Министерство образования Республики Беларусь

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра микро- и наноэлектроники

**Лабораторная работа №6**

**«Алгоритм поиска и сортировки в массивах»**

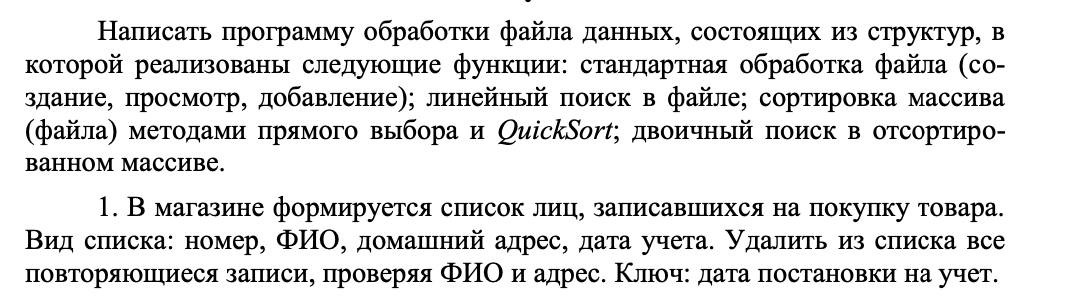
**Вариант 1**

|  |  |
| --- | --- |
| Выполнил: | студент группы 142701  Ахремчик Никита Андреевич |
| Проверил: |  |

Минск 2022

**Задачи:** изучить способы сортировки и поиска в массивах структур и файлах.

**Индивидуальное задание**



Текст программы:

#**include** <iostream>

#**include** <stdio.h>

#**include** <stdlib.h>

**using** **namespace** std;

**const** **char** EN [] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

**const** **char** en [] = "abcdefghijklmnopqrstuvwxyz";

**const** **char** num [] = "1234567890"; // добавить символов

**struct** Date{

**int** day;

**int** month;

**int** year;

};

**struct** Customer{

**char** fa[32]; // фамилия

**char** im[32]; // имя

**char** ot[32]; // отчество

**char** address[32]; // адрес

Date date; // дата учета

};

// Функция сортировки прямым выбором

**void** **selectionSort**(Customer \*customer, **int** size){

//Customer newArr = new Customer[size];

**int** m;

Customer r;

**for** (**int** i = 0; i < size-1; i++){

m = i;

**for** (**int** j = i + 1; j < size; j++)

**if** (customer[j].date.year < customer[m].date.year) m = j;

r = customer[m];

customer[m] = customer[i];

customer[i] = r;

}

}

// Функция сортировки QuickSort

**void** **quickSort**(Customer \*&customer, **int** size){

**int** i=0, j=size-1, x;

Customer r;

**int** L=0; **int** R=size;

x = customer[(L+R)/2].date.year;

**while** (i <= j) {

**while** (customer[i].date.year<x) i++;

**while** (customer[j].date.year>x) j--;

**if** (i <= j){

r = customer[i];

customer[i] = customer[j];

customer[j] = r;

i++;

j--;

}

}

}

// бинарный поиск в отсортированном массиве

**int** **binSearch**(**const** Customer \*customers, **int** amount, Date key){

**int** L=0; **int** R=amount-1;

**while**(L<=R){

**int** m = (L+R)/2;

**if** (customers[m].date.year==key.year){

**if** (customers[m].date.month==key.month){

**if** (customers[m].date.day==key.day) **return** m;

}

**if** (customers[m].date.month>key.month || customers[m].date.day>key.day) R=m-1;

**if** (customers[m].date.month<key.month || customers[m].date.day<key.day) L=m+1;

}

**if** (customers[m].date.year<key.year) L=m+1;

**if** (customers[m].date.year>key.year) R=m-1;

}

**return** -1;

}

// линейный поиск

**int** **linSearch**(Customer \*customer, **int** size, **int** year, **int** month, **int** day){

**for** (**int** i=0; i<size; i++){

**if** (customer[i].date.year == year){

**if**(customer[i].date.month == month){

**if**(customer[i].date.day == day){

**return** i;

}

}

}

}

**return** -1;

}

**void** **push\_back**(Customer \*&customers, **int** &size, **int** count=1){

Customer \*newArr = **new** Customer[size+count];

**for**(**int** i=0; i<size; i++) newArr[i] = customers[i];

puts("");

**for**(**int** i=0; i<count; i++){

printf("Фамилия: "); cin >> newArr[size].fa;

printf("Имя: "); cin >> newArr[size].im;

printf("Отчество: "); cin >> newArr[size].ot;

printf("Адрес: "); cin >> newArr[size].address;

printf("Дата (год, месяц, день): ");

cin >> newArr[size].date.year;

cin >> newArr[size].date.month;

cin >> newArr[size].date.day;

cin.get(); // считывает из потока Enter который пользователь нажимает после ввода даты

cout << endl;

}

size += count;

**delete** [] customers;

customers = newArr;

}

**void** **pop**(Customer \*customers, **int** &size, **int** pos){

**for**(**int** i=pos-1; i<size-1; i++) customers[i] = customers[i+1];

size--;

}

**void** **strChecker**(**char** \*str, **int** length){

**bool** success=true;

**int** i=0;

**while**(success){

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin >> str;

**while**(i<strlen(str)){

**if** (strchr(num, str[i])){

printf("Вы ввели символы! Повторите ввод: ");

**break**;

}

**else** **if** (!strchr(en, str[i]) && !strchr(EN, str[i])){

printf("Приветствуется только ввод на латинице! Повторите ввод: ");

**break**;

}

**else**{

success = false;

cout << "Введите значения строки: ";

}

i++;

}

}

puts("");

}

**void** **print**(Customer \*Obj, **int** amount){

cout << "№\t\t" << "Фамилия\t\t\t" << "Имя\t\t\t" << "Отчество\t\t" << "Адрес\t\t" << "Дата\t\t" << endl;

cout << "=============================================================================" << endl;

**for** (**int** i=0; i<amount; i++)

{

cout << i + 1 << "\t\t" << Obj[i].fa << "\t\t" << Obj[i].im

<< "\t\t" << Obj[i].ot << "\t\t" << Obj[i].address << "\t"

<< Obj[i].date.year << '/' << Obj[i].date.month << '/' << Obj[i].date.day

<< '\t' << endl;

}

}

**int** **save**(**char** **const** \*filename, **struct** Customer \*customer, **int** n){

FILE \*fp=nullptr;

**char** \*c=nullptr;

// число записываемых байтов

**int** size=n\***sizeof**(Customer);

**if** ((fp=fopen(filename, "wb"))==NULL)

{

perror("Ошибка открытия файла!");

**return** 1;

}

// записываем количество структур

c = (**char**\*)&n;

**for** (**int** i=0; i<**sizeof**(**int**); i++){

putc(\*c++, fp);

}

// посимвольно записываем в файл все структуры

c = (**char**\*)customer;

**for** (**int** i=0; i<size; i++){

putc(\*c, fp);

c++;

}

fclose(fp);

**return** 0;

}

// загрузка из файла массива структур

**int** **load**(**char** **const** \*filename){

FILE \*fp=nullptr;

**char** \*c=nullptr;

**int** m=**sizeof**(**int**);

**int** n, i;

// выделяем блок память размером sizeof(int) для количества данных

**int** \*pti = (**int**\*)malloc(m); // возвращает указатель на начало блока

**if** ((fp=fopen(filename, "r"))==NULL){

perror("Ошибка открытия файла!");

**return** 1;

}

// считываем количество структур

c = (**char**\*)pti;

**while** (m>0){

i = getc(fp);

**if** (i == EOF) **break**;

\*c = i;

c++;

m--;

}

//получаем число элементов

n = \*pti;

// выделяем память для считанного массива структур

Customer \*customer = (Customer\*)malloc(n\***sizeof**(Customer));

c = (**char**\*)customer;

// после записи считываем посимвольно из файла

**while** ((i=getc(fp))!=EOF){

\*c = i;

c++;

}

// перебор загруженных элементов и вывод на консоль

printf("\n%d количество лиц\n\n", n);

cout << "№\t\t" << "Фамилия\t\t\t" << "Имя\t\t\t" << "Отчество\t\t" << "Адрес\t\t" << "Дата\t\t" << endl;

cout << "=============================================================================" << endl;

**for** (**int** k=0; k<n; k++){

cout << k + 1 << "\t\t" << (customer+k)->fa << "\t\t" << (customer+k)->im

<< "\t\t" << (customer+k)->ot << "\t\t" << (customer+k)->address << "\t"

<< (customer+k)->date.year << '/' << (customer+k)->date.month << '/' << (customer+k)->date.day

<< '\t' << endl;

}

free(pti);

free(customer);

fclose(fp);

**return** 0;

}

// код проверки даты

/\*

Месяц - кол-вл дней:

1 - 31 день

12 - 31 день

3 - 31 день

5 - 31 день

7 - 31 день

8 - 31 день

10 - 31 день

4 - 30 дней

6 - 30 дней

9 - 30 дней

11 - 30 дней

2 - 28 дней (29 дней в вискосном)

\*/

**bool** **checkDate**(**int** y, **int** m, **int** d){

**bool** flag=false;

**if**(y>0){

**if**(y % 4 == 0){

**if**(m==2){

**if**(d == 29) printf("OK\n");

}

**else** printf("Год високосный! Февраль - 29дн\n");

}

**if**(m>0 && m<=12){

**if**(d>0 && d<=31){

**if**((m==1 || m==3 || m==5 || m==7 || m==8 || m==10 || m==12) && d==31) flag=true;

**else** **if**(m==4 || m==6 || m==9 || m==11) flag=true;

**else** flag=true;

}

**else** printf("Количество дней не может быть отрицательным!\n");

}

**else** printf("Номер месяца не может быть отрицательным!\n");

}

**else** printf("Год не может быть отрицательным(учитывается только Н.Э.)!\n");

**return** flag;

}

**int** **main**() {

setlocale(LC\_ALL, "Rus");

**int** amount=0; // количество лиц

printf("Введите длину списка лиц, записавшихся на покупку товара: ");

**bool** success = true;

**while**(success){

cin >> amount;

**if** (!cin.good()){

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

printf("Вы ввели не число! Повторите ввод: ");

}

**else** **if**(cin.good() && amount<0){

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

printf("Длина списка не может быть отрицательной или равным нулю! Повторите ввод: ");

}

**else**

success = false;

puts("");

}

Customer \*customers = **new** Customer[amount];

**for**(**int** i=0; i<amount; i++){

printf("Фамилия: "); cin >> customers[i].fa;

printf("Имя: "); cin >> customers[i].im;

printf("Отчество: "); cin >> customers[i].ot;

printf("Адрес: "); cin >> customers[i].address;

**bool** flag=false;

**while**(!flag){

printf("Дата (год, месяц, день): ");

cin >> customers[i].date.year;

cin >> customers[i].date.month;

cin >> customers[i].date.day;

flag=checkDate(customers[i].date.year, customers[i].date.month, customers[i].date.day);

}

cin.get(); // считывает из потока Enter который пользователь нажимает после ввода даты

cout << endl;

}

print(customers, amount);

//удаление повторяющихся записей

**int** temp=0;

**while**(temp<amount){

**for**(**int** i=1; i<amount; i++){

**if**((customers+temp)->address == (customers+i)->address){

**if**((customers+temp)->fa == (customers+i)->fa){

**if**((customers+temp)->im == (customers+i)->im){

**if**((customers+temp)->ot == (customers+i)->ot){

pop(customers, amount, i);

}

}

}

}

}

temp++;

}

print(customers, amount);

quickSort(customers, amount);

print(customers, amount);

selectionSort(customers, amount);

print(customers, amount);

Date key=customers[0].date;

**int** t;

t = binSearch(customers, amount, key);

**if**(t!=-1) printf("\nИндекс искомого элемента: %d\n", t);

**else** printf("Элемент отсутствует в списке");

**char** **const** \*path= "/Users/timocruz/Desktop/БГУИР/ОАиП/Lab13/Lab13/file.txt";

save(path, customers, amount);

load(path);

push\_back(customers, amount);

save(path, customers, amount);

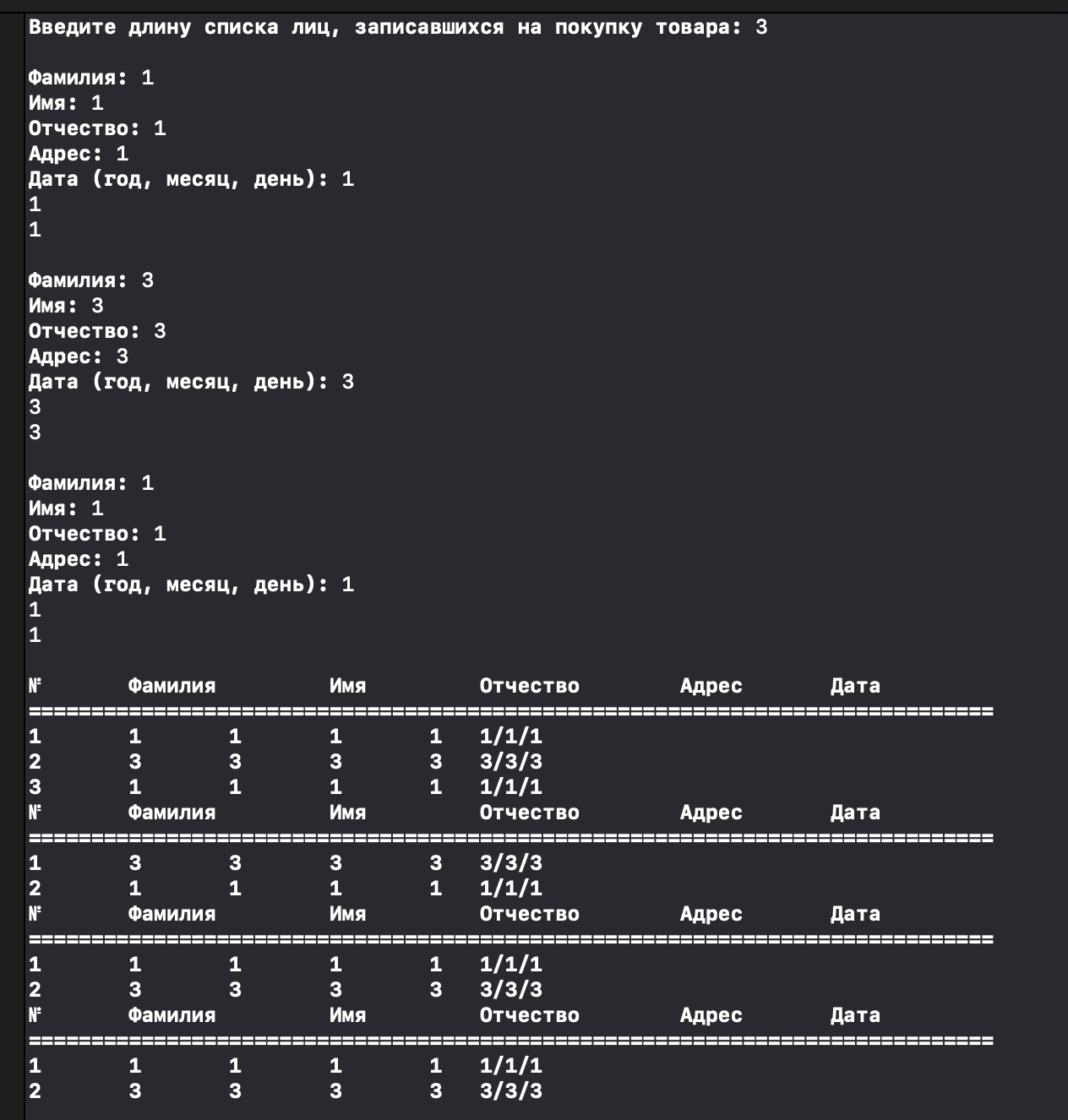
load(path);

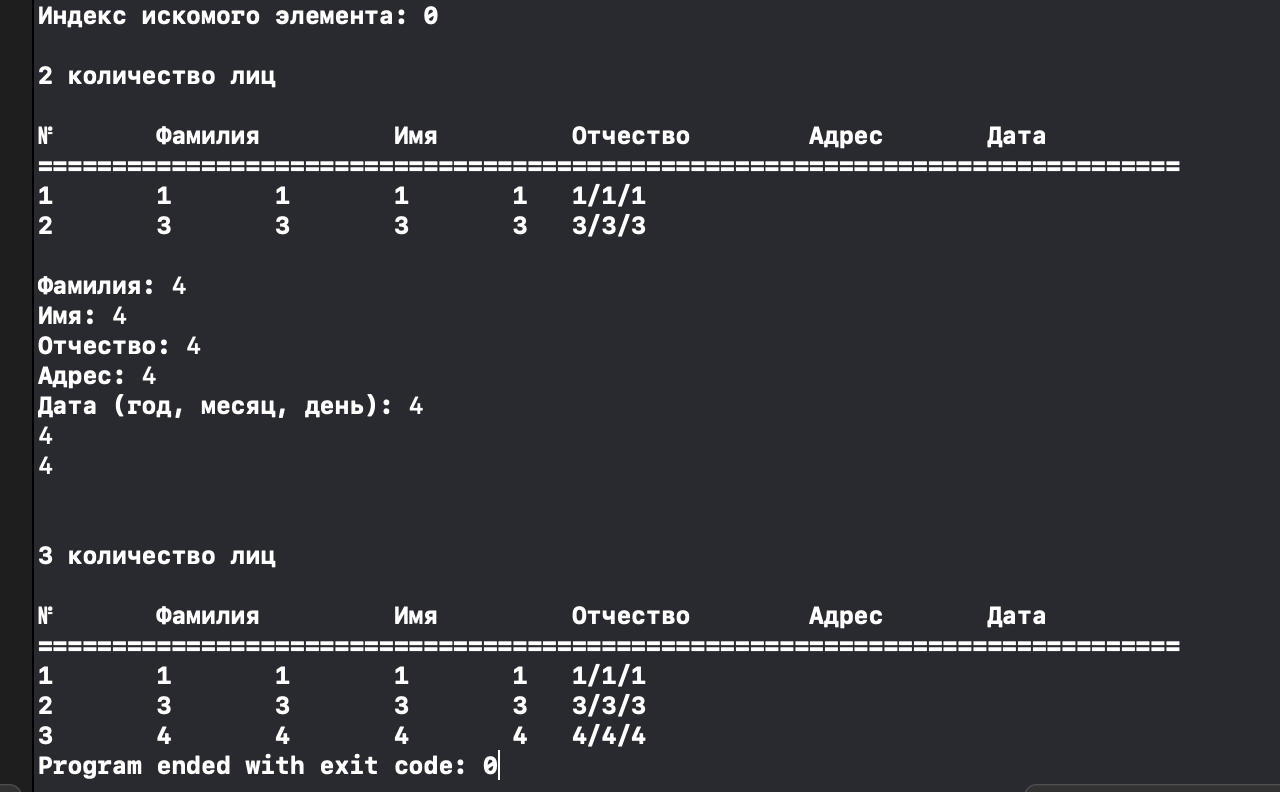
**delete** [] customers;

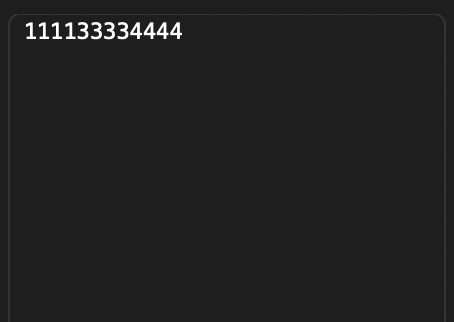
**return** 0;

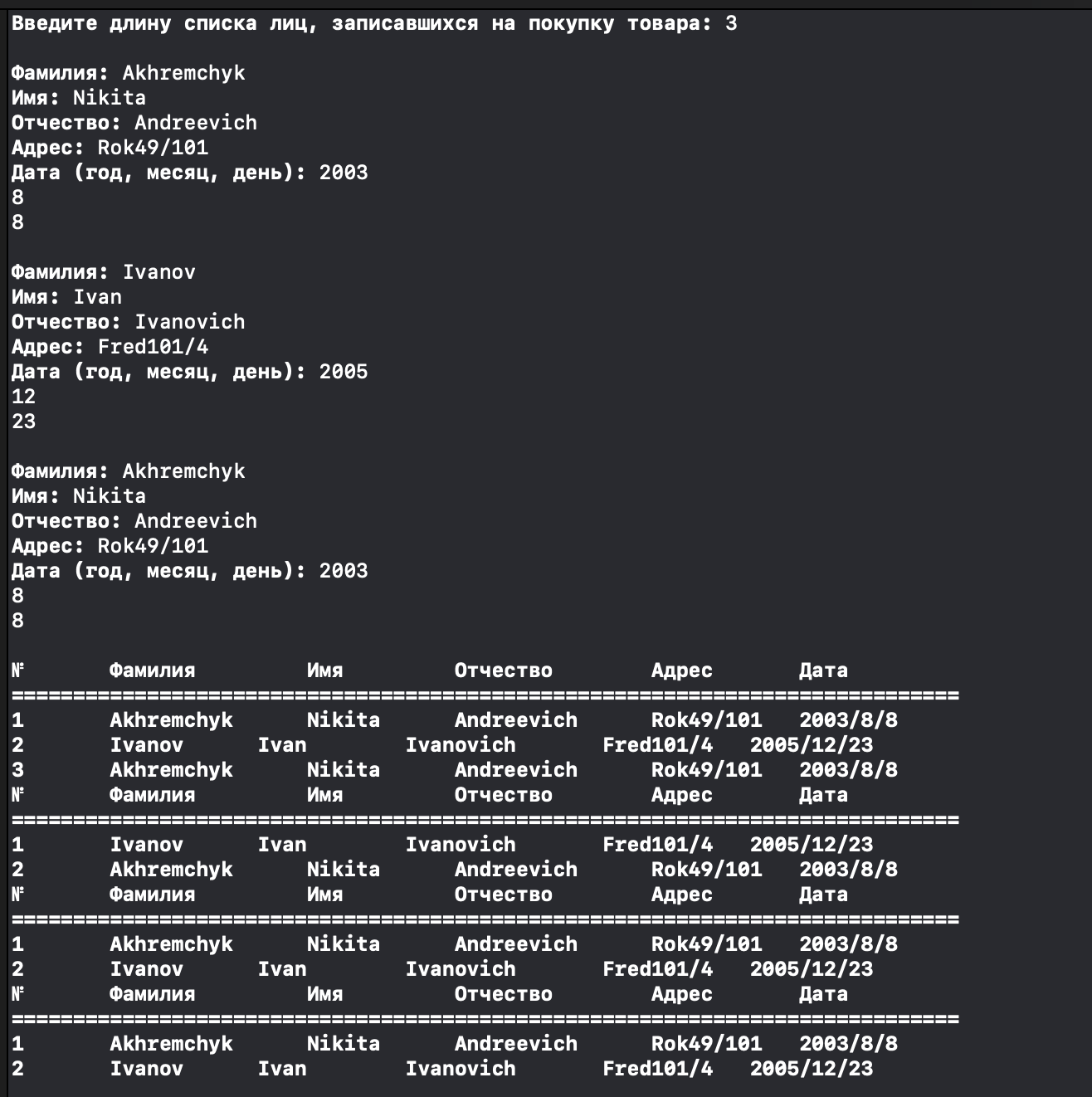
}

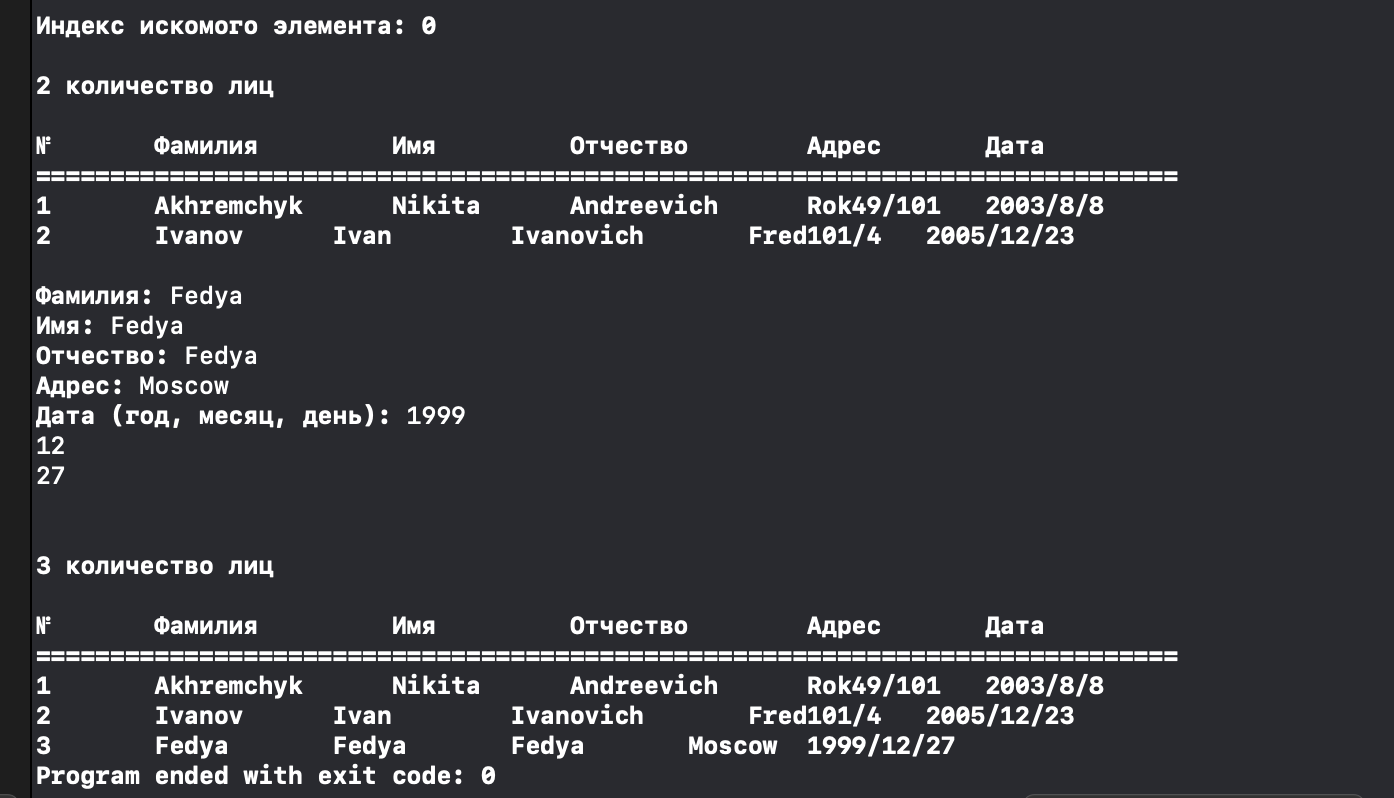
**Результат вывода программы:**

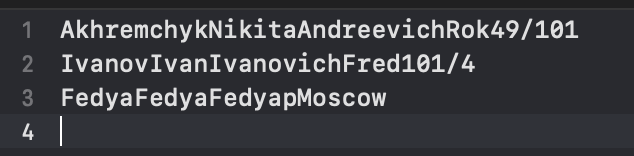












**Выводы:** Проведя данную лабораторную работу, я научился работать с использованием алгоритмов поиска и сортировки в массивах в С++. Также закрепил знания по прошлым темам.