**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №1**

**по дисциплине «Программирование»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 2373 |  | Звягина М.С. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Изучение структур в c++, массивов структур, создание прототипа баз данный в c++ при помощи структур, добавление функционала.

**Основные теоретические положения.**

В языке c++, структура — композитный тип данных, инкапсулирующий без сокрытия набор значений различных типов. Порядок размещения значений в памяти задаётся при определении типа и сохраняется на протяжении времени жизни объектов, что даёт возможность косвенного доступа. Пример объявления структуры: struct str\_name

Структуры можно использовать в качестве параметров функций, как и обычные переменные. Для структур поддерживаются все три механизма передачи данных: по значению, через указатели и по ссылке.

Объединения также хранят набор элементов, но в отличие от структуры все элементы объединения имеют нулевое смещение. А это значит, что разные элементы занимают в памяти один и тот же участок.

Для определения объединений применяется ключевое слово union и следующий формальный синтаксис:

union имя\_объединения

{

    определения\_элементов

};

**Постановка задачи**

Необходимо создать программу, позволяющую:

1.   Создание новой записи о студенте.

2.   Внесение изменений в уже имеющуюся запись.

3.   Вывод всех данных о студентах.

4.   Вывод информации обо всех студентах группы *N*. *N* – инициализируется пользователем.

5.   Вывод топа самых успешных студентов с наивысшим по рейтингу средним баллом за прошедшую сессию.

6.   Вывод количества студентов мужского и женского пола.

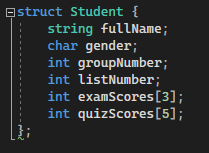
7.   Вывод данных о студентах, которые не получают стипендию; учатся только на «хорошо» и «отлично»; учатся только на «отлично»;

8.   Вывод данных о студентах, имеющих номер в списке – *k*.

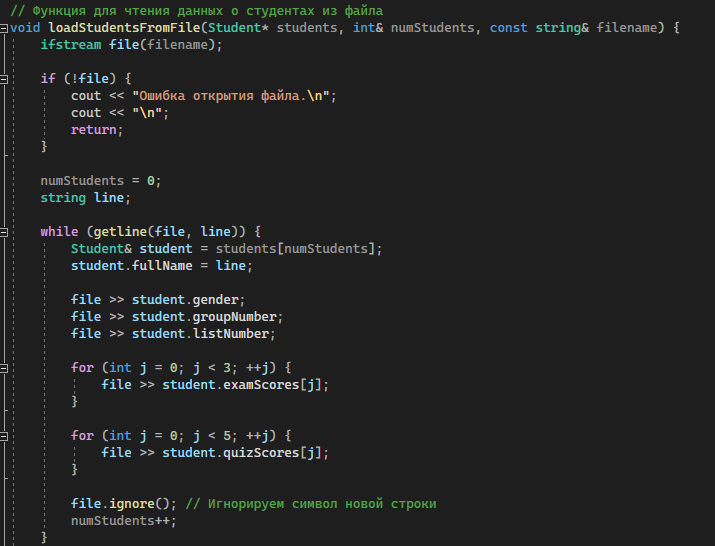
**Выполнение работы.**

Код программы представлен в приложении А.

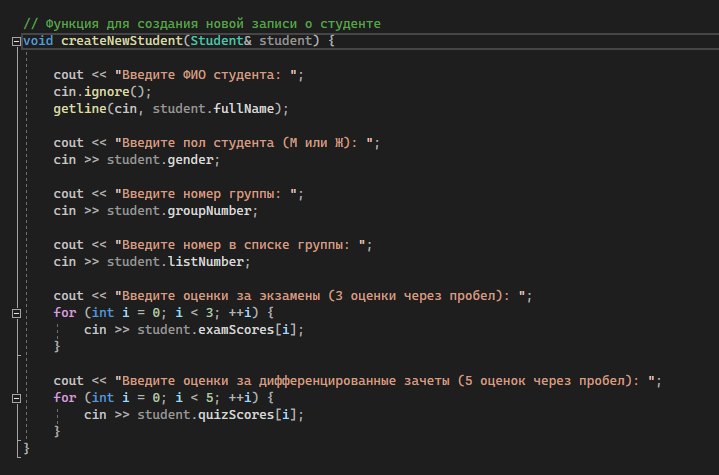
**Блок описания кода и использованных алгоритмов:**

Создаём динамический массив структур, размер которого узнаём считывая количество строк в изначальном файле List.txt.

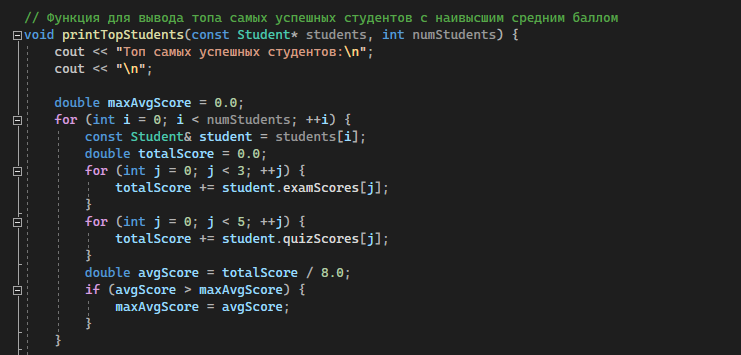
Далее просто считываем файл и полученные данные добавляем в массив структур.



1. Сначала массив расширяется на единицу, а затем в появившуюся ячейку записываются данные о новом студенте.

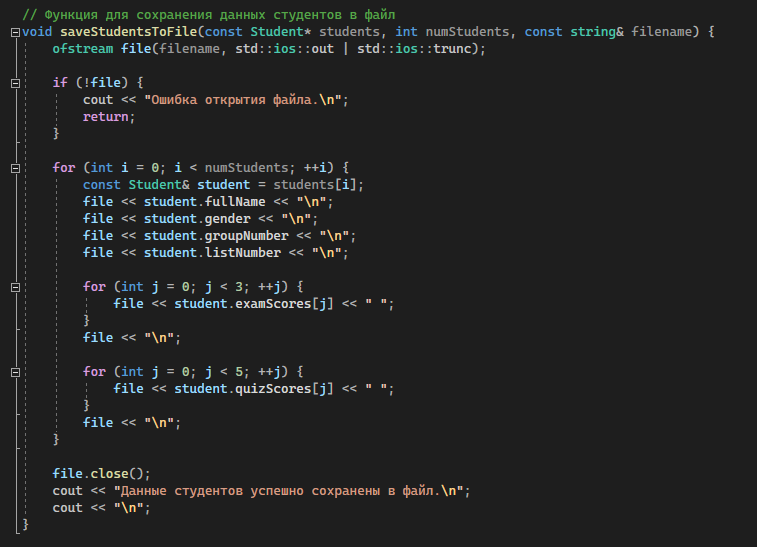


1. Пользователь вводит номер студента, данные которого хочет изменить, а затем меняет их.
2. Циклом for по очереди выводим всех студентов в списке.
3. Пользователь вводит номер группы и перебором выводятся студенты с соответствующим номером группы.
4. Сначала создаём функцию для высчитывания среднего значения оценок обучающегося.



1. Перебором проходимся по всему массиву и увеличиваем счётчики обеих полов.
2. Делаем проверку оценок студентов. Если находим соответствие, то выводим в консоль.
3. Пользователь вводит номер и перебором выводятся студенты с соответствующим номеров в группе.

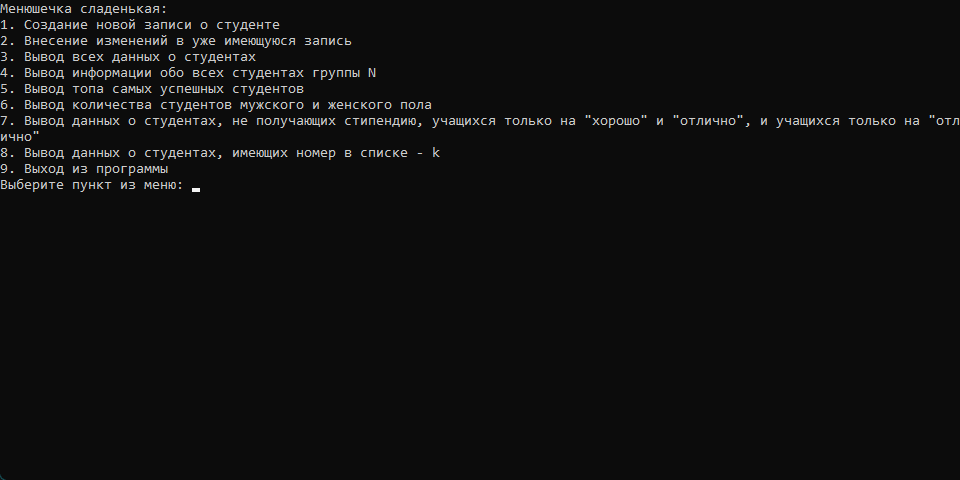
По завершению работы программы записываем получившиеся данные обратно в файл.

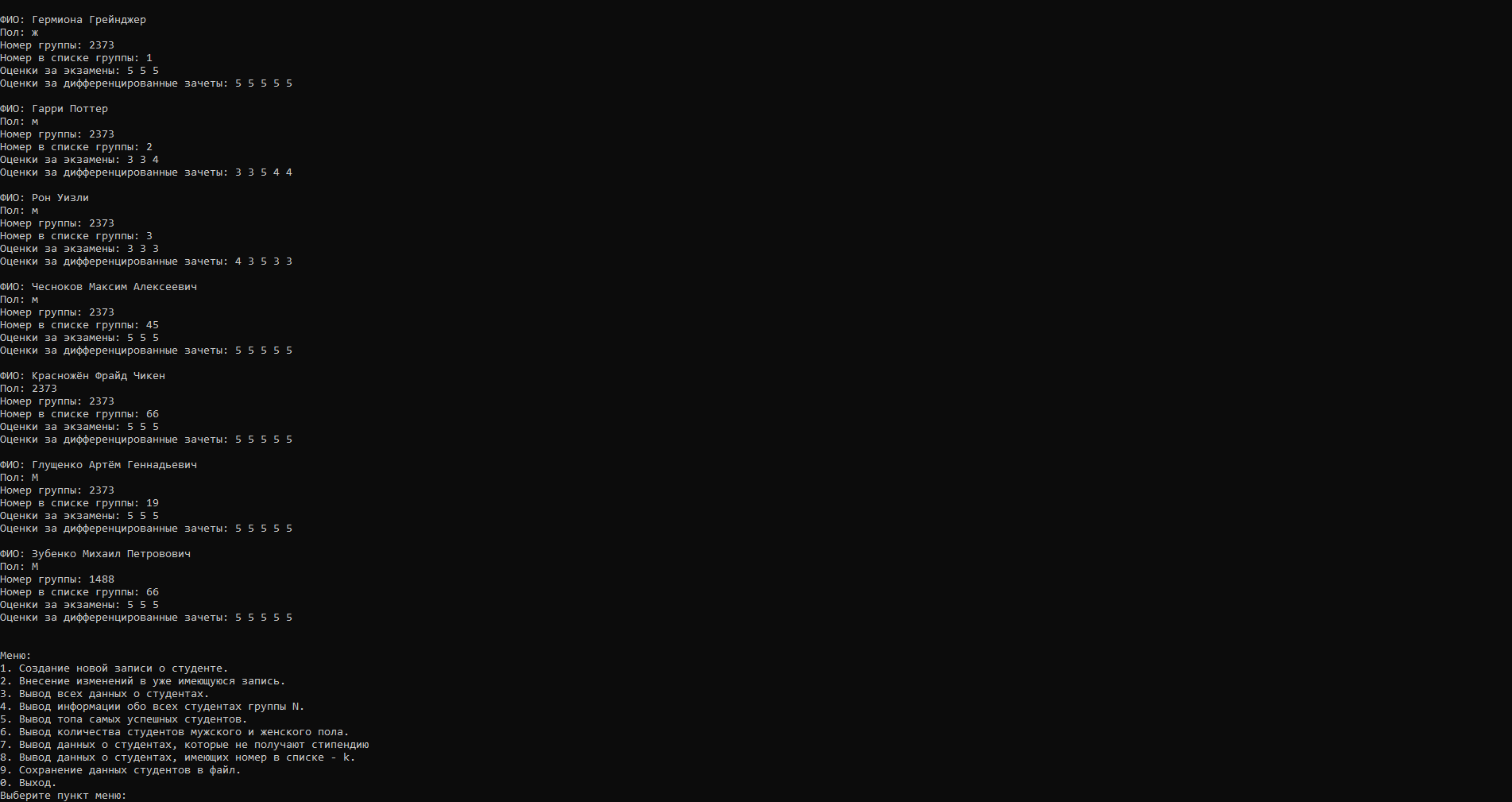


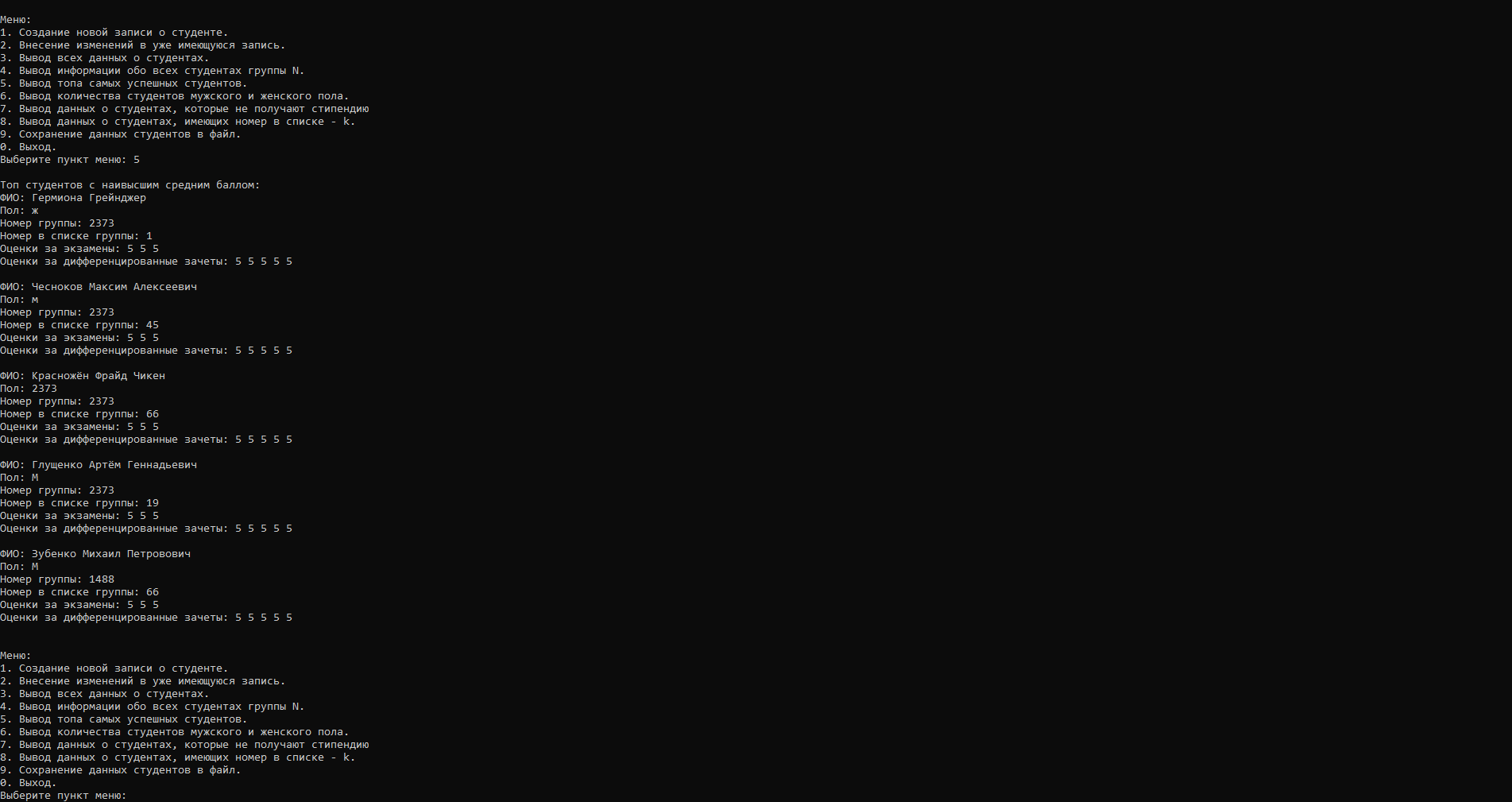
**Выводы.**

Мы научились использовать структуры c++, написали прототип баз данный c++, научились использовать функционал структур.

**Блок скриншотов работы программы**







Приложение А

рабочий код

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <Windows.h>

using namespace std;

struct Student { // Структура, представляющая информацию о студенте

string fullName;

string gender;

int groupNumber;

int listNumber;

vector<int> examGrades;

vector<int> quizGrades;

};

void createStudent(vector<Student>& students) { // Функция для создания новой записи о студенте

Student newStudent;

SetConsoleCP(1251);

cout << "Введите ФИО студента: ";

getline(cin >> ws, newStudent.fullName);

cout << "Введите пол студента: ";

getline(cin >> ws, newStudent.gender);

cout << "Введите номер группы студента: ";

cin >> newStudent.groupNumber;

cout << "Введите номер в списке группы студента: ";

cin >> newStudent.listNumber;

cout << "Введите оценки за экзамены (через пробел): ";

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

int grade;

cin >> grade;

newStudent.examGrades.push\_back(grade);

}

cout << "Введите оценки за дифференцированные зачеты (через пробел): ";

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

int grade;

cin >> grade;

newStudent.quizGrades.push\_back(grade);

}

students.push\_back(newStudent);

cout << "Запись о студенте успешно создана." << endl;

}

int countStudents() // Функция посчёта количества студентов

{

ifstream database("base");

if (database.is\_open())

{

int temp = 0;

string data;

while (!database.eof())

{

getline(database, data);

temp++;

}

database.close();

int n;

n = temp / 7;

return n;

}

else return 0;

}

void readFile(vector<Student>& students) { // Функция для сохранения данных студентов в файл

char trash;

ifstream file;

file.open("base");

if (file.is\_open()) {

for (int j = 0; j < countStudents(); j++) {

Student student;

getline(file, student.fullName);

getline(file, student.gender);

file >> student.groupNumber >> student.listNumber;

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

int grade;

file >> grade;

student.examGrades.push\_back(grade);

}

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

int grade;

file >> grade;

student.quizGrades.push\_back(grade);

}

for (int i = 0; i < 3; i++) file.get(trash);

students.push\_back(student);

}

}

else {

cout << "Не удалось открыть файл base для считывания." << endl;

}

file.close();

}

void modifyStudent(vector<Student>& students) { // Функция для внесения изменений в уже имеющуюся запись

int studentIndex;

cout << "Введите индекс студента для внесения изменений: ";

cin >> studentIndex;

if (studentIndex >= 0 && studentIndex < students.size()) {

Student& student = students[studentIndex];

SetConsoleCP(1251);

cout << "Введите новые данные студента:" << endl;

cout << "ФИО студента: ";

getline(cin >> ws, student.fullName);

cout << "Пол студента: ";

getline(cin >> ws, student.gender);

cout << "Номер группы студента: ";

cin >> student.groupNumber;

cout << "Номер в списке группы студента: ";

cin >> student.listNumber;

cout << "Оценки за экзамены (через пробел): ";

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

int grade;

cin >> grade;

student.examGrades[i] = grade;

}

cout << "Оценки за дифференцированные зачеты (через пробел): ";

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

int grade;

cin >> grade;

student.quizGrades[i] = grade;

}

cout << "Данные о студенте успешно изменены." << endl;

}

else {

cout << "Студент с указанным индексом не существует." << endl;

}

}

void displayAllStudents(const vector<Student>& students) { // Функция для вывода всех данных о студентах

for (const Student& student : students) {

cout << "ФИО: " << student.fullName << endl;

cout << "Пол: " << student.gender << endl;

cout << "Номер группы: " << student.groupNumber << endl;

cout << "Номер в списке группы: " << student.listNumber << endl;

cout << "Оценки за экзамены: ";

for (int grade : student.examGrades) {

cout << grade << " ";

}

cout << endl;

cout << "Оценки за дифференцированные зачеты: ";

for (int grade : student.quizGrades) {

cout << grade << " ";

}

cout << endl << endl;

}

}

void displayStudentsInGroup(const vector<Student>& students) { // Функция для вывода информации обо всех студентах группы N

int groupNumber;

cout << "Введите номер группы: ";

cin >> groupNumber;

cout << "Студенты группы " << groupNumber << ":" << endl;

for (const Student& student : students) {

if (student.groupNumber == groupNumber) {

cout << "ФИО: " << student.fullName << endl;

cout << "Пол: " << student.gender << endl;

cout << "Номер в списке группы: " << student.listNumber << endl;

cout << "Оценки за экзамены: ";

for (int grade : student.examGrades) {

cout << grade << " ";

}

cout << endl;

cout << "Оценки за дифференцированные зачеты: ";

for (int grade : student.quizGrades) {

cout << grade << " ";

}

cout << endl << endl;

}

}

}

void displayTopStudents(const vector<Student>& students) { // Функция для вывода топа самых успешных студентов (не я)

vector<Student> topStudents;

double highestAverage = 0.0; // Находим средний балл каждого студента и выбираем наивысшие значения

for (const Student& student : students) {

double sum = 0.0;

for (int grade : student.examGrades) {

sum += grade;

}

for (int grade : student.quizGrades) {

sum += grade;

}

double average = sum / (student.examGrades.size() + student.quizGrades.size());

if (average > highestAverage) {

highestAverage = average;

}

}

for (const Student& student : students) { // Добавляем студентов с наивысшим средним баллом в список топ стьюдентс

double sum = 0.0;

for (int grade : student.examGrades) {

sum += grade;

}

for (int grade : student.quizGrades) {

sum += grade;

}

double average = sum / (student.examGrades.size() + student.quizGrades.size());

if (average == highestAverage) {

topStudents.push\_back(student);

}

}

cout << "Топ студентов с наивысшим средним баллом:" << endl;

for (const Student& student : topStudents) {

cout << "ФИО: " << student.fullName << endl;

cout << "Пол: " << student.gender << endl;

cout << "Номер группы: " << student.groupNumber << endl;

cout << "Номер в списке группы: " << student.listNumber << endl;

cout << "Оценки за экзамены: ";

for (int grade : student.examGrades) {

cout << grade << " ";

}

cout << endl;

cout << "Оценки за дифференцированные зачеты: ";

for (int grade : student.quizGrades) {

cout << grade << " ";

}

cout << endl << endl;

}

}

void countStudentsByGender(const vector<Student>& students) { //Функция для подсчета количества студентов женского и мадамского пола

int maleCount = 0;

int femaleCount = 0;

SetConsoleCP(1251);

for (const Student& student : students) {

if (student.gender == "м") {

maleCount++;

}

else if (student.gender == "ж") {

femaleCount++;

}

}

cout << "Количество студентов мужского пола: " << maleCount << endl;

cout << "Количество студентов женского пола: " << femaleCount << endl;

}

void displayStudentsWithoutScholarship(const vector<Student>& students) { //Функция для вывода данных о студентах, которые не получают стипендию

cout << "Студенты, не получающие стипендию:" << endl;

for (const Student& student : students) {

bool hasLowGrades = false;

for (int grade : student.examGrades) {

if (grade < 4) {

hasLowGrades = true;

break;

}

}

for (int grade : student.quizGrades) {

if (grade < 4) {

hasLowGrades = true;

break;

}

}

if (hasLowGrades) {

cout << "ФИО: " << student.fullName << endl;

cout << "Пол: " << student.gender << endl;

cout << "Номер группы: " << student.groupNumber << endl;

cout << "Номер в списке группы: " << student.listNumber << endl;

cout << "Оценки за экзамены: ";

for (int grade : student.examGrades) {

cout << grade << " ";

}

cout << endl;

cout << "Оценки за дифференцированные зачеты: ";

for (int grade : student.quizGrades) {

cout << grade << " ";

}

cout << endl << endl;

}

}

}

void displayStudentsByListNumber(const vector<Student>& students) { //Функция для вывода данных о студентах с номером в списке k

int listNumber;

cout << "Введите номер в списке: ";

cin >> listNumber;

cout << "Студенты с номером в списке " << listNumber << ":" << "\n" << endl;

for (const Student& student : students) {

if (student.listNumber == listNumber) {

cout << "ФИО: " << student.fullName << endl;

cout << "Пол: " << student.gender << endl;

cout << "Номер группы: " << student.groupNumber << endl;

cout << "Оценки за экзамены: ";

for (int grade : student.examGrades) {

cout << grade << " ";

}

cout << endl;

cout << "Оценки за дифференцированные зачеты: ";

for (int grade : student.quizGrades) {

cout << grade << " ";

}

cout << endl << endl;

}

}

}

void saveDataToFile(const vector<Student>& students, const string& filename) { // Функция для сохранения данных студентов в файл

ofstream file(filename); // Открытие файла для записи с параметром ios::app

if (file.is\_open()) {

for (const Student& student : students) {

file << student.fullName << endl;

file << student.gender << endl;

file << student.groupNumber << endl;

file << student.listNumber << endl;

for (int grade : student.examGrades) {

file << grade << " ";

}

file << endl;

for (int grade : student.quizGrades) {

file << grade << " ";

}

file << endl << endl;

}

cout << "Данные успешно сохранены в файл " << filename << endl;

}

else {

cout << "Не удалось открыть файл " << filename << " для записи." << endl;

}

file.close();

}

int main() {

setlocale(0, "");

vector<Student> students;

readFile(students);

int choice;

do {

cout << "Меню:" << endl;

cout << "1. Создание новой записи о студенте." << endl;

cout << "2. Внесение изменений в уже имеющуюся запись." << endl;

cout << "3. Вывод всех данных о студентах." << endl;

cout << "4. Вывод информации обо всех студентах группы N." << endl;

cout << "5. Вывод топа самых успешных студентов." << endl;

cout << "6. Вывод количества студентов мужского и женского пола." << endl;

cout << "7. Вывод данных о студентах, которые не получают стипендию" << endl;

cout << "8. Вывод данных о студентах, имеющих номер в списке – k." << endl;

cout << "9. Сохранение данных студентов в файл." << endl;

cout << "0. Выход." << endl;

cout << "Выберите пункт меню: ";

cin >> choice;

cout << endl;

switch (choice) {

case 1:

createStudent(students);

break;

case 2:

modifyStudent(students);

break;

case 3:

displayAllStudents(students);

break;

case 4:

displayStudentsInGroup(students);

break;

case 5:

displayTopStudents(students);

break;

case 6:

countStudentsByGender(students);

break;

case 7:

displayStudentsWithoutScholarship(students);

break;

case 8:

displayStudentsByListNumber(students);

break;

case 9: {

string filename;

cout << "Введите имя файла для сохранения данных: ";

cin >> filename;

saveDataToFile(students, filename);

break;

}

case 0:

cout << "Программа завершена." << endl;

break;

default:

cout << "Некорректный ввод. Попробуйте снова." << endl;

break;

}

cout << endl;

} while (choice != 0);

return 0;

}}