок 2.7

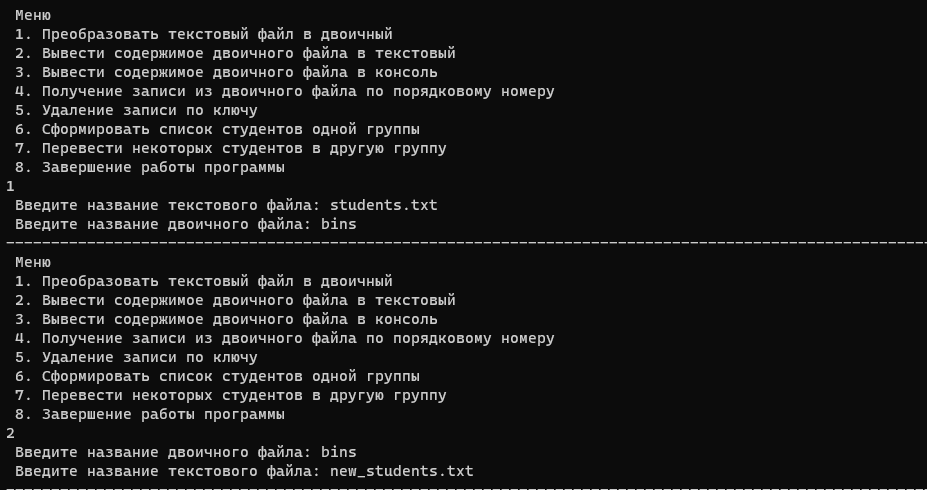


Рисунок 2.6 – Результат тестирования функций преобразования текстового файла в двоичный и вывод содержимого двоичного файла в текстовый

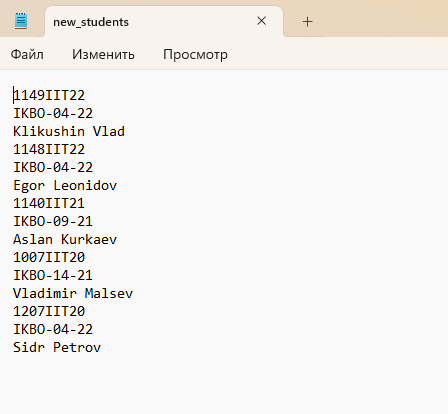


Рисунок 2.7 – Содержание файла new\_students.txt

2) Вывод содержимого двоичного файла в консоль

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.8 – Результат тестирования функции вывода содержимого двоичного файла в консоль

3) Получение записи из двоичного файла по порядковому номеру

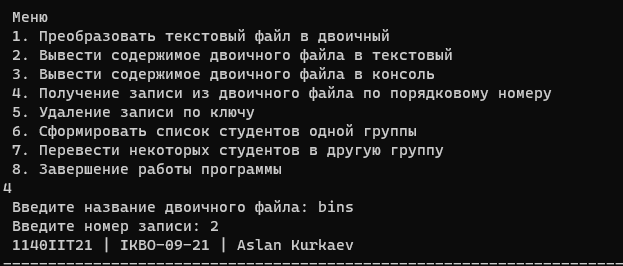


Рисунок 2.9 – Результат теста функции получения записи из двоичного файла по индексу

4) Удаление записи по ключу

Входные данные: Рисунок 2.1

Выходные данные: Рисунок 2.10

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.10 – Результат теста функции удаления записи по ключу

5) Сформировать новый список студентов одной группы

Входные данные: рисунок 2.1

Выходные данные: рисунок 2.11

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.11 – Результат теста формирования нового двоичного файла из студентов одной группы

6) Перевести некоторых студентов в новую группу

Входные данные: исходный двоичный файл – рисунок 2.1, текстовый файл – рисунок 2.12

Выходные данные: рисунок 2.13

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.12 – Содержание файла move.txt

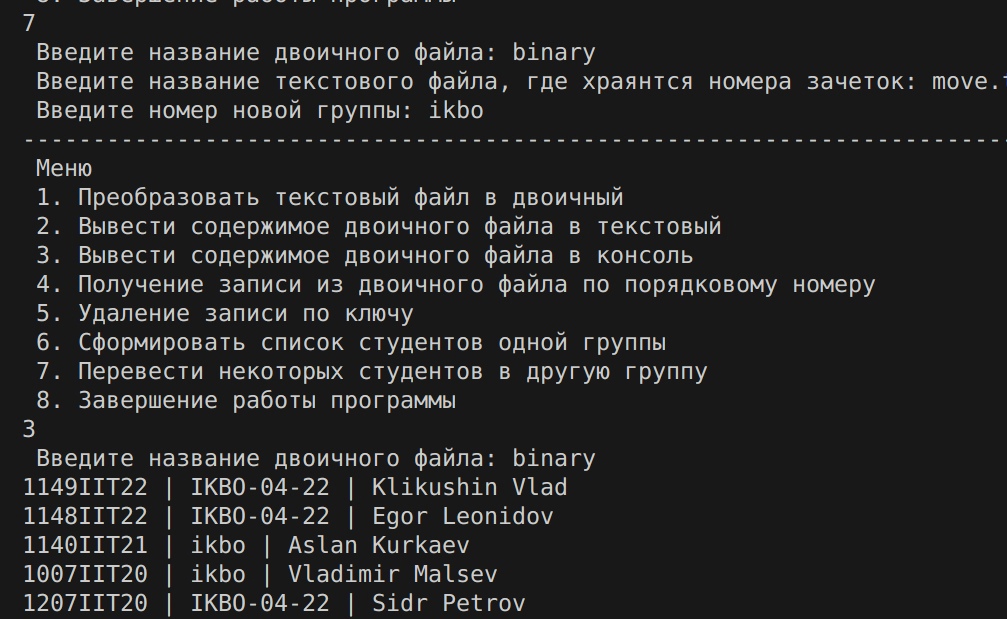


Рисунок 2.13 – Результат теста функции перевода некоторых студентов в другую группу

# 3 ВЫВОДЫ

В ходе выполнения практической работы были получены навыки работы с текстовыми и двоичными файлами.

# 4 ИСПОЛЬЗУЕМЫЕ ИСТОЧНИКИ

1. Учебно-методическое пособие СиАОД (часть 2)
2. Приложение к практическим работам – СДО (online-edu.mirea.ru)