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**

**Индивидуальная работа**

**Основы алгоритмизации и программирования**

**Вариант №6**

Выполнил:

студент гр. КТбо1-8

Лаврухин К. М.

«\_\_\_» 2017 г.

Проверил:

Профессор кафедры ИАСБ

Беляков Станислав Леонидович

«\_\_\_» 2017 г.

Таганрог – 2017

Оглавление

[Формулировка индивидуального задания 3](#_Toc501907825)

[Алгоритм выполнения работы 4](#_Toc501907826)

[Программный код 5](#_Toc501907827)

[Тестирование программы 9](#_Toc501907828)

[Вывод 10](#_Toc501907829)

# Формулировка индивидуального задания

Задание состоит в разработке программы, которая считывает настроечные параметры и формирует отчет по имеющимся данным в файлах. Отчет записывается в текстовый файл. Число записей в каждом из описанных выше файлов произвольно.

Вариант задания:

В отчетный файл: Перечислить ФИО клиентов, пользовавшихся указанной услугой в дневное время.

Параметры: Наименование услуги.

# Алгоритм выполнения работы

Сначала программа проверяет файлы на заполненность, если один из файлов не заполнен – выводит на экран текст с предупреждением и закрывается, если все файлы в порядке, то пользователь вводит наименование услуги. Затем в первой функции введенное наименование сравнивается с каждым в файле, если есть совпадение, то берет код этой услуги и возвращает в мэйн. Там этот код передается в другую функцию, где сравнивается с каждым кодом во втором файле, если есть совпадение, то берет время использование данной услугой и, если удостоверяется, что услуга была использована в дневное время, помещает номер телефона в массив. Так алгоритм делает с каждым номером телефона. Затем в третьей функции каждый номер в строке сравнивается с каждым номером в массиве. Если они равны, то пишет ФИО, на который оформлен данный номер, в файл Report.txt. Так алгоритм делает с каждым номером телефона.

# Программный код

#define \_CRT\_SECURE\_NO\_WARNINGS

#define MAX\_LENGHT\_SERVICE 90

#define MAX\_LENGHT\_STRUCT\_SERVICE\_ARRAY 100

#define MAX\_LENGHT\_STRING\_IN\_FILE 200

#define MAX\_LENGHT\_PHONE\_NUMBER 100

#define MAX\_LENGHT\_FIO 256

#define MAX\_LENGHT\_STRING\_ARRAY 500

#define MAX\_LENGHT\_HOUR 2

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <stdlib.h>

struct services

{

char nameOfService[MAX\_LENGHT\_SERVICE];

char serviceCode[MAX\_LENGHT\_STRUCT\_SERVICE\_ARRAY];

};

struct renderServices

{

char phoneNumber[MAX\_LENGHT\_PHONE\_NUMBER];

char serviceCode[MAX\_LENGHT\_STRUCT\_SERVICE\_ARRAY];

char hourUseTheService[MAX\_LENGHT\_HOUR];

};

struct clients

{

char FIO[MAX\_LENGHT\_FIO];

char phoneNumber[MAX\_LENGHT\_PHONE\_NUMBER];

};

void checkFile(FILE \*RenderServices, FILE \*Client, FILE \*Services);

int checkServiceInFile(char \*service, FILE \*services);

void checkForCoincidenceInServices(int codeOfService, FILE\* Report);

void phoneNumberSearch(FILE \*RenderServices, int code, char\* phoneNumber);

void searchClients(FILE\* Client, FILE\* Report, char \*phoneNumbers);

int main()

{

system("chcp 1251");

system("cls");

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

FILE\* Client = fopen(".clientInfo.txt", "r");

FILE\* Services = fopen(".servicesInfo.txt", "r");

FILE\* RenderServices = fopen(".renderServicesInfo.txt", "r");

checkFile(RenderServices, Client, Services);

printf("Введите услугу: ");

char service[MAX\_LENGHT\_SERVICE];

char phoneNumber[MAX\_LENGHT\_PHONE\_NUMBER];

fgets(service, MAX\_LENGHT\_SERVICE, stdin);

service[strlen(service) - 1] = '\0';

int codeOfService = checkServiceInFile(service, Services);

checkForCoincidenceInServices(codeOfService, Report);

phoneNumberSearch(RenderServices, codeOfService, phoneNumber);

searchClients(Client, Report, phoneNumber);

return 0;

}

void checkFile(FILE \*RenderServices, FILE \*Client, FILE \*Services)

{

if (RenderServices == NULL)

{

printf("Пожалуйста, введите текст в файл .renderServicesInfo.txt в след. раз");

exit(0);

}

else if (Client == NULL)

{

printf("Пожалуйста, введите текст в файл .clientInfo.txt в след. раз");

\_getch();

exit(0);

}

else if (Services == NULL)

{

printf("Пожалуйста, введите текст в файл .servicesInfo.txt в след. раз");

\_getch();

exit(0);

}

}

int checkServiceInFile(char \*service, FILE \*Services)

{

struct services arrayOfService;

int code = 0, j = 0, k = 0;

char massiveOfServices[MAX\_LENGHT\_STRUCT\_SERVICE\_ARRAY];

while (strcmp(arrayOfService.nameOfService, service) != 0 && !feof(Services))

{

j = 0;

k = 0;

fgets(massiveOfServices, MAX\_LENGHT\_STRUCT\_SERVICE\_ARRAY, Services);

while (massiveOfServices[j] != ',')

{

arrayOfService.nameOfService[j] = massiveOfServices[j];

j++;

}

arrayOfService.nameOfService[j] = '\0';

j++;

if (strcmp(arrayOfService.nameOfService, service) == 0)

{

while (massiveOfServices[j] != ',')

{

arrayOfService.serviceCode[k] = massiveOfServices[j];

j++; k++;

}

}

}

code = atoi(arrayOfService.serviceCode);

return code;

}

void checkForCoincidenceInServices(int codeOfService, FILE\* Report)

{

if (codeOfService == NULL)

{

fprintf(Report, "Нет данных.");

\_getch();

exit(0);

}

}

void phoneNumberSearch(FILE \*RenderServices, int code, char\* phoneNumber)

{

struct renderServices renderService;

char arrayOfrenderService[MAX\_LENGHT\_STRING\_ARRAY];

phoneNumber[0] = '\0';

int iteration = 0;

while (!feof(RenderServices))

{

int j = 0, k = 0, i = 0;

fgets(arrayOfrenderService, MAX\_LENGHT\_STRING\_ARRAY, RenderServices);

while (arrayOfrenderService[j] != ',')

{

renderService.phoneNumber[j] = arrayOfrenderService[j];

j++;

}

j++;

while (arrayOfrenderService[j] != ',')

{

renderService.serviceCode[k] = arrayOfrenderService[j];

j++; k++;

}

int reservCode = atoi(renderService.serviceCode);

if (reservCode == code)

{

j = j + 13;

while (arrayOfrenderService[j] != ':')

{

renderService.hourUseTheService[i] = arrayOfrenderService[j];

j++; i++;

}

int reservHour = atoi(renderService.hourUseTheService);

if (reservHour <= 16 && reservHour >= 12)

{

strcat(phoneNumber, renderService.phoneNumber);

iteration++;

phoneNumber[iteration \* 10] = '\0';

}

while (arrayOfrenderService[j] != '\n')

{

j++;

}

}

}

}

void searchClients(FILE\* Client, FILE\* Report, char \*phoneNumbers)

{

struct clients clientsInfo;

while (!feof(Client))

{

int j = 0, k, m = 0;

char arrayOfClients[MAX\_LENGHT\_STRING\_ARRAY];

char phoneNumberForCheck[MAX\_LENGHT\_PHONE\_NUMBER];

fgets(arrayOfClients, MAX\_LENGHT\_STRING\_ARRAY, Client);

while (arrayOfClients[j] != ',')

{

clientsInfo.FIO[j] = arrayOfClients[j];

j++;

}

clientsInfo.FIO[j] = '\0';

j = j + 2;

while (arrayOfClients[j] != ',')

{

clientsInfo.phoneNumber[m] = arrayOfClients[j];

j++; m++;

}

clientsInfo.phoneNumber[m] = '\0';

k = 0;

while (phoneNumbers[k] != '\0')

{

int t = 0;

int counter = 0;

while (counter != 10)

{

phoneNumberForCheck[t] = phoneNumbers[k];

t++; k++; counter++;

}

int checkNumbersForCoincidence = 0;

while ((phoneNumberForCheck[checkNumbersForCoincidence] == clientsInfo.phoneNumber[checkNumbersForCoincidence]) && (checkNumbersForCoincidence < 9))

{

checkNumbersForCoincidence++;

}

if (checkNumbersForCoincidence == 9)

{

fprintf(Report, "%s, ", clientsInfo.FIO);

}

}

while (arrayOfClients[j] != '\n')

{

j++;

}

}

}

# Тестирование программы

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

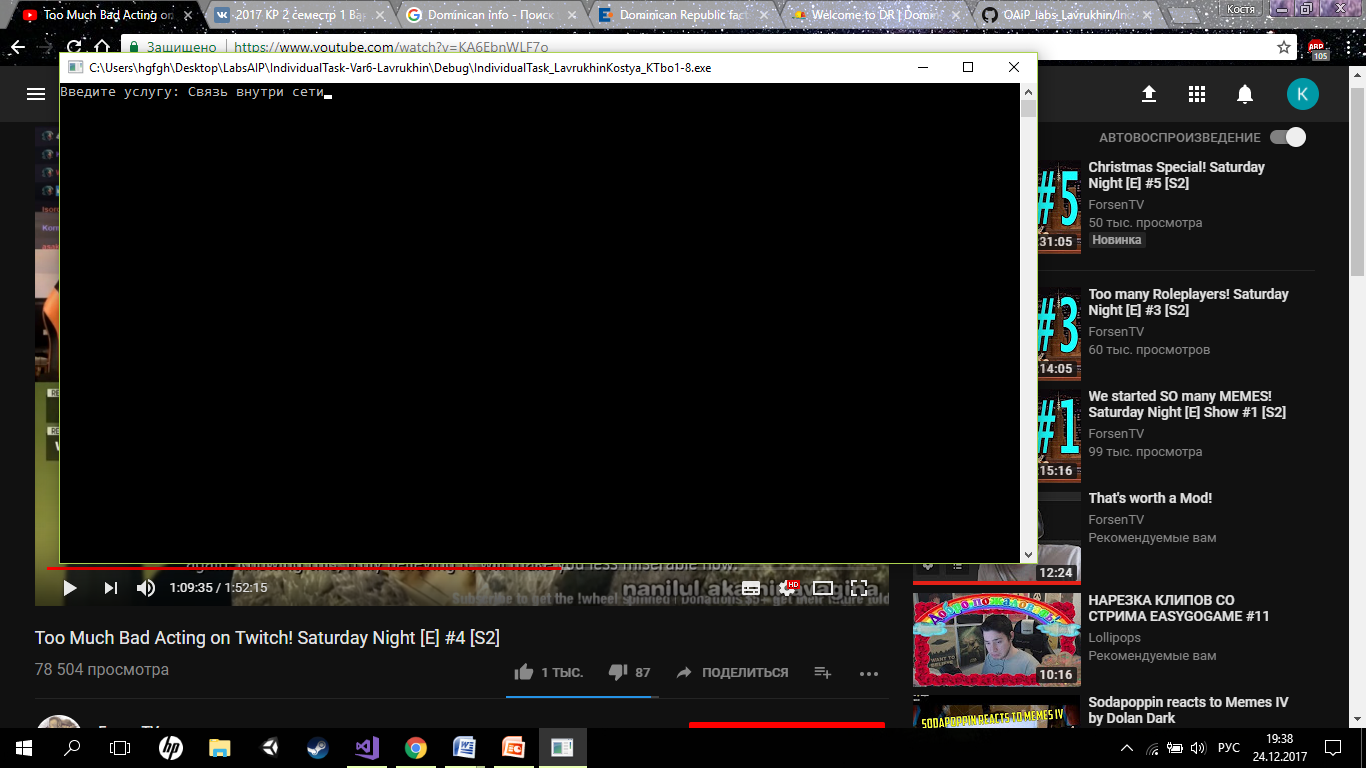


рис.1 Главное меню

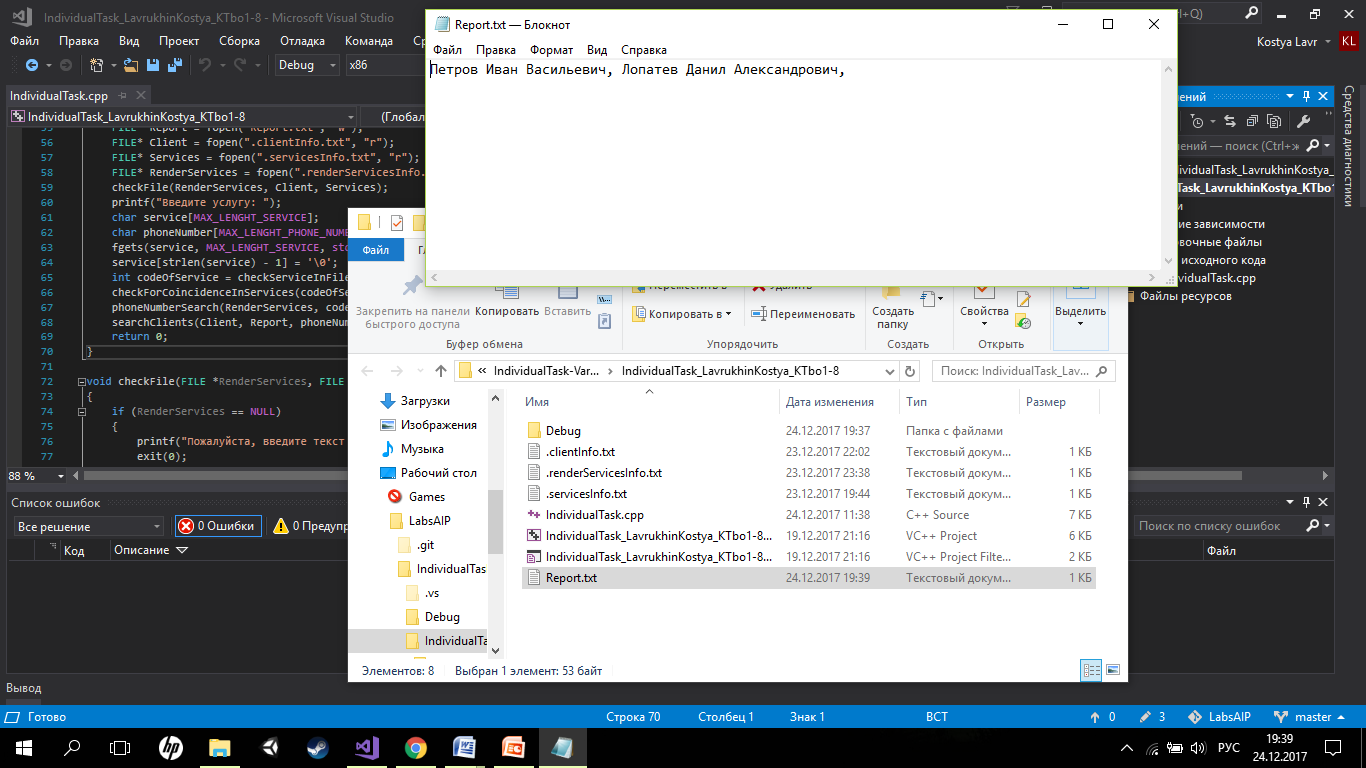


рис.2 Вывод результата работы в файл

# Вывод

В индивидуальном задании были использованы: сложные типы данных, работа с текущей датой и временем и операции над ними. Благодаря этой работе были укреплены навыки программирования.