**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Программирование»**

**Тема: Структуры**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6382 |  | Вайгачёв А.О. |
| Преподаватель |  | Шолохова О.М. |

Санкт-Петербург

2016

**Цель работы.**

Описание структуры и ее обработка.

**Постановка задачи.**

Описание структуры, содержащую следующие поля: фамилия и инициалы; номер группы; успеваемость (массив из пяти элементов). Написать программу, выполняющую (по запросу) следующие действия:

ввод с клавиатуры данных в массив структур, записи должны быть упорядочены по возрастанию номера группы;

вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0;

если таких студентов нет, вывести соответствующее сообщение

**Спецификация программы.**

*Описание программы*.

Программа написана на языке C с использованием компилятора gcc 4.8.2. Входными данными для программы является фамилия и инициалы; номер группы; успеваемость (массив из пяти элементов), вводимое пользователем с клавиатуры. Проверка значений, вводимых пользователем отсутствует, поведение программы при некорректном вводе не определено. Выходными данными являются фамилии и номера групп для всех студентов, включенных в массив, когда средний балл студента больше 4.0 или «No students» если таких студентов нет.

*Пример диалога с пользователем*.

*Реализация*.

*Структуры*:

struct studen {

char surname[25]; - Фамилия студента

char initials[5]; - Инициалы студента

int group; - Группа студента

int progress[5]; Массив из оценок студента

};

struct StudentBase\_ {

Student\* base\_; - база ,на которой содержится струра student

size\_t size\_; - размер

size\_t capacity\_; - вместимость

}

*Функции*:

1. int readStudent( FILE\* fp,Student\* student); - считывает данные студента. На вход: указатель на файл, который содержит данные, и указатель на структуру куда считываются данные.
2. int writeStudent(FILE\* fp,const Student\* student);- выводит данные студента. На вход: указатель на файл куда считываются данные и указатель на откуда данные берутся.
3. int init(StudentBase \*base,size\_t capacity); - инициализирует базу данных. На вход: база данных и её размер.
4. int addStudent(StudentBase \*base,const Student \*student);- добавляет студента к базе данных. На вход: указатель на базу данных, и структура, содержащую информацию о студента.
5. int writeBase(FILE \*fp,const StudentBase \*base);- пишет базу данных. На вход: указатель на файл, куда будет выводится БД, указатель на БД.
6. int readBase(FILE \*fp,StudentBase \*base);-читает базу данных. На вход: указатель на файл, откуда берётся информация о БД, указатель на БД.
7. void menu(); - выводит интерактивное меню.
8. void load(StudentBase\* base);-загружает БД. На вход: указатель на БД.
9. void save(StudentBase\* base);-сохраняет БД. На вход: указатель на БД.
10. void add( StudentBase\* base);- добавляет студента к БД. На вход: указатель на БД.
11. void rem( StudentBase \*base);-удаляет студента из БД. На вход: указатель на БД.
12. void showAll(const StudentBase \*base);-покзывает список студентов в БД. На вход: указатель на БД.
13. void showAG( const StudentBase \*base);-покзывает список студентов в БД, у которых среднее арифметическое оценок >= 4.0. На вход: указатель на БД.
14. int cmp(const void \*x1,const void \*x2);-функция для qsort.
15. double alg(const int grades[5]);-считает среднее арифметичекое. На вход: массив из 5 элементов.

**Выводы.**

Мы описали структуры и ее обработали ее.

**Приложение А. Исходный код. main.c**

#include <stdio.h>

#include <stdlib.h>

#include "studentbase.h"

void menu();

void load(StudentBase\* base);

void save(StudentBase\* base);

void add( StudentBase\* base);

void rem( StudentBase \*base);

void showAll(const StudentBase \*base);

void showAG( const StudentBase \*base);

int cmp(const void \*x1,const void \*x2);

int main()

{

StudentBase base;

init(&base,100);

int menu\_option;

while(1){

qsort(base.base\_,base.size\_,sizeof(Student),cmp);

menu();

scanf("%i",&menu\_option);

fflush(stdin);

switch (menu\_option) {

case 1:

load(&base);

break;

case 2:

save(&base);

break;

case 3:

add( &base);

break;

case 4:

showAll(&base);

break;

case 5:

showAG(&base);

break;

case 6:

rem(&base);

break;

case 7:

exit(EXIT\_SUCCESS);

default:

printf("Error:Incorrect command\n");

}

}

return 0;

}

int cmp(const void\* x1,const void \*x2){

Student y1 = \*(Student\*)x1;

Student y2 = \*(Student\*)x2;

return y1.group - y2.group;

}

void menu(){

printf("");

printf("---------------------------------------------\n");

printf("Lab1. Vaigachev A., 6382\n");

printf("---------------------------------------------\n");

printf("1. Load base from file\n");

printf("2. Save base to file\n");

printf("3. Add student\n");

printf("4. Show all students\n");

printf("5. Show students with average grades >= 4.0\n");

printf("6. Remove student from all\n");

printf("7. Exit\n");

printf("---------------------------------------------\n");

printf("");

}

void load(StudentBase\* base){

FILE\* fp;

fp = fopen("data.txt","r");

if(fp == NULL){

printf("Error: File not find!\n");

return;

}

if(readBase(fp,base) == -1){

printf("Error: error while load base!\n");

};

fclose(fp);

}

void save(StudentBase\* base){

FILE\* fp;

fp = fopen("data.txt","w");

if (fp == NULL){

printf("Error:Cannot open file!\n");

return;

}

writeBase(fp, base);

fclose(fp);

}

void add(StudentBase\* base){

Student student;

printf("Enter student data:");

readStudent(stdin,&student);

if(addStudent(base,&student) != 0){

printf("Error: error while add student to base!\n");

return;

}

}

void showAll(const StudentBase\* base){

int flag = 0;

for(size\_t i = 0; i < base->size\_; i++){

if(flag == 0) printf("\n# Surname init group grades\n");

printf("%d",i+1);

writeStudent(stdout,&base->base\_[i]);

flag = 1;

}

if(flag == 0) printf("-----------\nNo students\n-----------\n");

}

double alg(const int grades[5]);

void showAG(const StudentBase\* base){

int flag = 0;

printf("\n# Surname init group grades\n");

for(size\_t i = 0; i < base->size\_; i++){

if(alg(&base->base\_[i].grades) >= 4.0){

if(flag == 0) printf("\n Surname init group grades\n");

printf("%d",i+1);

writeStudent(stdout,&base->base\_[i]);

flag = 1;

}

}

if(flag == 0) printf("-----------\nNo students\n-----------\n");

}

double alg(const int grades[5]){

return (grades[0]+grades[1]+grades[2]+grades[3]+grades[4]) / 4.0 ;

}

void rem( StudentBase \*base){

printf("Which number from all you want to remove: ");

int i;

printf("\n");

if(base->size\_ > 0 && scanf("%d",&i) == 1 && base->size\_ >= i){

base->base\_[i-1] = base->base\_[base->size\_-1];

base->size\_ -= 1;

return;

}

else{

printf("Error: Nothing to delete!\n");

return;

}

}