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 №1**

Выполнил:

студент гр. КТбо1-8

Елоев Георгий

«\_\_\_» 2017 г.

Проверил:

Профессор кафедры ИАСБ

Беляков Станислав Леонидович

«\_\_\_» 2017 г.

Таганрог – 2017

Оглавление

[Цели работы 3](#_Toc501565970)

[Алгоритм выполнения работы 4](#_Toc501565971)

[Создание алгоритма 4](#_Toc501565972)

[Программный код 5](#_Toc501565973)

[Тестирование программы 20](#_Toc501565974)

[Вывод 22](#_Toc501565975)

# Цели работы

Цель данной лабораторной работы состоит в том, чтобы научить студентов работать со сложными типами данных, каковыми являются структуры данных.

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.

Варианты заданий:

1. «Человек»: фамилия, имя, пол, рост, вес, датарождения (число, месяц, год), номер телефона, домашний адрес (индекс, страна, область, город, улица, дом, квартира).

# Алгоритм выполнения работы

# **Создание алгоритма**



# Программный код

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <conio.h>

#include <locale>

#define MAX\_LENGTH 1024

#define STRUCTURE\_LENGTH 512

struct Struct

{

char surname[MAX\_LENGTH];

char name[MAX\_LENGTH];

char sex[MAX\_LENGTH];

char growth[MAX\_LENGTH];

char weight[MAX\_LENGTH];

char date\_number[MAX\_LENGTH];

char date\_month[MAX\_LENGTH];

char date\_year[MAX\_LENGTH];

char phone\_number[MAX\_LENGTH];

char address\_index[MAX\_LENGTH];

char address\_country[MAX\_LENGTH];

char address\_region[MAX\_LENGTH];

char address\_city[MAX\_LENGTH];

char address\_street[MAX\_LENGTH];

char address\_house[MAX\_LENGTH];

char address\_apartment[MAX\_LENGTH];

};

int initialization(Struct \*People);

void input(Struct \*People, int i);

void output(Struct \*People, int i);

void print(Struct \*People, int number);

void new\_input(Struct \*People, int number);

int printMenu();

void inputNewEntry(Struct \*People, int number);

int printMenuSearch();

int compareByField(Struct \*People, int field, int i, char \*Search);

void searchByField(Struct \*People, int number, int struc, int field);

void searchMenu(Struct \*People, int search, int number, int struc);

void outputEntry(Struct \*People, int number);

int printMenuSroting();

int compareByField1(Struct \*People, int field, int i);

void sortByField(Struct \*People, int number, int field);

void sortMenu(Struct \*People, int sorting, int number);

void main()

{

system("chcp 1251");

system("cls");

Struct \*People = new Struct[STRUCTURE\_LENGTH];

Struct \*\*q = new Struct\*[STRUCTURE\_LENGTH];

int main\_screen = 0;

int number\_of\_structures = initialization(People);

int sorting = 0;

int search = 0;

int i = 0;

int struc = 0;

int code;

do

{

main\_screen = printMenu();

switch (main\_screen)

{

case 1: inputNewEntry(People, number\_of\_structures); break;

case 2: searchMenu(People, search, number\_of\_structures, struc); break;

case 3: outputEntry(People, number\_of\_structures); break;

case 4: sortMenu(People, sorting, number\_of\_structures); break;

case 5: exit(0); //выход и программы break;

default: printf("Такого варианта выбора нет.\n");

}

} while (true);

system("pause");

}

int printMenu()

{

int menu;

printf("---Введите номер---\n");

printf("1. Дополнить новой записью структуру.\n");

printf("2. Поиск структуры по заданным значениям выбранного элемента.\n");

printf("3. Вывод на экран содержимого структур.\n");

printf("4. Упорядочение структур по заданному полю.\n");

printf("5. Выход из программы.\n");

printf("--> ");

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

system("cls");

return menu;

}

void inputNewEntry(Struct \*People, int number)

{

printf("--- Дополнить новой записью структуру ---\n");

if (number < STRUCTURE\_LENGTH) //проверка

{

input(People, number);

number++;

system("pause");

system("cls");

}

else

{

printf("--- Введенно максимально возможное количество структур ---");

}

}

int printMenuSearch()

{

int sear;

printf("---Введите номер---\n");

printf("---Поиск структуры по заданным значениям выбранного элемента--- \n");

printf("1. Поиск по Фамилии.\n");

printf("2. Поиск по Имени.\n");

printf("3. Поиск по Полу.\n");

printf("4. Поиск по Росту.\n");

printf("5. Поиск по Весу.\n");

printf("6. Поиск по Числу рождения.\n");

printf("7. Поиск по Месяцу рождения.\n");

printf("8. Поиск по Году рождения.\n");

printf("9. Поиск по Номер телефона.\n");

printf("10. Поиск по Индексу.\n");

printf("11. Поиск по Стране.\n");

printf("12. Поиск по Региону.\n");

printf("13. Поиск по Городу.\n");

printf("14. Поиск по Улице.\n");

printf("15. Поиск по Номеру дома.\n");

printf("16. Поиск по Номеру квартиры.\n");

printf("--> ");

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

system("cls");

return sear;

}

int initialization(Struct \*People)

{

FILE \*File = fopen("Struct.txt", "r");

FILE \*File1 = fopen("Struct.txt", "r");

int i = 0;

if (File != NULL)

{

while (fscanf(File, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s", &People[i].surname, &People[i].name, &People[i].sex, &People[i].growth, &People[i].weight, &People[i].date\_number, &People[i].date\_month, &People[i].date\_year, &People[i].phone\_number, &People[i].address\_index, &People[i].address\_country, &People[i].address\_region, &People[i].address\_city, &People[i].address\_street, &People[i].address\_house, &People[i].address\_apartment) != EOF)

{

fscanf(File1, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s", &People[i].surname, &People[i].name, &People[i].sex, &People[i].growth, &People[i].weight, &People[i].date\_number, &People[i].date\_month, &People[i].date\_year, &People[i].phone\_number, &People[i].address\_index, &People[i].address\_country, &People[i].address\_region, &People[i].address\_city, &People[i].address\_street, &People[i].address\_house, &People[i].address\_apartment);

i++;

}

}

else

{

printf("---Файла нет или не открылся---\n");

system("pause");

exit(0);

}

fclose(File);

fclose(File1);

return i;

}

void input(Struct \*People, int i)

{

printf("---Введите свои данный---\n");

printf("Введите свою фамилию: ");

scanf("%s", &People[i].surname);

printf("Введите своё имя: ");

scanf("%s", &People[i].name);

printf("Укажите пол: ");

scanf("%s", &People[i].sex);

printf("Введите сой рост: ");

scanf("%s", &People[i].growth);

printf("Введите свой вес: ");

scanf("%s", &People[i].weight);

printf("Введите число рождения: ");

scanf("%s", &People[i].date\_number);

printf("Введите месяц рождения(цифрами): ");

scanf("%s", &People[i].date\_month);

printf("Введите год рождения: ");

scanf("%s", &People[i].date\_year);

printf("Введите свой номер телефона: ");

scanf("%s", &People[i].phone\_number);

printf("Введите ваш домашний индекс: ");

scanf("%s", &People[i].address\_index);

printf("Введите вашу страну: ");

scanf("%s", &People[i].address\_country);

printf("Введите свою область или регион: ");

scanf("%s", &People[i].address\_region);

printf("Введите свой город: ");

scanf("%s", &People[i].address\_city);

printf("Введите свою улицу: ");

scanf("%s", &People[i].address\_street);

printf("Введите номер дома: ");

scanf("%s", &People[i].address\_house);

printf("Введите номер квартиры(если дом не квартирный то поставь 0): ");

scanf("%s", &People[i].address\_apartment);

printf("Конец ввода.\n\n");

FILE \*File = fopen("Struct.txt", "a");

fprintf(File, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n", &People[i].surname, &People[i].name, &People[i].sex, &People[i].growth, &People[i].weight, &People[i].date\_number, &People[i].date\_month, &People[i].date\_year, &People[i].phone\_number, &People[i].address\_index, &People[i].address\_country, &People[i].address\_region, &People[i].address\_city, &People[i].address\_street, &People[i].address\_house, &People[i].address\_apartment);

fclose(File);

}

void output(Struct \*People, int i)

{

printf("Фамилия: %s\t", &People[i].surname);

printf("Имя: %s\t", &People[i].name);

printf("Пол: %s\t", &People[i].sex);

printf("Рост: %s\t", &People[i].growth);

printf("Все: %s\t", &People[i].weight);

printf("Дата рождения: %s.%s.%s\t", &People[i].date\_number, &People[i].date\_month, &People[i].date\_year);

printf("Номер телефона: %s\t", &People[i].phone\_number);

printf("Индекс: %s\t", &People[i].address\_index);

printf("Страна: %s\t", &People[i].address\_country);

printf("Регион: %s\t", &People[i].address\_region);

printf("Город: %s\t", &People[i].address\_city);

printf("Адрес: ул.%s дом №%s кв.№%s\n\n", &People[i].address\_street, &People[i].address\_house, &People[i].address\_apartment);

}

void print(Struct \*People, int number)

{

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

{

output(People, i);

}

}

void new\_input(Struct \*People, int number)

{

FILE \*File = fopen("Sort.txt", "w");

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

{

fprintf(File, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n", &People[i].surname, &People[i].name, &People[i].sex, &People[i].growth, &People[i].weight, &People[i].date\_number, &People[i].date\_month, &People[i].date\_year, &People[i].phone\_number, &People[i].address\_index, &People[i].address\_country, &People[i].address\_region, &People[i].address\_city, &People[i].address\_street, &People[i].address\_house, &People[i].address\_apartment);

}

fclose(File);

}

int compareByField(Struct \*People, int field, int i, char \*Search)

{

int code;

switch (field) {

case 1:code = strcmp(Search, People[i].surname); return code;

case 2:code = strcmp(Search, People[i].name); return code;

case 3:code = strcmp(Search, People[i].sex); return code;

case 4:code = strcmp(Search, People[i].growth); return code;

case 5:code = strcmp(Search, People[i].weight); return code;

case 6:code = strcmp(Search, People[i].date\_number); return code;

case 7:code = strcmp(Search, People[i].date\_month); return code;

case 8:code = strcmp(Search, People[i].date\_year); return code;

case 9:code = strcmp(Search, People[i].phone\_number); return code;

case 10:code = strcmp(Search, People[i].address\_index); return code;

case 11:code = strcmp(Search, People[i].address\_country); return code;

case 12:code = strcmp(Search, People[i].address\_region); return code;

case 13:code = strcmp(Search, People[i].address\_city); return code;

case 14:code = strcmp(Search, People[i].address\_street); return code;

case 15:code = strcmp(Search, People[i].address\_house); return code;

case 16:code = strcmp(Search, People[i].address\_apartment); return code;

}

}

void searchByField(Struct \*People, int number, int struc, int field)

{

char Search[MAX\_LENGTH];

scanf("%s", &Search);

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

{

if ((compareByField(People, field, i, Search)) == 0)

{

struc++;

printf("Струра найденна\n Номер структуры -> %d\n", i + 1);

output(People, i);

}

}

if (struc == 0)

{

printf("Структура не найденна\n");

}

struc = 0;

system("pause");

system("cls");

}

void searchMenu(Struct \*People, int search, int number, int struc)

{

if (number != 0)

{

search = printMenuSearch();

switch (search)

{

case 1: printf(" --- Поиск по Фамилии --- \nВведите Фамилию\n -->");

searchByField(People, number, struc, search);

break;

case 2:printf(" --- Поиск по Имени --- \nВведите Имя\n -->");

searchByField(People, number, struc, search);

break;

case 3:printf(" --- Поиск по Полу --- \nВведите Пол\n -->");

searchByField(People, number, struc, search);

break;

case 4:printf(" --- Поиск по Росту --- \nВведите Рост\n -->");

searchByField(People, number, struc, search);

break;

case 5:printf(" --- Поиск по Весу --- \nВведите Вес\n -->");

searchByField(People, number, struc, search);

break;

case 6:printf(" --- Поиск по Числу рождения --- \nВведите Число\n -->");

searchByField(People, number, struc, search);

break;

case 7:printf(" --- Поиск по Месяцу рождения --- \nВведите Месяц\n -->");

searchByField(People, number, struc, search);

break;

case 8:printf(" --- Поиск по Году рождения --- \nВведите Год\n -->");

searchByField(People, number, struc, search);

break;

case 9:printf(" --- Поиск по Номеру телефона --- \nВведите Номер\n -->");

searchByField(People, number, struc, search);

break;

case 10:printf(" --- Поиск по Индексу --- \nВведите Индекс\n -->");

searchByField(People, number, struc, search);

break;

case 11:printf(" --- Поиск по Стране --- \nВведите Страну\n -->");

searchByField(People, number, struc, search);

break;

case 12:printf(" --- Поиск по Региону --- \nВведите Регион\n -->");

searchByField(People, number, struc, search);

break;

case 13:printf(" --- Поиск по Городу --- \nВведите Город\n -->");

searchByField(People, number, struc, search);

break;

case 14:printf(" --- Поиск по Улице --- \nВведите Улицу\n -->");

searchByField(People, number, struc, search);

break;

case 15:printf(" --- Поиск по Номеру дома --- \nВведите Дом\n -->");

searchByField(People, number, struc, search);

break;

case 16:printf(" --- Поиск по Номеру квартиры --- \nВведите Квартиру\n -->");

searchByField(People, number, struc, search);

break;

default: printf("Такого варианта выбора нет.\n"); system("pause");

}

}

else

{

printf("--- Заполненных структур нет ---\n");

system("pause");

}

system("cls");

}

void outputEntry(Struct \*People, int number)

{

printf("---Вывод на экран содержимого структур---\n");

if (number == 0)

{

printf("Заполненных структур нет\n");

}

else

{

print(People, number);

}

system("pause");

system("cls");

}

int printMenuSroting()

{

int sorting;

printf(" ---Упорядочение структур по заданному полю--- \n");

printf("(введите номер)\n");

printf("--- Сортировка по заданному полю --- \n");

printf("1. Сортировка по Фамилии.\n");

printf("2. Сортировка по Имени.\n");

printf("3. Сортировка по Полу.\n");

printf("4. Сортировка по Росту.\n");

printf("5. Сортировка по Весу.\n");

printf("6. Сортировка по Числу рождения.\n");

printf("7. Сортировка по Месяцу рождения.\n");

printf("8. Сортировка по Году рождения.\n");

printf("9. Сортировка по Номер телефона.\n");

printf("10. Сортировка по Индексу.\n");

printf("11. Сортировка по Стране.\n");

printf("12. Сортировка по Региону.\n");

printf("13. Сортировка по Городу.\n");

printf("14. Сортировка по Улице.\n");

printf("15. Сортировка по Номеру дома.\n");

printf("16. Сортировка по Номеру квартиры.\n");

printf("--> ");

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

system("cls");

return sorting;

}

int compareByField1(Struct \*People, int field, int i)

{

int code;

switch (field) {

case 1:code = strcmp(People[i + 1].surname, People[i].surname); return code;

case 2:code = strcmp(People[i + 1].name, People[i].name); return code;

case 3:code = strcmp(People[i + 1].sex, People[i].sex); return code;

case 4:code = strcmp(People[i + 1].growth, People[i].growth); return code;

case 5:code = strcmp(People[i + 1].weight, People[i].weight); return code;

case 6:code = strcmp(People[i + 1].date\_number, People[i].date\_number); return code;

case 7:code = strcmp(People[i + 1].date\_month, People[i].date\_month); return code;

case 8:code = strcmp(People[i + 1].date\_year, People[i].date\_year); return code;

case 9:code = strcmp(People[i + 1].phone\_number, People[i].phone\_number); return code;

case 10:code = strcmp(People[i + 1].address\_index, People[i].address\_index); return code;

case 11:code = strcmp(People[i + 1].address\_country, People[i].address\_country); return code;

case 12:code = strcmp(People[i + 1].address\_region, People[i].address\_region); return code;

case 13:code = strcmp(People[i + 1].address\_city, People[i].address\_city); return code;

case 14:code = strcmp(People[i + 1].address\_street, People[i].address\_street); return code;

case 15:code = strcmp(People[i + 1].address\_house, People[i].address\_house); return code;

case 16:code = strcmp(People[i + 1].address\_apartment, People[i].address\_apartment); return code;

}

}

void sortByField(Struct \*People, int number, int field)

{

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

{

for (int j = 0; j < number - i - 1; j++)

{

if ((compareByField1(People, field, j)) < 0)

{

Struct Buff = People[j];

People[j] = People[j + 1];

People[j + 1] = Buff;

}

}

}

print(People, number);

new\_input(People, number);

system("pause");

system("cls");

}

void sortMenu(Struct \*People, int sorting, int number)

{

if (number != 0)

{

sorting = printMenuSroting();

switch (sorting)

{

case 1:printf(" --- Сортировка по Фамилии --- \n");

sortByField(People, number, sorting);

break;

case 2:printf(" --- Сортировка по Имени --- \n");

sortByField(People, number, sorting);

break;

case 3:printf(" --- Сортировка по Полу --- \n");

sortByField(People, number, sorting);

break;

case 4:printf(" --- Сортировка по Росту --- \n");

sortByField(People, number, sorting);

break;

case 5:printf(" --- Сортировка по Весу --- \n");

sortByField(People, number, sorting);

break;

case 6:printf(" --- Сортировка по Числу рождения --- \n");

sortByField(People, number, sorting);

break;

case 7:printf(" --- Сортировка по Месяцу рождения --- \n");

sortByField(People, number, sorting);

break;

case 8:printf(" --- Сортировка по Году рождения --- \n");

sortByField(People, number, sorting);

break;

case 9:printf(" --- Сортировка по Номеру телефона --- \n");

sortByField(People, number, sorting);

break;

case 10:printf(" --- Сортировка по Индексу --- \n");

sortByField(People, number, sorting);

break;

case 11:printf(" --- Сортировка по Странам --- \n");

sortByField(People, number, sorting);

break;

case 12:printf(" --- Сортировка по Регионам --- \n");

sortByField(People, number, sorting);

break;

case 13:printf(" --- Сортировка по Городам --- \n");

sortByField(People, number, sorting);

break;

case 14:printf(" --- Сортировка по Улицам --- \n");

sortByField(People, number, sorting);

break;

case 15:printf(" --- Сортировка по Номерам домов --- \n");

sortByField(People, number, sorting);

break;

case 16:printf(" --- Сортировка по Номерам квартир --- \n");

sortByField(People, number, sorting);

break;

default: printf("Такого варианта выбора нет.\n"); system("pause");

}

}

else

{

printf("--- Заполненных структур нет ---\n");

system("pause");

}

system("cls");

}

# Тестирование программы

Данные, которые записываются в файл, находятся на рисунке 1

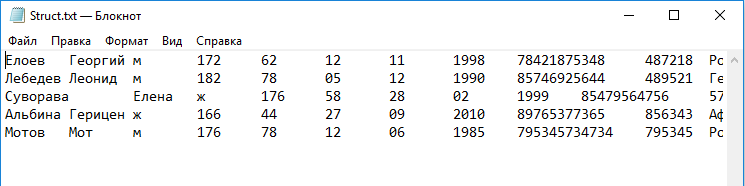


рис.1 входные данные, содержащиеся в файле

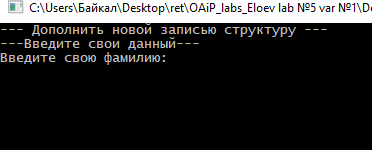


рис.2 функция добавления файла

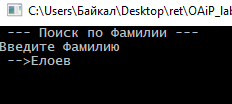


рис.3 функция поиска по структуре

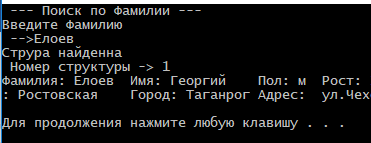


рис.4 функция поиска по структуре

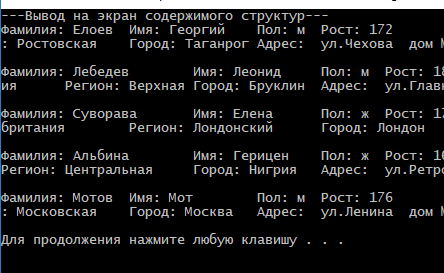


рис.5 функция вывода всей структуры

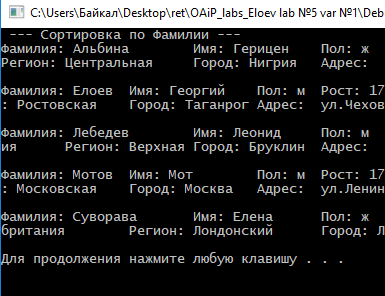


рис.6 функция сортировки

Все отсортированные структуры вводятся в новом порядке в другой файл(рис.7)

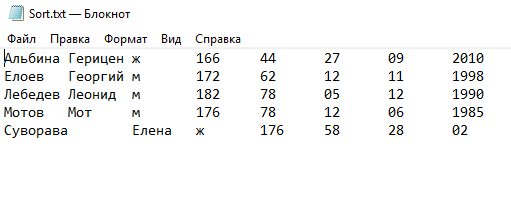


рис.7 отсортированные структуры в файле

# Вывод

Выполняя данную лабораторной работы, мы научились работать со сложными типами данных, каковыми являются структуры данных.