МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра информационно-аналитических систем безопасности

имени Л.С. Берштейна

**ОТЧЕТ**

**Лабораторная работа №5**

**Основы алгоритмизации и программирования**

**«Работа со структурами»**

**Вариант №5**

Выполнил:

студент гр. КТбо1-8

Герасименко В.А.

«\_\_\_» 2017 г.

Проверил:

Профессор кафедры ИАСБ

Беляков Станислав Леонидович

«\_\_\_» 2017 г.

Таганрог – 2017

Оглавление

[Цели работы 3](#_Toc499755574)

[Алгоритм выполнения работы 4](#_Toc499755575)

[Создание алгоритма 4](#_Toc499755576)

[Программный код 5](#_Toc499755577)

[Тестирование программы 7](#_Toc499755578)

[Вывод 8](#_Toc499755579)

# Цели работы

Цель данной лабораторной работы ознакомить студента с понятием структуры данных, научить его правильно использовать ее и работать с ней.

Задание. Написать программу, которая записывает с клавиатуры в файл структуру

согласно выданному варианту задания. В качестве разделителя полей структуры

использовать символ табуляции. В программе реализовать:

а) дополнение существующего массива структур новыми структурами;

б) поиск структуры с заданным значением выбранного элемента;

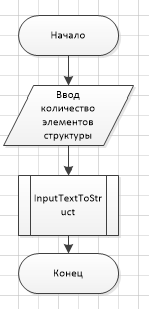
в) вывод на экран содержимого массива структур;

г) упорядочение массива структур по заданному полю (элементу), например

государство по численности.

Вариант задания: 5«Студент»: фамилия, имя, отчество, дата рождения, факультет, кафедра, группа, домашний адрес, успеваемость (плохо, удовлетворительно, хорошо, отлично).

# Алгоритм выполнения работы



# Программный код

/\*

Герасименко Владимир Александрович

Лабораторная работа №5.

Вариант №5

Задание: Написать программу, которая записывает с клавиатуры в файл структуру согласно выданному варианту задания. В качестве разделителя полей структуры использовать символ табуляции. В программе реализовать:

а) дополнение существующего массива структур новыми структурами;

б) поиск структуры с заданным значением выбранного элемента;

в) вывод на экран содержимого массива структур;

г) упорядочение массива структур по заданному полю (элементу), например государство по численности.

«Студент»: фамилия, имя, отчество, дата рождения, факультет, кафедра, группа, домашний адрес, успеваемость (плохо, удовлетворительно, хорошо, отлично).

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#include<locale.h>

#define SIZE 30

#define MAX\_STUDENT 100

struct student

{

char surname[SIZE];

char name[SIZE];

char patronymic[SIZE];

char birthDate[SIZE];

char faculty[SIZE];

char department[SIZE];

char group[SIZE];

char homeAdress[SIZE];

char performance[SIZE];

} student\_list[MAX\_STUDENT];

int menu\_select(void);

void enter(void), search(void), printList(void),sorting(void);

void menu(void), printConsole(int J),printInFile(int J),swapStruct(int J);

void surname(void), name(void), patronymic(void), birthDate(void), faculty(void), department(void), group(void), homeAdress(void), performance(void);

void sortingSurname(void), sortingName(void), sortingPatronymic(void), sortingBirthDate(void), sortingDepartment(void), sortingFaculty(void), sortingGroup(void), sortingHomeAdress(void), sortingPerformance(void);

int item,n,J;

void main()

{

system("chcp 1251");

system("cls");

menu();

system("PAUSE");

}

void menu()

{

item = menu\_select();

switch (item)

{

case 1: enter(); break;

case 2: search(); break;

case 3: printList(); break;

case 4: sorting(); break;

case 5: exit(0); break;

}

}

int menu\_select()

{

L: printf("1.Добавить студента\n2.Поиск студентов по данным\n3.Вывод студентов\n4.Упорядочивание(сортировка) массива по данным\n5.Выход\n----------------");

printf("Введите пункт: ");

scanf("%d", &item);

if ((item>=0)&&(item<=5))

return(item);

else goto L;

}

void enter(void)

{

int i = 0;

printf("Введите кол-во вписываемых студентов: ");

scanf\_s("%d", &n);

for (i; i < n; i++)

{

printf("Студент № %d----------\n ", i + 1);

printf("Введите фамилию студента: ");

scanf("%s", &student\_list[i].surname);

printf("Введите имя студента: ");

scanf("%s", &student\_list[i].name);

printf("Введите отчество студента: ");

scanf("%s", &student\_list[i].patronymic);

printf("Введите дату рождения студента через слеш(GGGG/MM/DD): ");

scanf("%s", &student\_list[i].birthDate);

printf("Введите факультет студента: ");

scanf("%s", &student\_list[i].faculty);

printf("Введите кафедру студента: ");

scanf("%s", &student\_list[i].department);

printf("Введите группу студента: ");

scanf("%s", &student\_list[i].group);

printf("Введите домашний адресс студента: ");

scanf("%s", &student\_list[i].homeAdress);

printf("Введите успеваемость студента (плохо, удовлетворительно, хорошо, отлично): ");

scanf("%s", &student\_list[i].performance);

printf("\n\n");

}

menu();

}

void search(void)

{

int result;

printf("Введите по каким данным вы хотите отсортировать студентов:\n1.По фамилии\n2.По Имени\n3.По отчеству\n4.По дате рождения\n5.По факультету\n6.По кафедре\n7.По группе\n8.По домашний адрессу\n9.По успеваемость студента (плохо, удовлетворительно, хорошо, отлично)\n");

printf("Введите пункт: ");

scanf("%d", &result);

if ((result >= 0) && (result <= 9))

{

switch (result)

{

case 1: surname(); break;

case 2: name(); break;

case 3: patronymic(); break;

case 4: birthDate(); break;

case 5: faculty(); break;

case 6: department(); break;

case 7: group(); break;

case 8: homeAdress(); break;

case 9: performance(); break;

}

}

}

void printList(void)

{

for (J = 0; J < n; J++)

{

printf("Студент № %d ----------\n", J + 1);

printConsole(J);

}

printf("\n\n");

system("PAUSE");

menu();

}

void sorting(void)

{

int result;

printf("Введите по каким данным вы хотите упорядочить студентов:\n1.По фамилии\n2.По Имени\n3.По отчеству\n4.По дате рождения\n5.По факультету\n6.По кафедре\n7.По группе\n8.По домашний адрессу\n9.По успеваемость студента (плохо, удовлетворительно, хорошо, отлично)\n");

printf("Введите пункт: ");

scanf("%d", &result);

if ((result >= 0) && (result <= 9))

{

switch (result)

{

case 1: sortingSurname(); break;

case 2: sortingName(); break;

case 3: sortingPatronymic(); break;

case 4: sortingBirthDate(); break;

case 5: sortingDepartment(); break;

case 6: sortingFaculty(); break;

case 7: sortingGroup(); break;

case 8: sortingHomeAdress(); break;

case 9: sortingPerformance(); break;

}

}

}

void printConsole(int J)

{

printf("Фамилия: %s\n", student\_list[J].surname);

printf("Имя: %s\n", student\_list[J].name);

printf("Отчество: %s\n", student\_list[J].patronymic);

printf("Дата рождения: %s\n", student\_list[J].birthDate);

printf("Факультет: %s\n", student\_list[J].faculty);

printf("Кафедра: %s\n", student\_list[J].department);

printf("Группа: %s\n", student\_list[J].group);

printf("Домашний адресс: %s\n", student\_list[J].homeAdress);

printf("Успеваемость: %s\n", student\_list[J].performance);

printf("\n\n");

printInFile(J);

}

void printInFile(int J)

{

FILE \*writeFile;

writeFile = fopen("writeFile", "а");

fprintf(writeFile, "Фамилия: %s\n", student\_list[J].surname);

fprintf(writeFile, "Имя: %s\n", student\_list[J].name);

fprintf(writeFile, "Отчество: %s\n", student\_list[J].patronymic);

fprintf(writeFile, "Дата рождения: %s\n", student\_list[J].birthDate);

fprintf(writeFile, "Факультет: %s\n", student\_list[J].faculty);

fprintf(writeFile, "Кафедра: %s\n", student\_list[J].department);

fprintf(writeFile, "Группа: %s\n", student\_list[J].group);

fprintf(writeFile, "Домашний адресс: %s\n", student\_list[J].homeAdress);

fprintf(writeFile, "Успеваемость: %s\n", student\_list[J].performance);

fclose(writeFile);

printf("Данные успешно записанны в файл!\n");

}

void surname(void)

{

char surname[SIZE];

printf("Введите фамилию: ");

scanf("%s", &surname);

for (J = 0; J < n; J++)

{

if ((strcmp(surname,student\_list[J].surname))==0)

{

printConsole(J);

}

}

}

void name(void)

{

char name[SIZE];

printf("Введите имя");

scanf("%s", &name);

for (J = 0; J < n; J++)

{

if ((strcmp(name, student\_list[J].name)) == 0)

{

printConsole(J);

}

}

}

void patronymic(void)

{

char patronymic[SIZE];

printf("Введите отчество: ");

scanf("%s", &patronymic);

for (J = 0; J < n; J++)

{

if ((strcmp(patronymic, student\_list[J].patronymic)) == 0)

{

printConsole(J);

}

}

}

void birthDate(void)

{

char birthDate[SIZE];

printf("Введите дату рождения: ");

scanf("%s", &birthDate);

for (J = 0; J < n; J++)

{

if ((strcmp(birthDate, student\_list[J].patronymic)) == 0)

{

printConsole(J);

}

}

}

void faculty(void)

{

char faculty[SIZE];

printf("Введите факультет: ");

scanf("%s", &faculty);

for (J = 0; J < n; J++)

{

if ((strcmp(faculty, student\_list[J].patronymic)) == 0)

{

printConsole(J);

}

}

}

void department(void)

{

char department[SIZE];

printf("Введите кафедру");

scanf("%s", &department);

for (J = 0; J < n; J++)

{

if ((strcmp(department, student\_list[J].patronymic)) == 0)

{

printConsole(J);

}

}

}

void group(void)

{

char group[SIZE];

printf("Введите группу");

scanf("%s", &group);

for (J = 0; J < n; J++)

{

if ((strcmp(group, student\_list[J].group)) == 0)

{

printConsole(J);

}

}

}

void homeAdress(void)

{

char homeAdress[SIZE];

printf("Введите домашний адресс");

scanf("%s", &homeAdress);

for (J = 0; J < n; J++)

{

if ((strcmp(homeAdress, student\_list[J].homeAdress)) == 0)

{

printConsole(J);

}

}

}

void performance(void)

{

char performance[SIZE];

printf("Введите успеваемость студента (плохо, удовлетворительно, хорошо, отлично");

scanf("%s", &performance);

for (J = 0; J < n; J++)

{

if ((strcmp(performance, student\_list[J].performance)) == 0)

{

printConsole(J);

}

}

}

void swapStruct(int J)

{

student buf;

buf = student\_list[J];

student\_list[J] = student\_list[J + 1];

student\_list[J + 1] = buf;

}

void sortingSurname()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].surname, student\_list[J + 1].surname) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingName()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].name, student\_list[J + 1].name) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingPatronymic()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].patronymic, student\_list[J + 1].patronymic) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingBirthDate()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].birthDate, student\_list[J + 1].birthDate) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingDepartment()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].department, student\_list[J + 1].department) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingFaculty()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].faculty, student\_list[J + 1].faculty) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingGroup()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].group, student\_list[J + 1].group) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingHomeAdress()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].homeAdress, student\_list[J + 1].homeAdress) != 0)

{

swapStruct(J);

}

}

menu();

}

void sortingPerformance()

{

for (J = 0; J < (n - 1); J++)

{

if (strcmp(student\_list[J].performance, student\_list[J + 1].performance) != 0)

{

swapStruct(J);

}

}

menu();

}

# Тестирование программы

Пример входных и выходных данных, представлены на рисунках 1 и 2

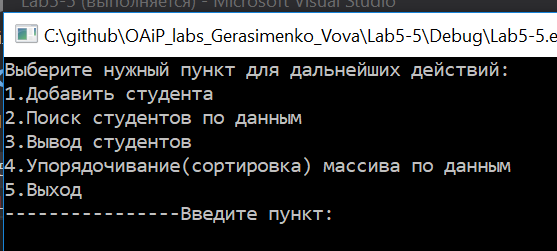


рис. 1 Главное меню

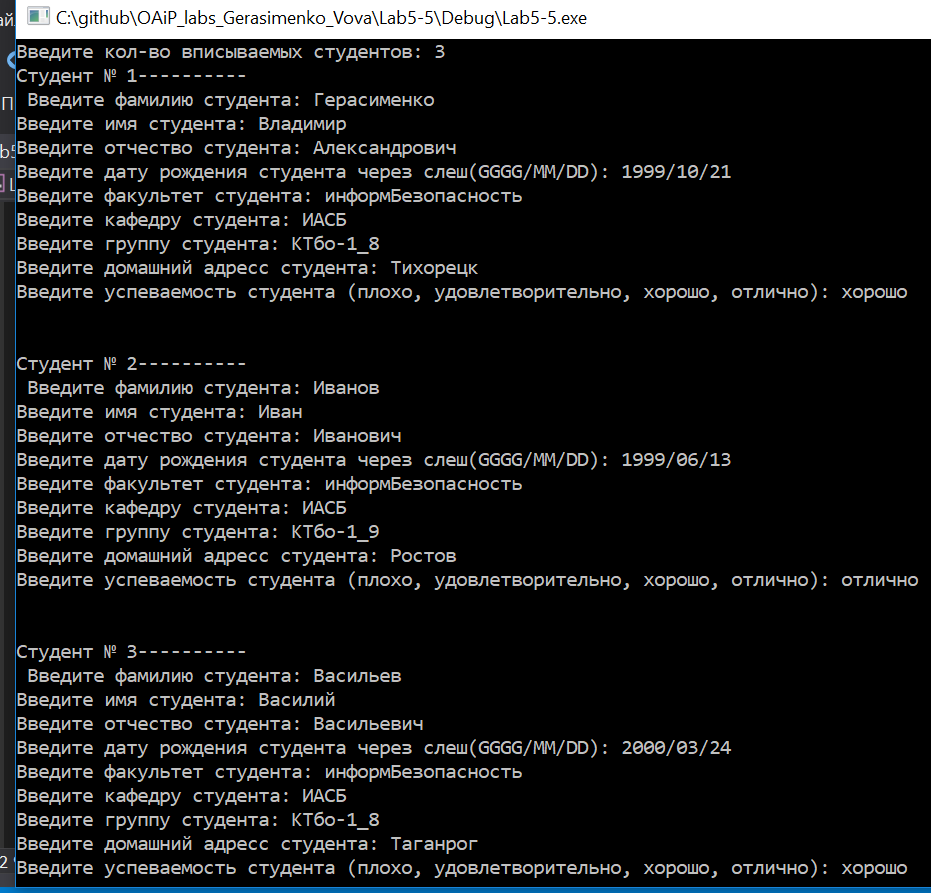


рис. 2Ввод данных о студенте

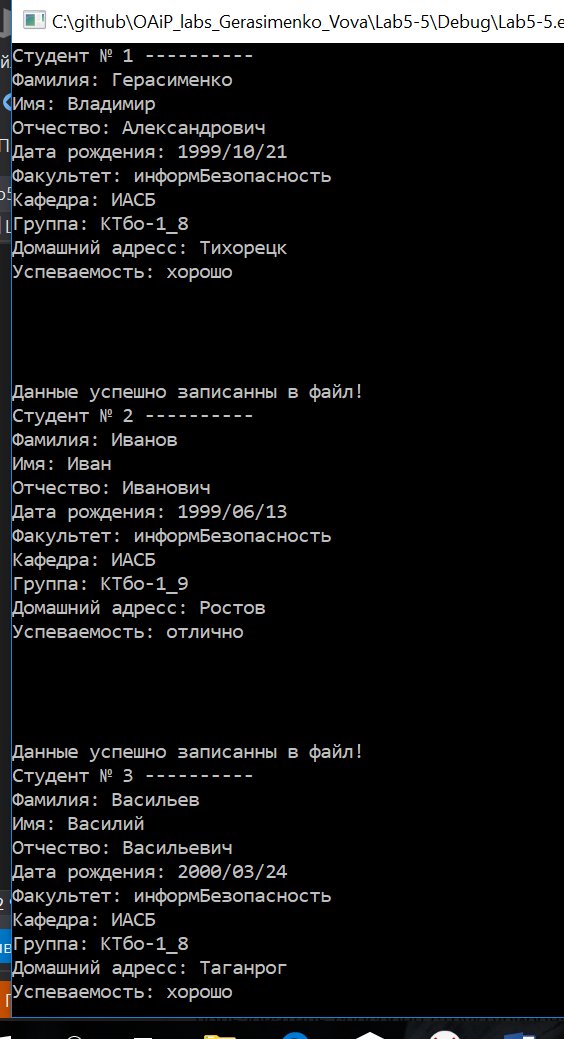


рис. 3 Вывод списка студентов на экран

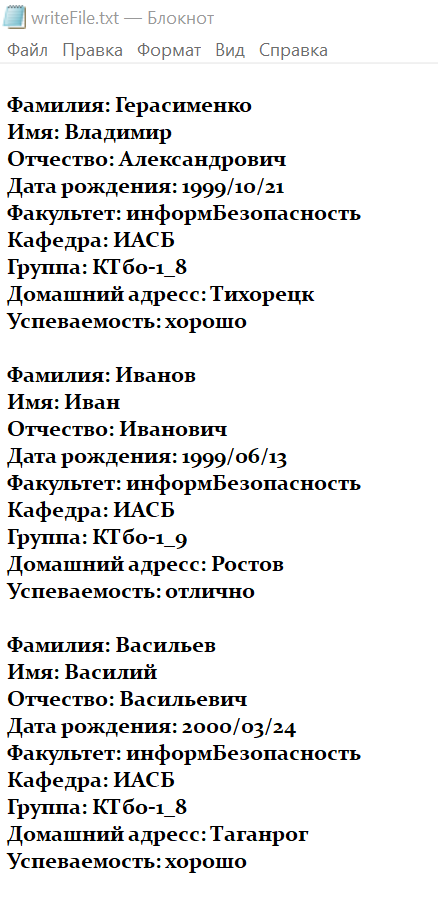


рис. 4 Вывод списка студентов в файл

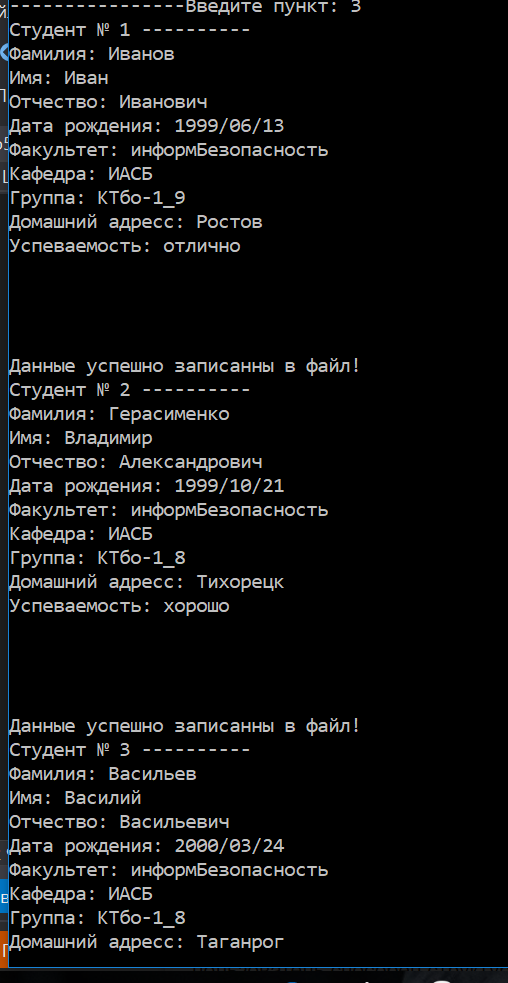
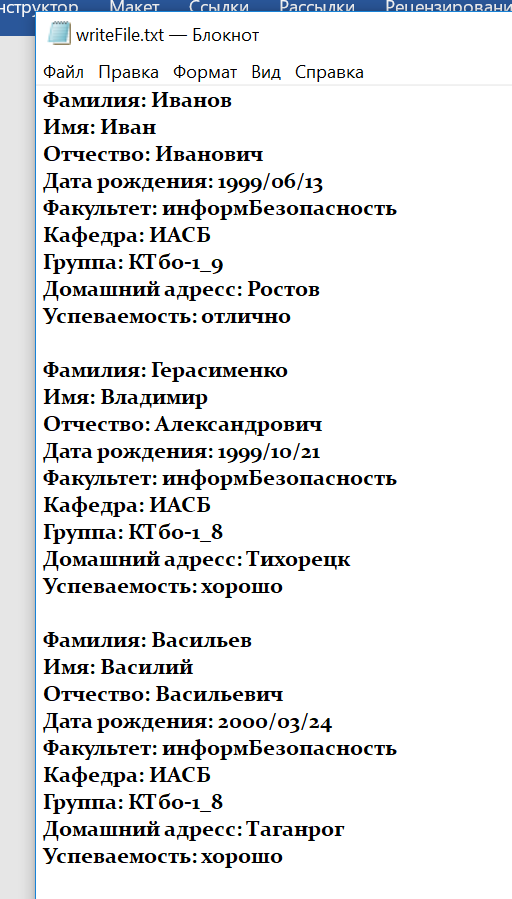
 

рис. 5 Результат работы программы, когда применили сортировку по успеваемости

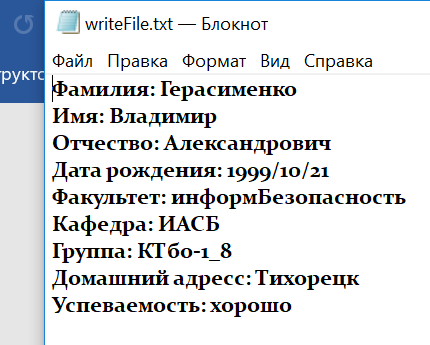
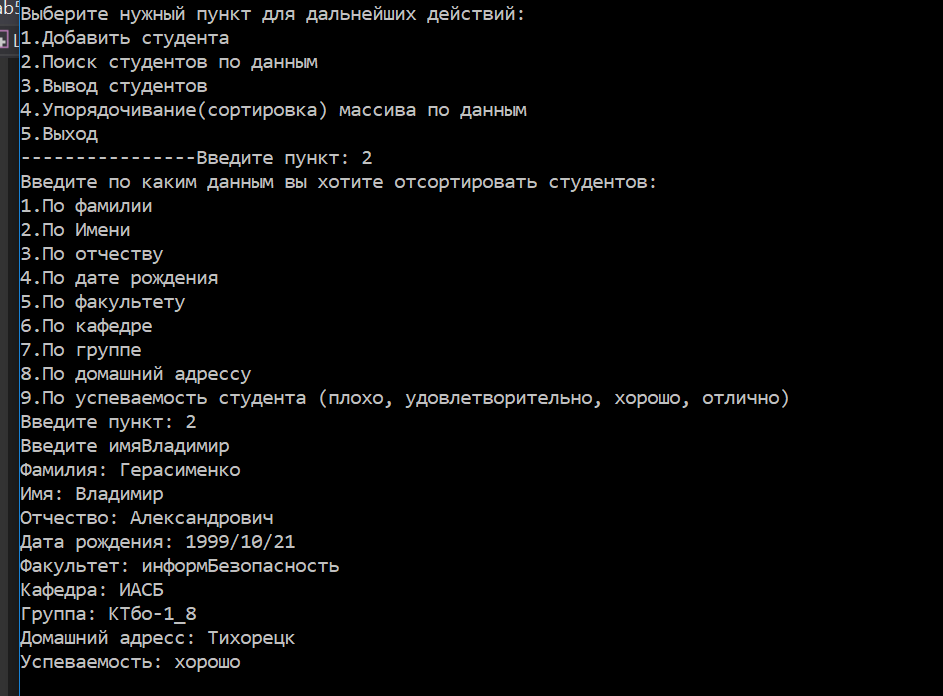


рис. 6 Результат работы программы, когда применён поиск по имени студента "Владимир"

# Вывод

Выполняя данную лабораторной работы, мы ознакомились с понятием структуры, научились правильно использовать и работать с ней.