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 №8**

Выполнил:

студент гр. КТбо1-8

Ованесян Д.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: 10. «Государство»: название страны, столица, язык, население, площадь, территории, денежная единица, государственный строй.

# Алгоритм выполнения работы

Когда программа запускается, сразу выводит на экран меню выбора: 1) Внести новый фильм в файл, 2) Поиск нового фильма в файле 3) Вывод всех фильмов на экран 4) Сортировка фильмов в файле 5) Завершение работы программы.

Рассмотрим работу каждого пункта программы на основе двух введенных фильмов.

1) Выбор цифры 1 перебрасывает нас в функцию addStruct, где пользователь циклично вводит структуру, которая затем записывается в файл.

2) Выбор цифры 3 перебрасывает нас в функцию outputFile. Функция циклично выводит по 1 структуре из файла.

3) Выбор цифры 4 перебрасывает нас в функцию sortirovkaGosudarstva. Сначала функция записывает все структуры из файла в один массив структур. Затем, применяя пузырьковую сортировку, выводи отсортированный массив структур на экран(сортирует по возрастнию).

4) Выбор цифры 5 перебрасывает нас обратно в функцию main, где программа просто завершается.

# Программный код

/\*

Ованесян Даниил

Лабораторная работа №5

Вариант 10

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: «Государство»: название страны, столица, язык, население, площадь

территории, денежная единица, государственный строй.

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "locale"

#include <stdio.h>

#define LENGTH\_WORD\_FOR\_STRUCT 1000

#define COUNT\_STRUCTS 100

using namespace std;

struct country

{

char nameCountry[LENGTH\_WORD\_FOR\_STRUCT];

char maincity[LENGTH\_WORD\_FOR\_STRUCT];

char language[LENGTH\_WORD\_FOR\_STRUCT];

char peoples[LENGTH\_WORD\_FOR\_STRUCT];

char ploshad[LENGTH\_WORD\_FOR\_STRUCT];

char money[LENGTH\_WORD\_FOR\_STRUCT];

char gosstroy[LENGTH\_WORD\_FOR\_STRUCT];

};

void sorting\_f(int number, int field, struct country \*gos);

void sort(int number\_of\_structers, struct country \*gos);

void addStruct(int k, struct country \*gos);

void outputFile(int p, struct country \*gos);

int gettingStructs(struct country \*gos);

//void sortirovkaGosudarstva(country \*gos);

//void swap(struct country &gos1, struct country &gos2);

void new\_INPUT(int number\_of\_structures, struct country \*gos);

int main()

{

struct country gos[COUNT\_STRUCTS];

int main\_screen = 0;

int number\_of\_structures = gettingStructs(gos);

int sorting = 0;

int search = 0;

int not\_yes\_structs = 0;

int i = 0;

do

{

printf("(vvedite nomer)\n");

printf("1. Dopolnit' massiv struktur novymi strukturami.\n");

printf("2. Poisk struktury po zadannym znachenijam vybrannogo jelementa.\n");

printf("3. Vyvod na jekran soderzhimogo massiva struktur.\n");

printf("4. Uporjadochenie massiva struktur po gosudarstvu.\n");

printf("5. Vyhod iz programmy.\n");

printf("--> ");

scanf\_s("%d", &main\_screen);

system("cls");

switch (main\_screen)

{

case 1:

printf("--- Dopolnit' massiv struktur novymi strukturami ---\n");

if (number\_of\_structures < COUNT\_STRUCTS)

{

addStruct(number\_of\_structures, gos);

number\_of\_structures++;

system("pause");

system("cls");

}

else

{

printf("--- Vvedenno maksimal'no vozmozhnoe kolichestvo struktur ---");

}

break;

case 2:

if (number\_of\_structures != 0)

{

printf("(vvedite nomer)\n");

printf("--- Poisk struktury po zadannym znachenijam vybrannogo jelementa --- \n");

printf("1. Poisk po gosvu.\n");

printf("2. Poisk po maincity.\n");

printf("3. Poisk po language.\n");

printf("4. Poisk po ludiam.\n");

printf("5. Poisk po ploshadi.\n");

printf("6. Poisk po dengam.\n");

printf("7. Poisk po gossstroyu.\n");

printf("--> ");

scanf("%d", &search);

system("cls");

switch (search)

{

case 1: printf(" --- Poisk po nazvaniyu gosudarstva --- \n"); printf("Vvedite nazvanie gosva\n -->"); char search\_gosvo[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_gosvo);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_gosvo, gos[i].nameCountry))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 2: printf(" --- Poisk po glavnomu gorodu --- \n"); printf("Vvedite glavniy gorod\n -->"); char search\_maincity[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_maincity);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_maincity, gos[i].maincity))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 3: printf(" --- Poisk po yaziky --- \n"); printf("Vvedite yazik\n -->"); char search\_language[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_language);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_language, gos[i].language))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 4: printf(" --- Poisk ludiam --- \n"); printf("Vvedite kol-vo ludei\n -->"); char search\_peoples[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_peoples);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_peoples, gos[i].peoples))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 5: printf(" --- Poisk po ploshadi --- \n"); printf("Vvediteploshad\n -->"); char search\_ploshad[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_ploshad);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_ploshad, gos[i].ploshad))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 6: printf(" --- Poisk po dengam --- \n"); printf("Vvedite kol-vo deneg(rubley)\n -->"); char search\_money[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_money);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_money, gos[i].money))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

case 7: printf(" --- Poisk po gosstyoryu --- \n"); printf("Vvedite gosstroy\n -->"); char search\_gosstroy[LENGTH\_WORD\_FOR\_STRUCT]; scanf("%s", &search\_gosstroy);

for (i = 0; i<number\_of\_structures; i++)

{

if (!strcmp(search\_gosstroy, gos[i].gosstroy))

{

not\_yes\_structs++;

printf("Struktura najdenna\n Nomer struktury -> %d\n", i + 1);

outputFile(i, gos);

}

}

break;

default: printf("Takogo varianta vybora net.");

}

if (not\_yes\_structs == 0)

{

printf("Struktura ne najdenna\n");

}

not\_yes\_structs = 0;

system("pause");

system("cls");

}

else

{

printf("--- Zapolnennyh struktur net ---\n");

}

system("pause");

system("cls");

break;

case 3:

printf("--- Vyvod na jekran soderzhimogo massiva struktur ---\n");

if (number\_of\_structures == 0)

{

printf("Zapolnennyh struktur net\n");

}

else

{

for (i = 0; i < number\_of\_structures; i++)

{

outputFile(i, gos);

}

}

system("pause");

system("cls");

break;

case 4:

{

sort(number\_of\_structures, gos);

break;

}

case 5:

exit(0);

break;

default: printf("Takogo varianta vybora net.");

}

} while (true);

system("pause");

return 0;

}

void addStruct(int k, struct country \*gos)

{

printf("Vvod struktur\n");

printf("Vvedite nazvanie gosudarstva --> ");

scanf("%s", &gos[k].nameCountry);

printf("Vvedite nazvanie glavnogo goroda --> ");

scanf("%s", &gos[k].maincity);

printf("Vvedite yazik --> ");

scanf("%s", &gos[k].language);

printf("Vvedite kol-vo ludei --> ");

scanf("%s", &gos[k].peoples);

printf("Vvedite ploshad gos-va --> ");

scanf("%s", &gos[k].ploshad);

printf("Vvedite kol-vo deneg --> ");

scanf("%s", &gos[k].money);

printf("Vvedite gosstroy --> ");

scanf("%s", &gos[k].gosstroy);

printf("Struktura vvedenna.\n\n");

FILE \*text\_file\_output = fopen("output.txt", "a");

fprintf(text\_file\_output, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n", &gos[k].nameCountry, &gos[k].maincity, &gos[k].language, &gos[k].peoples, &gos[k].ploshad, &gos[k].money, &gos[k].gosstroy);

fclose(text\_file\_output);

}

int sorting\_comparison(int field, int i, struct country \*gos)

{

int code;

switch (field) {

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

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

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

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

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

case 6:code = strcmp(gos[i + 1].money, gos[i].money); return code;

case 7:code = strcmp(gos[i + 1].maincity, gos[i].maincity); return code;

}

}

void sort(int number\_of\_structures, struct country \*gos)

{

int sorting\_s = 0;

printf(" --- Uporjadochenie massiva struktur po zadannomu polju --- \n");

printf("(vvedite nomer)\n");

printf("--- Sortirovka po zadannomu polju --- \n");

printf("1. Sortirovka po gosadarstvu.\n");

printf("2. Sortirovka po stolice.\n");

printf("3. Sortirovka po yaziky.\n");

printf("4. Sortirovka po kol- vu ludiam.\n");

printf("5. Sortirovka po ploshadi.\n");

printf("5. Sortirovka po dengam.\n");

printf("6. Sortirovka po gosstroyu.\n");

printf("--> ");

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

system("cls");

if (number\_of\_structures != 0)

{

switch (sorting\_s)

{

case 1: printf(" --- Sortirovka po gosudarstvu --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 2: printf(" --- Sortirovka po stolice --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 3: printf(" --- Sortirovka po yaziky --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 4: printf(" --- Sortirovka po kol-vu ludei --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 5: printf(" --- Sortirovka po ploshadi --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 6: printf(" --- Sortirovka po money --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

case 7: printf(" --- Sortirovka po gosstroyu --- \n"); sorting\_f(number\_of\_structures, sorting\_s, gos);

break;

default: printf("Takogo varianta vybora net.");

}

printf("Structura otsortirovanna");

system("pause");

}

else

{

printf("--- Zapolnennyh struktur net ---\n");

system("pause");

}

system("cls");

}

void sorting\_s(int number, int field, struct country \*gos)

{

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

{

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

{

if ((sorting\_comparison(field, j, gos)) < 0)

{

country Buff = gos[j];

gos[j] = gos[j + 1];

gos[j + 1] = Buff;

}

}

}

outputFile(number, gos);

new\_INPUT(number, gos);

system("pause");

system("cls");

}

void outputFile(int p, struct country \*gos)

{

printf("Gosudarstvo --> %s\t", gos[p].nameCountry);

printf("Glavni gorod --> %s\t", gos[p].maincity);

printf("Yazik --> %s\t", gos[p].language);

printf("Ludi --> %s\t", gos[p].peoples);

printf("Ploshad --> %s\t", gos[p].ploshad);

printf("Dengi --> %s\t", gos[p].money);

printf("Gosstroy --> %s\t", gos[p].gosstroy);

}

void new\_INPUT(int number\_of\_structures, struct country \*gos)

{

FILE \*text\_file\_output = fopen("output.txt", "w");

int i = 0;

for (i = 0; i < number\_of\_structures; i++)

{

fprintf(text\_file\_output, "%s\t%s\t%s\t%s\t%s\t%s\t%s\n", &gos[i].nameCountry, &gos[i].maincity, &gos[i].language, &gos[i].peoples, &gos[i].ploshad, &gos[i].money, &gos[i].gosstroy);

}

fclose(text\_file\_output);

}

int gettingStructs(struct country \*gos)

{

FILE \*text\_file\_output = fopen("output.txt", "r");

FILE \*text\_file\_output\_test = fopen("output.txt", "r");

int k = 0;

if (text\_file\_output != NULL)

{

while (fscanf(text\_file\_output\_test, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s", &gos[k].nameCountry, &gos[k].maincity, &gos[k].language, &gos[k].peoples, &gos[k].ploshad, &gos[k].money, &gos[k].gosstroy) != EOF)

{

fscanf(text\_file\_output, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s", &gos[k].nameCountry, &gos[k].maincity, &gos[k].language, &gos[k].peoples, &gos[k].ploshad, &gos[k].money, &gos[k].gosstroy);

k++;

}

}

else

{

printf("--- Fajla net ili on ne otkrylsja ---");

system("pause");

exit(0);

}

fclose(text\_file\_output);

fclose(text\_file\_output\_test);

return k;

}

# Тестирование программы

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

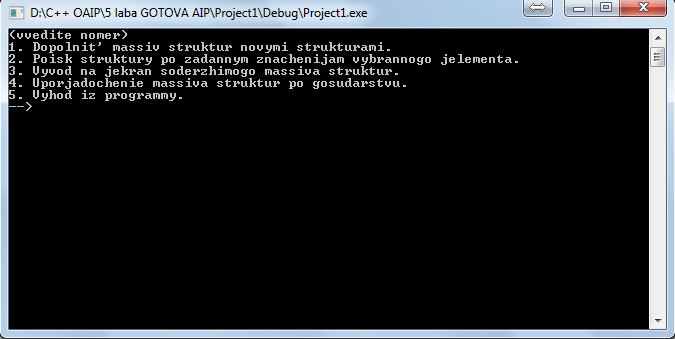


рис.1 Главное меню

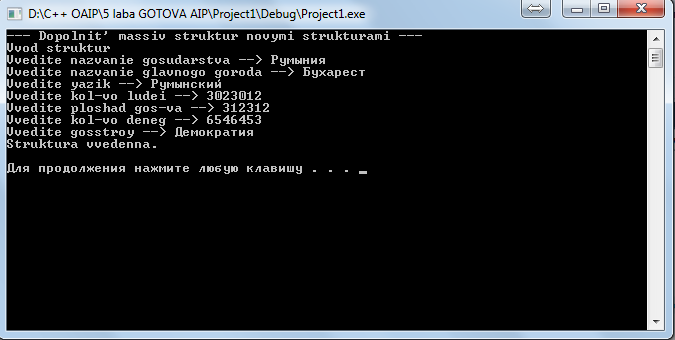


рис.2 Ввод нового фильма в файл

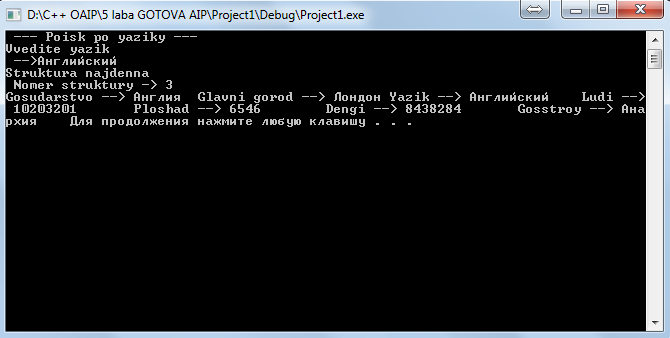
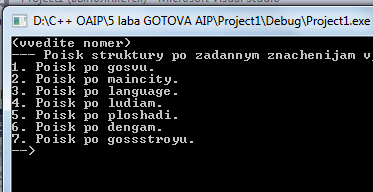


рис.3 Поиск по полю

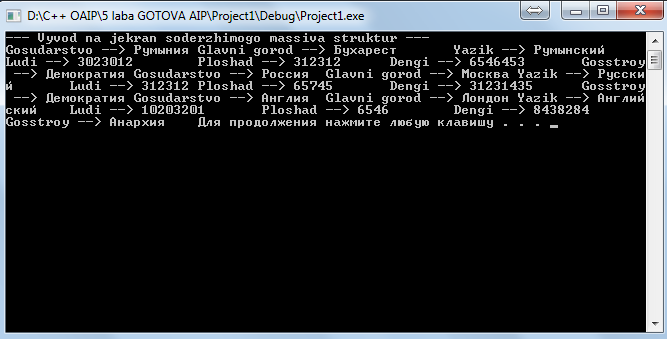


рис.4 Вывод всех структур

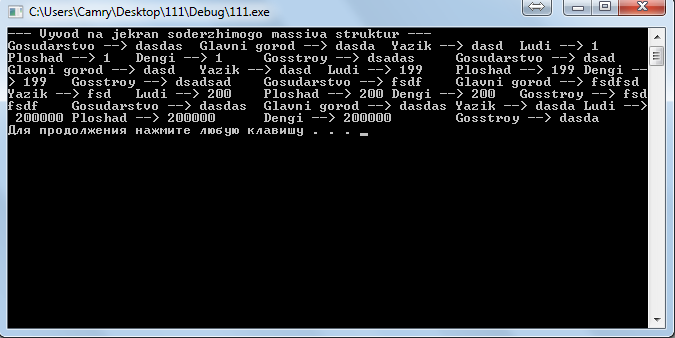


Рис.5 Отсортированная структура

# Вывод

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