**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Информационных систем**

отчет

**по лабораторной работе №1**

**по дисциплине «Программирование»**

Тема: Типы данных, определяемые пользователем

|  |  |  |
| --- | --- | --- |
| Студент гр. 0323 |  | Кольцов К.Э |
| Преподаватель |  | Глущенко А.Г. |

Санкт-Петербург

2021

**Цель работы.**

изучение и организация структур; получение практических навыков работы со структурами; определение преимуществ и недостатков использования структур.

Необходимо создать массив структур, содержащий информацию о студентах: ФИО, пол, номер группы, номер в списке группы, оценки за прошедшую сессию (всего 3 экзамена и 5 дифференцированных зачетов), форма обучения\*, отметка времени о внесении или изменении данных\*. Ввод и изменение данных обо всех студентах должен осуществляться в файл students\*.

Написать функции, реализующие операции со структурами (ввод данных с клавиатуры):

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.   Определение количества студентов, которые будут получать стипендию (стипендия начисляется, если у студента нет троек и очная форма обучения).\*

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

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

10.   Вывод всех записей, сделанных в день, который введет пользователь. Вывод всех записей, сделанных после полудня. Вывод всех записей, сделанных до полудня.\*

**Основные теоретические положения.**

Структуры представляют собой группы связанных между собой, как правило, разнотипных переменных, объединенных в единый объект, в отличие от массива, все элементы которого однотипны. В языке C++ структура является видом класса и обладает всеми его свойствами.

**Обработка результатов эксперимента.**

Программа выводит результат корректной обработки строки и правильно находит подстроку в предложении.

**Выводы.**

В ходе данной лабораторной работы я изучил создание собственных типов данных в c++.

ПРИЛОЖЕНИЕ А

Листинг программы

#include <iostream>

#include <string>

#include <ctime>

#include <vector>

#include <fstream>

struct Student

{

std::string name[3]; //ФИО студента, 0 - фамилия, 1 - имя, 2 - отчество

char gender; // при создании и изменении доступны лишь F - женский и M - мужской

int group; // номер группы

int count; // номер в группе

int exam[3]; // оценки за экзамены

int offset[5]; // оценки за диф зачеты

char study; // форма обучения, при создании и изменении доступны O - очная, E - очно-заочная, Z - заочная

std::string time; // строка, генерируемая методом ctime при любом изменении экземпляра структуры

};

void save(std::vector<Student> \*); // сохраняет список в файл

void addNew(std::vector<Student> \*); // добавляет новый элемент

void fromFile(std::vector<Student> \*); // загружает список студентов из файла

void showStudent(Student \*); // выводит данные о студенте, нужна для других функциях

void showAll(std::vector<Student> \*); // вывод всех студентов

void showGroup(std::vector<Student> \*); // выводит студентов определенной группы

void showGreat(std::vector<Student> \*); // выводит данные лучших студентов

void showSuperGreat(std::vector<Student> \*); // выводит данные об отличниках

void showGenderCount(std::vector<Student> \*); // выводит количество студентов мужского и женского пола

void showGrant(std::vector<Student> \*); // выводит данные о студентах, получающих стипендию

void showAntiGrant(std::vector<Student> \*); // выводит данные о студентах без стипендии

void showNedoGrant(std::vector<Student> \*); // выводит данные о хорошистах

void showCount(std::vector<Student> \*); // выводит данные о студентах по их номеру в группе

void change(std::vector<Student> \*); // спрашивает номер студента, и меняет его запись

void change(Student \*); // меняет данные конкретного студента

int main() {

setlocale(LC\_ALL, "Russian"); //для корректного вывода руских букв;

std::vector<Student> students;

fromFile(&students);

char answer;

bool flag = true;

while (flag) {

std::cout << "Выберите команду: \n" \

"n: создание новой записи \n" \

"c: изменение записи \n" \

"v: вывести данные о всех студентах \n" \

"w: вывести данные о студантах группы \n" \

"h: вывести список самых успешных студентов \n" \

"g: вывести количество студентов мужского и женского пола \n" \

"s: вывести данные о студентов, получающих стипендию \n" \

"i: вывести данные о студентах, не получающих стипендию \n" \

"b: вывести данные о хорошистах \n" \

"a: вывести данные об отличниках \n" \

"d: вывести данные о студентах по номеру в группе \n" \

"e: завершить работу программы \n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

switch (answer) {

case 'n':

addNew(&students);

break;

case 'c':

change(&students);

break;

case 'v':

showAll(&students);

break;

case 'w':

showGroup(&students);

break;

case 'h':

showGreat(&students);

break;

case 'g':

showGenderCount(&students);

break;

case 's':

showGrant(&students);

break;

case 'i':

showAntiGrant(&students);

break;

case 'b':

showNedoGrant(&students);

break;

case 'a':

showSuperGreat(&students);

break;

case 'd':

showCount(&students);

break;

case 'e':

flag = false;

break;

default:

std::cout << "Вы ввели неправильную команду \n";

break;

}

}

save(&students);

system("pause");

return 0;

}

void save(std::vector<Student> \*students) {

std::ofstream out;

out.open("Students.txt");

if (out.is\_open()) {

out << (\*students).size() << '\n';

for (int i = 0; i < (\*students).size(); i++) {

out << (\*students)[i].name[0] << ' ' << (\*students)[i].name[1] << ' ' << (\*students)[i].name[2] << '\n';

out << (\*students)[i].gender << '\n';

out << (\*students)[i].group << ' ' << (\*students)[i].count << '\n';

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

out << (\*students)[i].exam[j] << ' ';

out << '\n';

for (int j = 0; j < 5; j++)

out << (\*students)[i].offset[j] << ' ';

out << '\n';

out << (\*students)[i].study << '\n';

out << (\*students)[i].time << '\n';

}

}

else

std::cout << "Файл не доступен\n";

out.close();

}

void addNew(std::vector<Student> \*students) {

Student stud;

bool correct = false;

std::cout << "Введите данные о студенте\n" \

"Фамилия:\n";

while (!correct) {

correct = true;

std::cin >> stud.name[0];

char \* point = &stud.name[0][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

stud.name[0][0] = stud.name[0][0] + 'A' - 'a';

correct = false;

std::cout << "Введите имя:\n";

while (!correct) {

correct = true;

std::cin >> stud.name[1];

char \* point = &stud.name[1][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

stud.name[1][0] = stud.name[1][0] + 'A' - 'a';

correct = false;

std::cout << "Введите отчество:\n";

while (!correct) {

correct = true;

std::cin >> stud.name[2];

char \* point = &stud.name[2][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

stud.name[2][0] = stud.name[2][0] + 'A' - 'a';

correct = false;

std::cout << "Введите пол (F-женский, M-мужской):\n";

while (!correct) {

correct = true;

std::cin >> stud.gender;

if ((stud.gender == 'f') || (stud.gender == 'm'))

stud.gender = stud.gender + 'A' - 'a';

if ((stud.gender != 'F') && (stud.gender != 'M')) {

std::cout << "Некоректный ввод, достопны только F и M\n";

correct = false;

}

}

std::cout << "Введите четырехзначный номер группы\n";

while (true) {

std::cin >> stud.group;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((stud.group >= 1000) && (stud.group <= 9999))

break;

else {

std::cout << "Вы ввели неправильное число. Номер группы четырехзначный\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

std::cout << "Введите номер в группе\n";

while (true) {

std::cin >> stud.count;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((stud.count > 0) && (stud.count <= 30))

break;

else {

std::cout << "Вы ввели неправильное число. Номер в группе положительный, в группах не более 30 человек\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

std::cout << "Введите оценки по экзаменам\n";

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

while (true) {

std::cin >> stud.exam[i];

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((stud.exam[i] >= 2) && (stud.exam[i] <= 5))

break;

else {

std::cout << "Вы ввели неправильное число. оценка может быть 2, 3, 4 или 5\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

std::cout << "Введите оценки по дифзачетам\n";

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

while (true) {

std::cin >> stud.offset[i];

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((stud.offset[i] >= 2) && (stud.offset[i] <= 5))

break;

else {

std::cout << "Вы ввели неправильное число. оценка может быть 2, 3, 4 или 5\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

correct = false;

std::cout << "Введите форму обучения (O-очная, E-очно-заочная, Z-заочная):\n";

while (!correct) {

correct = true;

std::cin >> stud.study;

if ((stud.study == 'o') || (stud.study == 'e') || (stud.study == 'z'))

stud.study = stud.study + 'A' - 'a';

if ((stud.study != 'O') && (stud.study != 'E') && (stud.study != 'Z')) {

std::cout << "Некоректный ввод, достопны только O, E и Z\n";

correct = false;

}

}

std::time\_t result = std::time(nullptr);

stud.time = std::ctime(&result);

(\*students).push\_back(stud);

}

void fromFile(std::vector<Student> \*students) {

(\*students).clear();

std::ifstream in;

in.open("Students.txt");

if (in.is\_open()) {

int N;

in >> N;

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

Student stud;

(\*students).push\_back(stud);

in >> (\*students)[i].name[0] >> (\*students)[i].name[1] >> (\*students)[i].name[2];

in >> (\*students)[i].gender;

in >> (\*students)[i].group >> (\*students)[i].count;

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

in >> (\*students)[i].exam[j];

for (int j = 0; j < 5; j++)

in >> (\*students)[i].offset[j];

in >> (\*students)[i].study;

in.ignore(INT\_MAX, '\n');

std::getline(in, (\*students)[i].time);

(\*students)[i].time += '\n';

}

}

else

std::cout << "Файл не доступен\n";

in.close();

}

void showStudent(Student \*stud) {

std::cout << (\*stud).name[0] << ' ' << (\*stud).name[1] << ' ' << (\*stud).name[2] << '\n';

std::cout << (\*stud).gender << '\n';

std::cout << (\*stud).group << ' ' << (\*stud).count << '\n';

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

std::cout << (\*stud).exam[j] << ' ';

std::cout << '\n';

for (int j = 0; j < 5; j++)

std::cout << (\*stud).offset[j] << ' ';

std::cout << '\n';

std::cout << (\*stud).study << '\n';

std::cout << (\*stud).time << '\n';

}

void showAll(std::vector<Student> \*students) {

for (int i = 0; i < (\*students).size(); i++)

showStudent(&(\*students)[i]);

}

void change(std::vector<Student> \*students) {

std::cout << "Введите номер студента, информацию о котором нужно изменить\n";

for (int i = 0; i < (\*students).size(); i++) {

std::cout << i + 1 << ')';

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

std::cout << ' ' << (\*students)[i].name[j];

std::cout << " группа " << (\*students)[i].group << '\n';

}

int N;

while (true) {

std::cin >> N;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((N > 0) && (N <= (\*students).size()))

break;

else {

std::cout << "Вы ввели неправильное число\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

N--;

change(&(\*students)[N]);

save(students);

}

void change(Student \*stud) {

char answer;

bool correct = false;

std::cout << "Изменить фамилию (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (!correct) {

correct = true;

std::cin >> (\*stud).name[0];

char \* point = &(\*stud).name[0][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

(\*stud).name[0][0] = (\*stud).name[0][0] + 'A' - 'a';

}

correct = false;

std::cout << "Изменить имя (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (!correct) {

correct = true;

std::cin >> (\*stud).name[1];

char \* point = &(\*stud).name[1][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

(\*stud).name[1][0] = (\*stud).name[1][0] + 'A' - 'a';

}

correct = false;

std::cout << "Изменить отчество (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (!correct) {

correct = true;

std::cin >> (\*stud).name[2];

char \* point = &(\*stud).name[2][0];

while (\*point != 0) {

if ((\*point >= 'A') && (\*point <= 'Z'))

\*point = \*point - 'A' + 'a';

if ((\*point < 'a') || (\*point > 'z')) {

correct = false;

std::cout << "Недопустимые символы в имени, доступны только латиница\n";

break;

}

point++;

}

}

(\*stud).name[2][0] = (\*stud).name[2][0] + 'A' - 'a';

}

correct = false;

std::cout << "Изменить пол (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (!correct) {

correct = true;

std::cin >> (\*stud).gender;

if (((\*stud).gender == 'f') || ((\*stud).gender == 'm'))

(\*stud).gender = (\*stud).gender + 'A' - 'a';

if (((\*stud).gender != 'F') && ((\*stud).gender != 'M')) {

std::cout << "Некоректный ввод, достопны только F и M\n";

correct = false;

}

}

}

std::cout << "Изменить номер группы (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (true) {

std::cin >> (\*stud).group;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if (((\*stud).group >= 1000) && ((\*stud).group <= 9999))

break;

else {

std::cout << "Вы ввели неправильное число. Номер группы четырехзначный\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

std::cout << "Изменить номер в группе (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (true) {

std::cin >> (\*stud).count;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if (((\*stud).count > 0) && ((\*stud).count <= 30))

break;

else {

std::cout << "Вы ввели неправильное число. Номер в группе положительный, в группах не более 30 человек\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

std::cout << "Изменить оценки по экзаменам (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

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

while (true) {

std::cin >> (\*stud).exam[i];

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if (((\*stud).exam[i] >= 2) && ((\*stud).exam[i] <= 5))

break;

else {

std::cout << "Вы ввели неправильное число. оценка может быть 2, 3, 4 или 5\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

}

std::cout << "Изменить оценки за дифзачеты (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

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

while (true) {

std::cin >> (\*stud).offset[i];

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if (((\*stud).offset[i] >= 2) && ((\*stud).offset[i] <= 5))

break;

else {

std::cout << "Вы ввели неправильное число. оценка может быть 2, 3, 4 или 5\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

}

}

correct = false;

std::cout << "Изменить форму обучения (y/n)?\n";

std::cin >> answer;

std::cin.ignore(INT\_MAX, '\n');

if ((answer == 'y') || (answer == 'Y')) {

while (!correct) {

correct = true;

std::cin >> (\*stud).study;

if (((\*stud).study == 'o') || ((\*stud).study == 'e') || ((\*stud).study == 'z'))

(\*stud).study = (\*stud).study + 'A' - 'a';

if (((\*stud).study != 'O') && ((\*stud).study != 'E') && ((\*stud).study != 'Z')) {

std::cout << "Некоректный ввод, достопны только O, E и Z\n";

correct = false;

}

}

}

std::time\_t result = std::time(nullptr);

(\*stud).time = std::ctime(&result);

}

void showGroup(std::vector<Student> \*students) {

std::cout << "Введите номер группы\n";

int N;

while (true) {

std::cin >> N;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((N >= 1000) && (N <= 9999))

break;

else {

std::cout << "Вы ввели неправильное число. Номер группы четырехзначный\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

for (int i = 0; i < (\*students).size(); i++)

if ((\*students)[i].group == N)

showStudent(&(\*students)[i]);

}

void showGreat(std::vector<Student> \*students) {

int maxSum = 3 \* 8;

for (int i = 0; i < (\*students).size(); i++) {

int sum = 0;

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

sum += (\*students)[i].exam[j];

for (int j = 0; j < 5; j++)

sum += (\*students)[i].offset[j];

if (sum > maxSum)

maxSum = sum;

}

std::cout << "Высший средний балл " << (double)maxSum / 8 << '\n';

std::cout << "Самые успешные студенты:\n";

for (int i = 0; i < (\*students).size(); i++) {

int sum = 0;

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

sum += (\*students)[i].exam[j];

for (int j = 0; j < 5; j++)

sum += (\*students)[i].offset[j];

if (sum == maxSum)

showStudent(&(\*students)[i]);

}

}

void showGenderCount(std::vector<Student> \*students) {

int F = 0, M = 0;

for (int i = 0; i < (\*students).size(); i++) {

if ((\*students)[i].gender == 'F')

F++;

if ((\*students)[i].gender == 'M')

M++;

}

std::cout << "Количество мужчин - " << M << '\n';

std::cout << "Количество женщин - " << F << '\n';

}

void showGrant(std::vector<Student> \*students) {

bool grant, be = false;

for (int i = 0; i < (\*students).size(); i++) {

if ((\*students)[i].study != 'O')

continue;

grant = true;

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

if ((\*students)[i].exam[j] < 4) {

grant = false;

break;

}

if (!grant)

continue;

for (int j = 0; j < 5; j++)

if ((\*students)[i].offset[j] < 4) {

grant = false;

break;

}

if (grant)

if (!be) {

be = true;

std::cout << "Студенты, получающие стипендию:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

}

if (!be)

std::cout << "Упс! Кажется сессия погубила стипендию\n";

}

void showCount(std::vector<Student> \*students) {

std::cout << "Введите номер студента в группе\n";

int k;

while (true) {

std::cin >> k;

if (!std::cin) {

std::cout << "Вы неправильно ввели число. Повторите ввод\n";

std::cin.clear();

std::cin.ignore(INT\_MAX, '\n');

}

else {

if ((k > 0) && (k <= 30))

break;

else {

std::cout << "Вы ввели неправильное число. Номер в группе положительный, в группах не более 30 человек\n";

std::cin.ignore(INT\_MAX, '\n');

}

}

}

for (int i = 0; i < (\*students).size(); i++)

if ((\*students)[i].count == k)

showStudent(&(\*students)[i]);

}

void showAntiGrant(std::vector<Student> \*students) {

bool grant, be = false;

for (int i = 0; i < (\*students).size(); i++) {

if ((\*students)[i].study != 'O') {

if (!be) {

be = true;

std::cout << "Студенты, не получающие стипендию:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

continue;

}

grant = true;

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

if ((\*students)[i].exam[j] < 4) {

grant = false;

break;

}

if (!grant) {

if (!be) {

be = true;

std::cout << "Студенты, не получающие стипендию:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

continue;

}

for (int j = 0; j < 5; j++)

if ((\*students)[i].offset[j] < 4) {

grant = false;

break;

}

if (!grant)

if (!be) {

be = true;

std::cout << "Студенты, не получающие стипендию:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

}

if (!be)

std::cout << "Вау! На улице праздник, все студенты получают стипендии\n";

}

void showNedoGrant(std::vector<Student> \*students) {

bool grant, be = false;

for (int i = 0; i < (\*students).size(); i++) {

grant = true;

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

if ((\*students)[i].exam[j] < 4) {

grant = false;

break;

}

if (!grant)

continue;

for (int j = 0; j < 5; j++)

if ((\*students)[i].offset[j] < 4) {

grant = false;

break;

}

if (grant)

if (!be) {

be = true;

std::cout << "Студенты, закрывшие сессию без троек:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

}

if (!be)

std::cout << "Упс! Кажется сессия погубила даже хорошистов\n";

}

void showSuperGreat(std::vector<Student> \*students) {

bool grant, be = false;

for (int i = 0; i < (\*students).size(); i++) {

grant = true;

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

if ((\*students)[i].exam[j] < 5) {

grant = false;

break;

}

if (!grant)

continue;

for (int j = 0; j < 5; j++)

if ((\*students)[i].offset[j] < 5) {

grant = false;

break;

}

if (grant)

if (!be) {

be = true;

std::cout << "Студенты, закрывшие сессию на отлично:\n";

showStudent(&(\*students)[i]);

}

else

showStudent(&(\*students)[i]);

}

if (!be)

std::cout << "Ни одного отичника за сессию. Наконец образование работает как надо\n";

}