Выполнила

студентка группы КТбо1-2 В. Е. Денисова

Принял

ассистент ИКТИБ Д. С. Кочубей

Таганрог 2023

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

(ФГАОУ ВО «ЮФУ»)

Институт компьютерных технологий и информационной безопасности

Кафедра системного анализа и телекоммуникаций

**ЛАБОРАТОРНАЯ РАБОТА № 5**

по дисциплине «Основы алгоритмизации и программирования»

Вариант 10

на тему:

«Работа со структурами»

**Содержание**

Содержание 2

Тех задание 3

Цель 3

Задача 3

Ход работы 4

Алгоритмы 4

Блок-схема 6

Программы 7

Примеры работы 8

Вывод 12

Листинг 13

**Техническое задание**

**Цель работы**

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

**Задача**

Вариант 10.

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.

Содержание структуры:

“Государство”: название страны, столица, язык, население, площадь территории, денежная единица, государственный строй, процент женщин, процент мужчин.

**Ход работы**

**Алгоритмы**

В программе применяется система меню с выбором реализации функций: запись новых пользователей, поиск пользователя по заданному параметру и значению, сортировка по параметру, вывод списка покупателей в консоль, вывод списка покупателей в файл, очистка данных, чтение существующих данных. Далее описываются алгоритмы функций.

Запись новых пользователей – int record\_struct:

Переменные: FILE\* input – указатель на файл для записи данных; Int new\_count – счетчик ввода новых государств.

1. Ввод числа новых государств, для ограничения кол-ва вызова функции заполнения структуры.
2. Циклическое заполнение элементов массива структур.
3. Вывод элементов в файл исходных данных для сохранения и последующего обращения к данным.

Переменные:

1. Ввод пункта меню, выбор параметра для поиска.
2. Циклический перебор массива данных.
3. Для каждой итерации счетчик результата поиска обновляется, в соответствии с выбранным параметром, идет сравнение введенного значения поиска с существующим значением элемента массива.
4. Если есть совпадение, счетчик результата обновляется, выводится элемент, имеющий совпадение. Иначе – сообщение об отсутствии совпадений.

Сортировка по параметру void sorting:

1. Ввод пункта меню, выбор параметра для поиска.
2. Вывод отсортированного массива данных.

Вывод списка государств в консоль void print:

1. Циклический вывод каждого элемента массива данных.

Вывод списка государств в файл void output\_file:

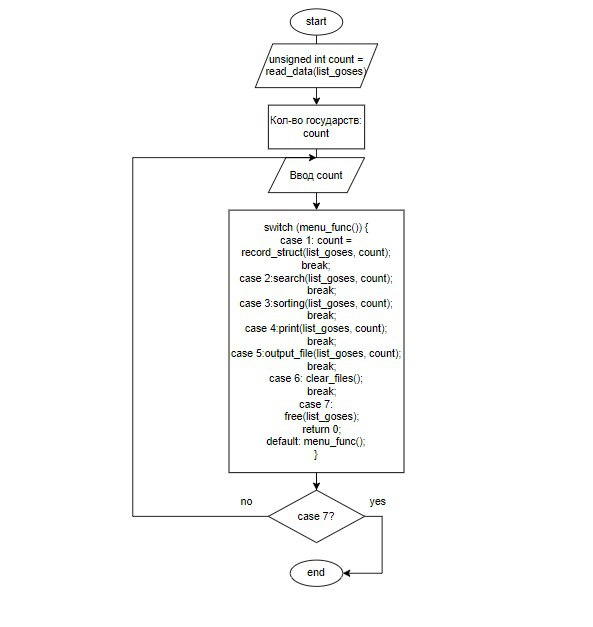
Параметр: FILE\* file – указатель на файл для вывода данных.

1. Циклический вывод каждого элемента массива данных.

Очистка данных clear\_files: Вызов открытия файлов с параметром перезаписи данных. Т.к. новые данные не записываются, файлы остаются пустыми.

Чтение существующих данных record\_struct: Пока не закончатся считываемые значение, заполнение элементов массива данных из считываемой строки.

**Блок-схема**



**Написание программы**

Для удобства были инициализированы глобальные переменные MAX\_INFO\_SIZE = 20 – максимальный размер строки данных в структуре и MAX\_LIST\_GOS\_SIZE = 20 - максимальный размер списка государств

Для выполнения задачи, необходимо хранить значения, в данном случае, используется структура struct gosudarstvo с параметрами: char country[MAX\_INFO\_SIZE] – Название страны, char capital[MAX\_INFO\_SIZE] – Столица, char language[MAX\_INFO\_SIZE] – Язык, char populations[MAX\_INFO\_SIZE] – Население, char area[MAX\_INFO\_SIZE] – Площадь, char valmoney[MAX\_INFO\_SIZE]– Денежная единица, char gosstoy[MAX\_INFO\_SIZE] – Гос. строй, char perw[MAX\_INFO\_SIZE] – Процент женщин, char perm[MAX\_INFO\_SIZE] - Процент мужчин. Заполненные структуры хранятся в массиве структур с динамическим выделением gosudarstvo\* list\_goses.

В начале выполнения программы, происходит считывание исходных данных с записью их количества в переменную, чтобы иметь представление о длине списка заполненных в массиве данных элементов. В последующем, переменная используется как длина массива.

Для реализации функционала меню выбора, используется команда switch, которая обрабатывает вводимое значение пункта меню, и в зависимости от него вызывает нужную функцию.

По завершению работы одной функции, есть необходимость заново вызывать главную функцию main(), чтобы иметь возможность вызвать нереализованную функцию.

Завершение программы происходит путем выбора соответствующего пункта меню, при этом происходит освобождение занимаемой массивом данных памяти.

**Примеры работы программы**

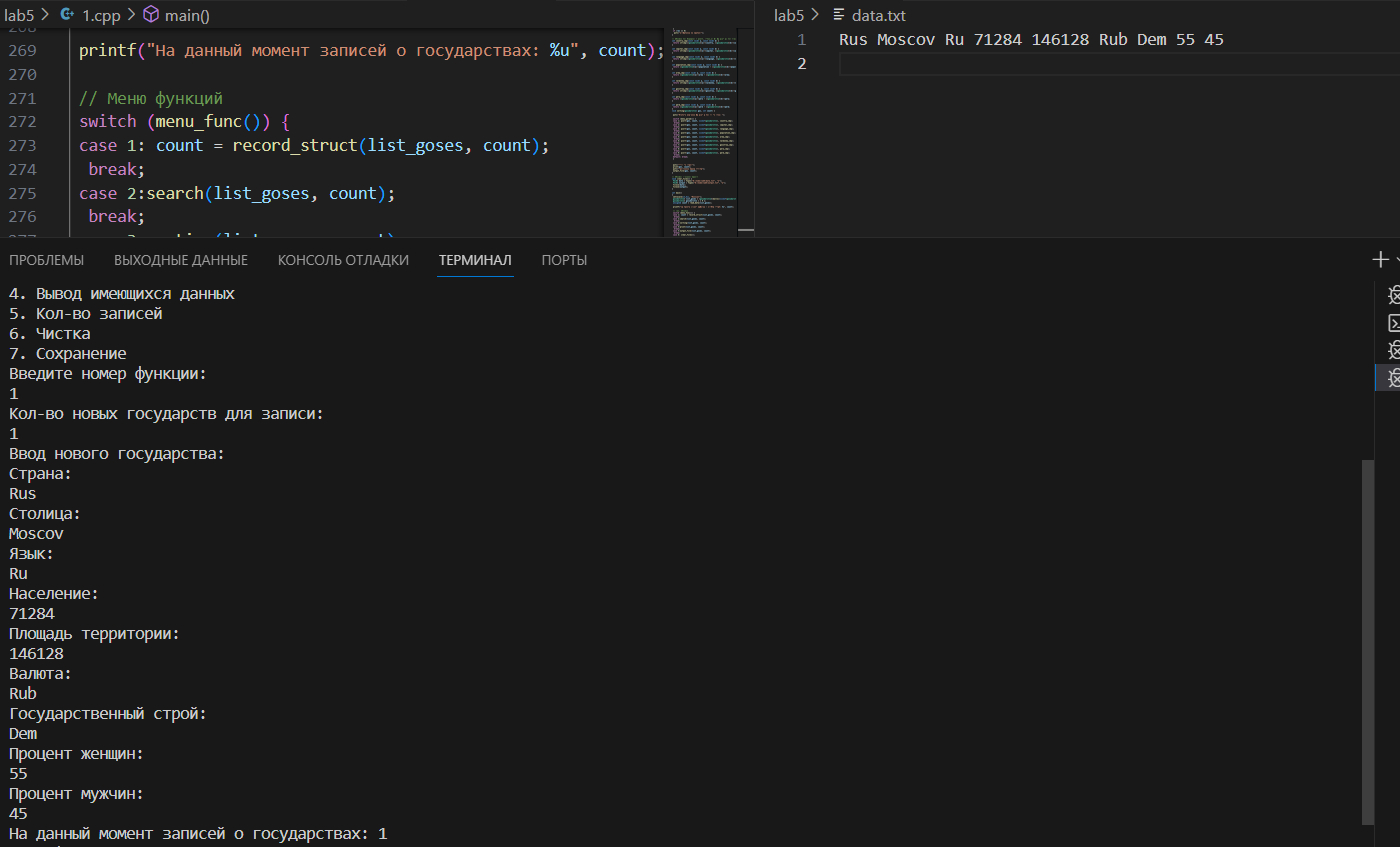


Рисунок 1 – Выполнение функции записи покупателя.

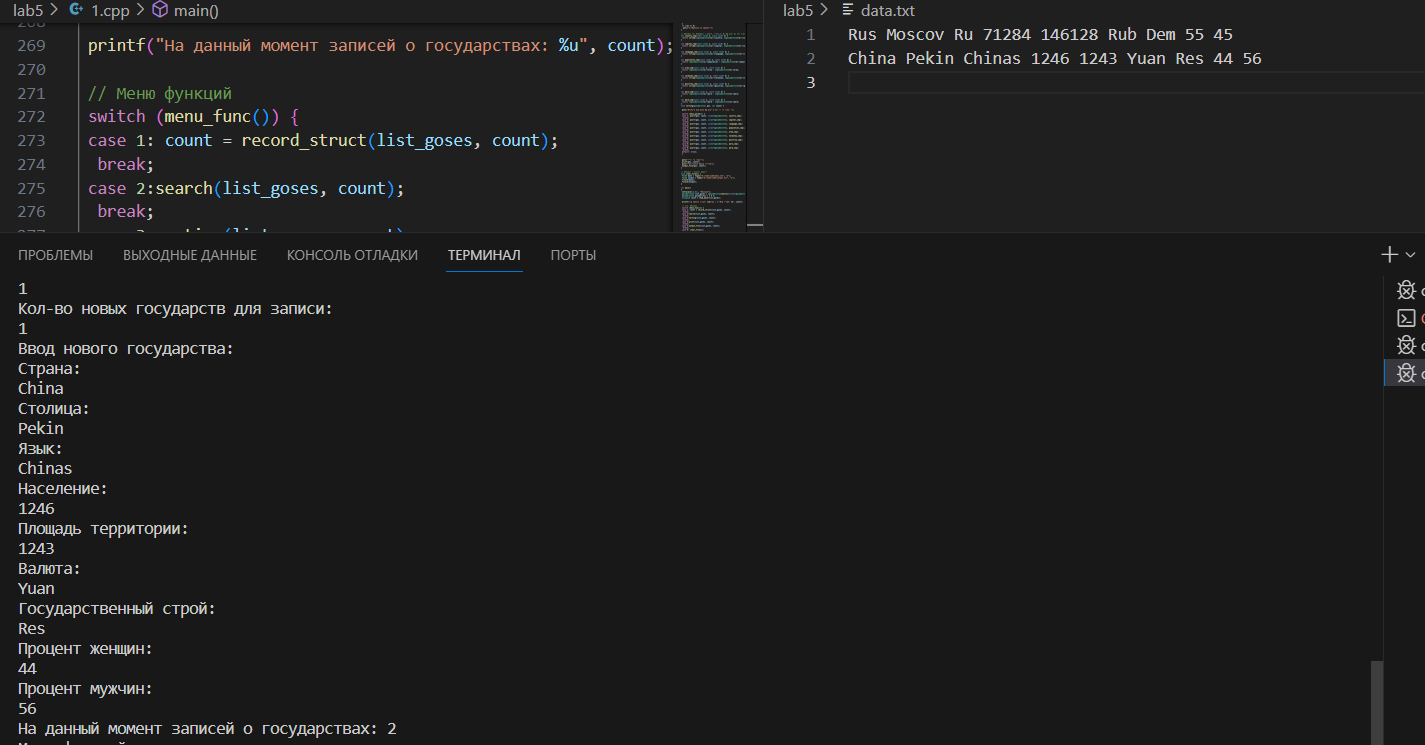
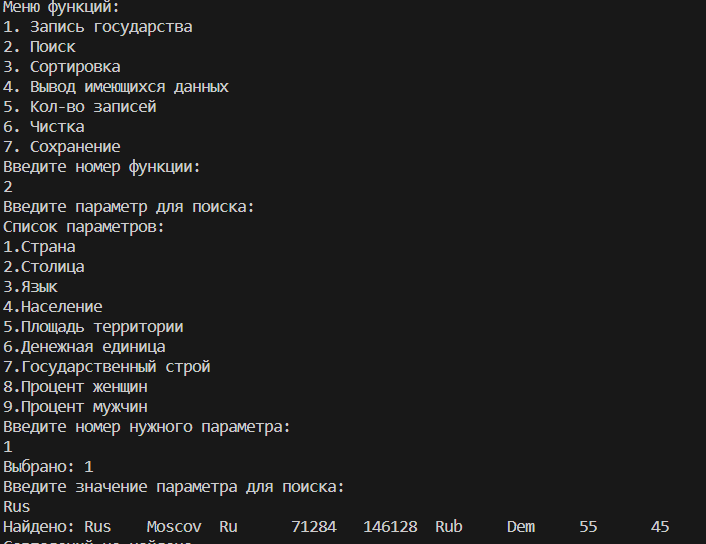


Рисунок 2 – Выполнение функции записи покупателя при уже существующих данных (добавление новых записей).



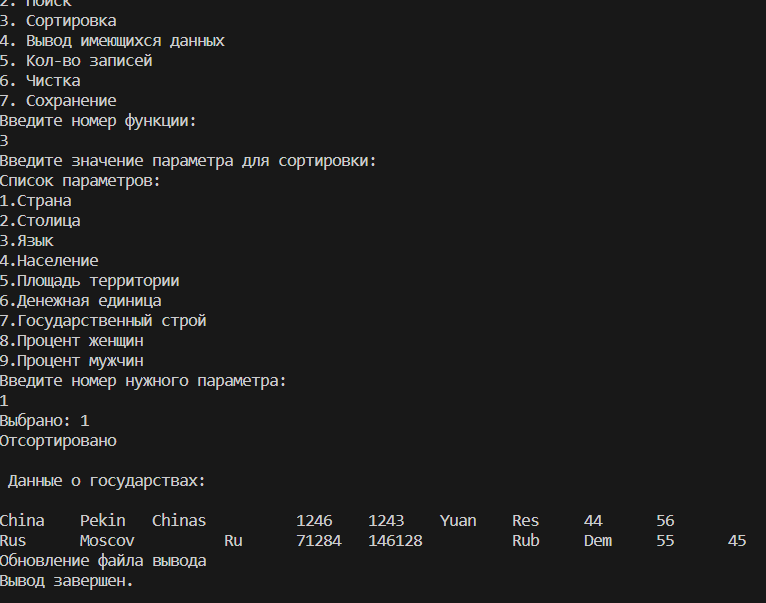
Рисунок 3 – Выполнение функции поиска по параметру и его значению.

Рисунок 4 – Выполнение функции сортировки по параметру.

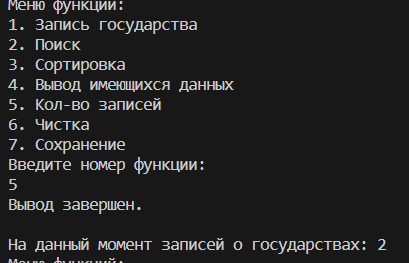


Рисунок 5 – Выполнение вывода данных в консоль.

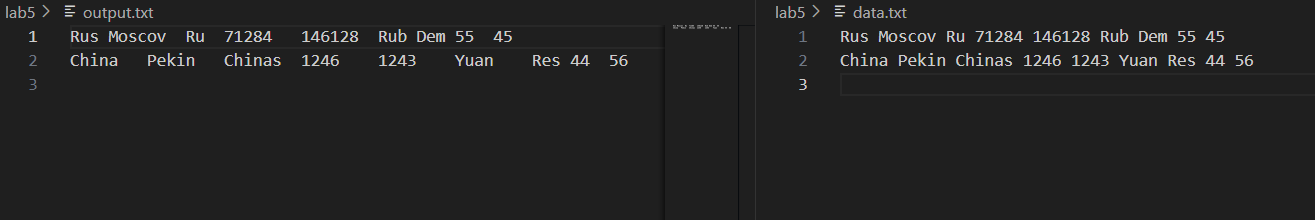


Рисунок 6 – Выполнение Вывода данных в файл.

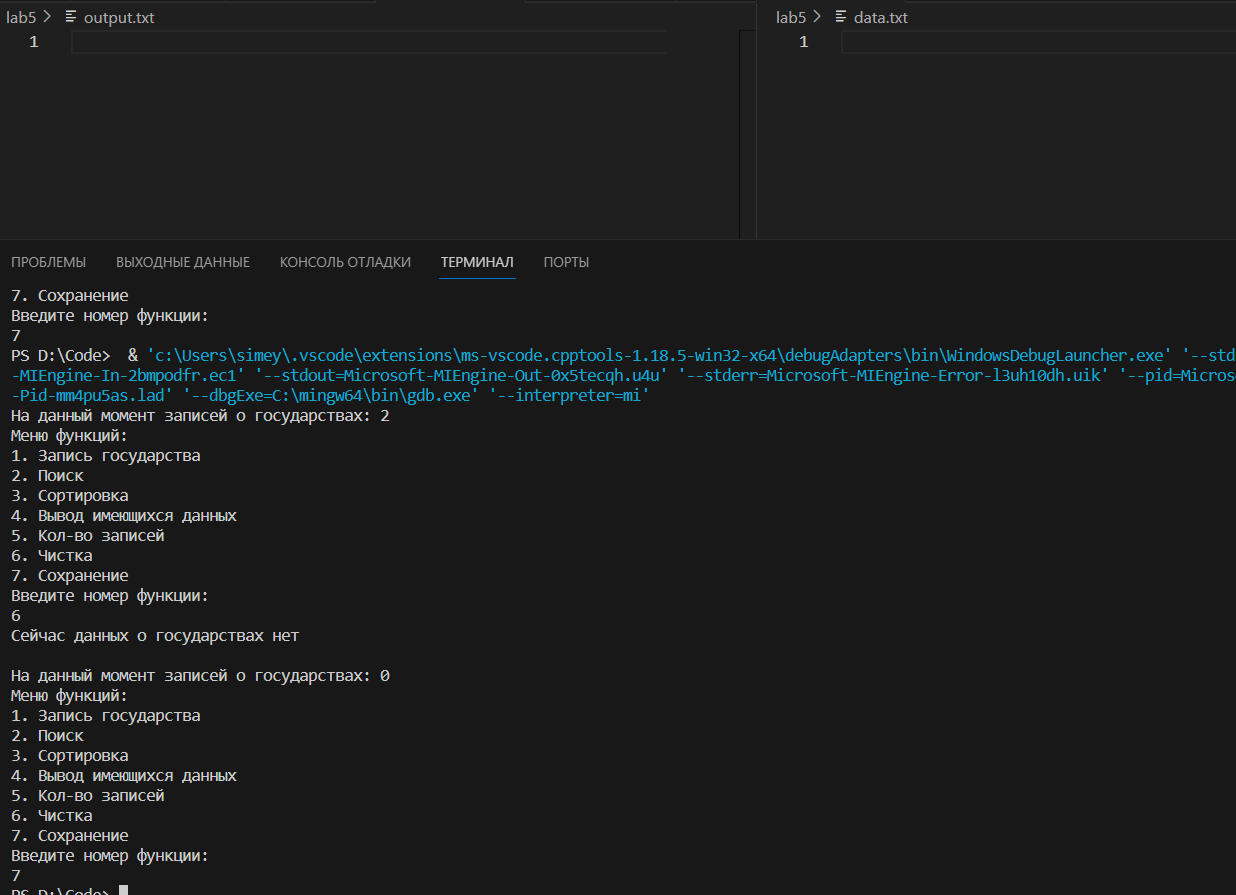


Рисунок 7 – Очищение данных и завершение программы.

**Вывод**

# Изучена работа со сложными типами данных, каковыми являются структур данных.

# **Листинг (Приложение а)**

#pragma warning(disable : 4996)

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_INFO\_SIZE 20 // Макс размер строки данных в структуре

#define MAX\_LIST\_GOS\_SIZE 20 // Макс размер списка

struct gosudarstvo {

 char country[MAX\_INFO\_SIZE];

 char capital[MAX\_INFO\_SIZE];

 char language[MAX\_INFO\_SIZE];

 char population[MAX\_INFO\_SIZE];

 char area[MAX\_INFO\_SIZE];

 char valmoney[MAX\_INFO\_SIZE];

 char gosstroy[MAX\_INFO\_SIZE];

 char perw[MAX\_INFO\_SIZE];

 char perm[MAX\_INFO\_SIZE];

};

int menu\_func() {

 int choise;

 puts("\nМеню функций: ");

 puts("1. Запись государства \n2. Поиск \n3. Сортировка \n4. Вывод имеющихся данных \n5. Кол-во записей \n6. Чистка \n7. Сохранение");

 puts("Введите номер функции: ");

 scanf("%i", &choise);

 return choise;

}

int record\_struct(gosudarstvo\* gosudarstvo, unsigned id) {

 FILE\* input = fopen("D:/Code/lab5/data.txt", "a");

 int new\_count;

 puts("Кол-во новых государств для записи: ");

 scanf("%i", &new\_count);

 if (input == NULL) perror("Ошибка открытия файла data.txt для записи и хранения данных о государствах");

 else {

  for (int i = 0; i < new\_count; i++) {

   puts("Ввод нового государства: ");

   puts("Страна: ");

   scanf("%s", &gosudarstvo[id + i].country);

   puts("Столица: ");

   scanf("%s", &gosudarstvo[id + i].capital);

   puts("Язык: ");

   scanf("%s", &gosudarstvo[id + i].language);

   puts("Население: ");

   scanf("%s", &gosudarstvo[id + i].population);

   puts("Площадь территории: ");

   scanf("%s", &gosudarstvo[id + i].area);

   puts("Валюта: ");

   scanf("%s", &gosudarstvo[id + i].valmoney);

   puts("Государственный строй: ");

   scanf("%s", &gosudarstvo[id + i].gosstroy);

   puts("Процент женщин: ");

   scanf("%s", &gosudarstvo[id + i].perw);

   puts("Процент мужчин: ");

   scanf("%s", &gosudarstvo[id + i].perm);

   fprintf(input, "%s %s %s %s %s %s %s %s %s \n",

    gosudarstvo[id + i].country, gosudarstvo[id + i].capital, gosudarstvo[id + i].language, gosudarstvo[id + i].population, gosudarstvo[id + i].area,

    gosudarstvo[id + i].valmoney, gosudarstvo[id + i].gosstroy, gosudarstvo[id + i].perw, gosudarstvo[id + i].perm);

  }

 }

 fclose(input);

 return id;

}

int read\_data(gosudarstvo\* gos) {

 FILE\* output = fopen("D:/Code/lab5/data.txt", "r");

 unsigned count = 0;

 if (output == NULL) perror("Ошибка открытия файла data.txt для записи информации о государствах в структуры");

 else {

  char stroke[256];

  int i = 0;

  while (fgets(stroke, sizeof(stroke), output) != NULL) {

   sscanf(stroke, "%s %s %s %s %s %s %s %s %s \n", &gos[i].country, &gos[i].capital, &gos[i].language, &gos[i].population, &gos[i].area, &gos[i].valmoney, &gos[i].gosstroy, &gos[i].perw, &gos[i].perm);

   count++;

   i++;

  }

 }

 if (count == 0) puts("Сейчас данных о государствах нет\n");

 fclose(output);

 return count;

}

// Функция вывода структур в консоль. Работает, проверено на практике - есть вывод

void print(gosudarstvo\* gos, int count) {

 if (gos == NULL) printf("Нет государств");

 else {

  puts("\n Данные о государствах: \n");

  for (int i = 0; i < count; i++) {

   printf("%s \t %s \t %s \t %s \t %s \t %s \t %s \t %s \t %s \n", &gos[i].country, &gos[i].capital, &gos[i].language, &gos[i].population, &gos[i].area, &gos[i].valmoney, &gos[i].gosstroy, &gos[i].perw, &gos[i].perm);

  }

 }

}

// Функция вывода структур в файл. Работает, проверено путем записи в файл

void output\_file(gosudarstvo\* gosudarstvo, unsigned count) {

 FILE\* file = fopen("D:/Code/lab5/output.txt", "w");

 if (file == NULL) perror("Ошибка открытия файла");

 else {

  if (gosudarstvo == NULL) fprintf(file, "Нет покупателей");

  else {

   for (int i = 0; i < count; i++) {

    fprintf(file, "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s \n", gosudarstvo[i].country, gosudarstvo[i].capital, gosudarstvo[i].language, gosudarstvo[i].population, gosudarstvo[i].area,gosudarstvo[i].valmoney, gosudarstvo[i].gosstroy, gosudarstvo[i].perw, gosudarstvo[i].perm);

   }

   puts("Вывод завершен.\n");

}

 }

 fclose(file);

}

unsigned int menu\_param() {

 unsigned int choise;

 puts("Список параметров: ");

 puts("1.Страна \n2.Столица \n3.Язык \n4.Население \n5.Площадь территории \n6.Денежная единица \n7.Государственный строй \n8.Процент женщин \n9.Процент мужчин");

 puts("Введите номер нужного параметра: ");

 do {

  scanf("%u", &choise);

  printf("Выбрано: %u\n", choise);

 } while (choise < 1 || choise > 9);

 return choise;

}

void search(gosudarstvo\* gosudarstvo, int count) {

 char s\_goal[MAX\_INFO\_SIZE];

 unsigned int int\_goal = 0, param, res;

 puts("Введите параметр для поиска: ");

 param = menu\_param();

 puts("Введите значение параметра для поиска: ");

 if (param >= 6)

  scanf("%s", &int\_goal);

 else scanf("%s", &s\_goal);

 for (int i = 0; i <= count; i++) {

  res = 0;

  switch (param) {

  case 1: if (strcmp(gosudarstvo[i].country, s\_goal) == 0) res = 1;

   break;

  case 2: if (strcmp(gosudarstvo[i].capital, s\_goal) == 0) res = 1;

   break;

  case 3:  if (strcmp(gosudarstvo[i].language, s\_goal) == 0) res = 1;

   break;

  case 4: if (strcmp(gosudarstvo[i].population,s\_goal) == 0) res = 1;

   break;

  case 5: if (strcmp(gosudarstvo[i].area, s\_goal) ==0) res = 1;

   break;

  case 6: if (strcmp(gosudarstvo[i].valmoney, s\_goal) == 0) res = 1;

   break;

  case 7: if (strcmp(gosudarstvo[i].gosstroy, s\_goal) == 0) res = 1;

   break;

  case 8: if (strcmp(gosudarstvo[i].perw, s\_goal) == 0) res = 1;

   break;

  case 9:if (strcmp(gosudarstvo[i].perm, s\_goal) == 0) res = 1;

   break;

  default: break;

  }

  if (res == 1) printf("Найдено: %s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s \n",

   gosudarstvo[i].country, gosudarstvo[i].capital, gosudarstvo[i].language, gosudarstvo[i].population, gosudarstvo[i].area,

   gosudarstvo[i].valmoney, gosudarstvo[i].gosstroy, gosudarstvo[i].perw, gosudarstvo[i].perm);

 }

 if (res == 0)

  puts("Совпадений не найдено");

}

// Функции для указания в qsort, отдельно по параметрам для возможности выбора

int country\_cmp(const void\* a, const void\* b) {

 return strcmp(((gosudarstvo\*)a)->country, ((gosudarstvo\*)b)->country);

}

int capital\_cmp(const void\* a, const void\* b) {

 return strcmp(((gosudarstvo\*)a)->capital, ((gosudarstvo\*)b)->capital);

}

int language\_cmp(const void\* a, const void\* b) {

 return strcmp(((gosudarstvo\*)a)->language, ((gosudarstvo\*)b)->language);

}

int population\_cmp(const void\* a, const void\* b) {

 return ((gosudarstvo\*)a)->population - ((gosudarstvo\*)b)->population;

}

int area\_cmp(const void\* a, const void\* b) {

 return ((gosudarstvo\*)a)->area - ((gosudarstvo\*)b)->area;

}

int valmoney\_cmp(const void\* a, const void\* b) {

 return strcmp(((gosudarstvo\*)a)->valmoney, ((gosudarstvo\*)b)->valmoney);

}

int gosstroy\_cmp(const void\* a, const void\* b) {

 return strcmp(((gosudarstvo\*)a)->gosstroy, ((gosudarstvo\*)b)->gosstroy);

}

int perw\_cmp(const void\* a, const void\* b) {

 return ((gosudarstvo\*)a)->perw - ((gosudarstvo\*)b)->perw;

}

int perm\_cmp(const void\* a, const void\* b) {

 return ((gosudarstvo\*)a)->perm - ((gosudarstvo\*)b)->perm;

}

void sorting(gosudarstvo\* gos, int count) {

 puts("Введите значение параметра для сортировки: ");

 switch (menu\_param()) {

 case 1: qsort(gos, count, sizeof(gosudarstvo), country\_cmp);

  break;

 case 2: qsort(gos, count, sizeof(gosudarstvo), capital\_cmp);

  break;

 case 3: qsort(gos, count, sizeof(gosudarstvo), language\_cmp);

  break;

 case 4: qsort(gos, count, sizeof(gosudarstvo), population\_cmp);

  break;

 case 5: qsort(gos, count, sizeof(gosudarstvo), area\_cmp);

  break;

 case 6: qsort(gos, count, sizeof(gosudarstvo), valmoney\_cmp);

  break;

 case 7: qsort(gos, count, sizeof(gosudarstvo), gosstroy\_cmp);

  break;

 case 8: qsort(gos, count, sizeof(gosudarstvo), perw\_cmp);

  break;

 case 9: qsort(gos, count, sizeof(gosudarstvo), perm\_cmp);

  break;

 default: break;

 }

 puts("Отсортировано");

 print(gos, count);

 puts("Обновление файла вывода");

 output\_file(gos, count);

}

// Функция очищения файлов

void clear\_files() {

 FILE\* data = fopen("D:/Code/lab5/data.txt", "w");

 FILE\* output = fopen("D:/Code/lab5/output.txt", "w");

 fclose(data);

 fclose(output);

}

int main()

{

 setlocale(LC\_ALL, "Russian");

 gosudarstvo\* list\_goses = (gosudarstvo\*)malloc(sizeof(gosudarstvo) \* MAX\_LIST\_GOS\_SIZE);

 gosudarstvo\* gosudarstvo = { 0 };

 unsigned count = read\_data(list\_goses);

 printf("На данный момент записей о государствах: %u", count);

 // Меню функций

 switch (menu\_func()) {

 case 1: count = record\_struct(list\_goses, count);

  break;

 case 2:search(list\_goses, count);

  break;

 case 3:sorting(list\_goses, count);

  break;

 case 4:print(list\_goses, count);

  break;

 case 5:output\_file(list\_goses, count);

  break;

 case 6: clear\_files();

  break;

 case 7:

  free(list\_goses);

  return 0;

 default: menu\_func();

 }

 main();

}