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**

**По индивидуальному заданию**

**по дисциплине**

**Основы алгоритмизации и программирования**

**Вариант - 8**

Выполнил:

студент гр. КТбо1-10

Козловский Д.А

Проверил:

Профессор кафедры ИАСБ

Барковский С.А.

«\_\_\_» 2020 г.

Таганрог – 2020

Код Программы:

|  |
| --- |
| #include <iostream> |
|  | #include <fstream> |
|  | #include <windows.h> |
|  | #include <ctime> |
|  |  |
|  | using namespace std; |
|  |  |
|  | int oper = 0; // Переменная хранит количество операций |
|  | int user = 0; // Переменная хранит количество участников |
|  |  |
|  |  |
|  | struct People { |
|  |  |
|  | char F[25]; |
|  | char I[25]; |
|  | char O[25]; |
|  | char phone[20]; |
|  | char date\_1[20]; |
|  | char date\_2[20]; |
|  | char money\_1[10]; |
|  | char money\_2[10]; |
|  |  |
|  | }people[10] {}; |
|  |  |
|  | struct services { |
|  |  |
|  | char name\_ser[40]; |
|  | char id\_ser[10]; |
|  | char price\_ser[10]; |
|  | char min\_ser[10]; |
|  |  |
|  | }serv[11] {}; |
|  |  |
|  | struct info\_serv { |
|  |  |
|  | char phone[20]; |
|  | char id[10]; |
|  | char date[20]; |
|  | char time[20]; |
|  | char price[10]; |
|  |  |
|  | }info[40] {}; |
|  |  |
|  |  |
|  | int main() { |
|  |  |
|  | SetConsoleCP(1251); |
|  | SetConsoleOutputCP(1251); |
|  |  |
|  | cout << "Индивидуальное задание: " << endl |
|  | << "Получить список клиентов, которые имеют задолженность не более заданной и не \n" |
|  | <<"пользовались услугами в течение предыдущего месяца от момента запуска программы" << endl << endl; |
|  |  |
|  | // Считываем Параметры |
|  | ifstream Parametr; |
|  | Parametr.open("Param.txt"); |
|  | int line = 0; |
|  |  |
|  | char DANO[10]; |
|  |  |
|  | while (!Parametr.eof()) { |
|  |  |
|  | Parametr >> DANO; |
|  |  |
|  | } |
|  |  |
|  | // Считываем все данные в структуры |
|  |  |
|  | ifstream file\_in; // Открыли user\_id |
|  | file\_in.open("user\_id.txt"); |
|  |  |
|  | line = 0; |
|  | while (!file\_in.eof()) { // Передаём данные в структуру People |
|  |  |
|  | file\_in >> people[line].F >> people[line].I >> people[line].O >> people[line].phone >> people[line].date\_1 >> people[line].date\_2 >> people[line].money\_1 >> people[line].money\_2; |
|  |  |
|  | line++; |
|  | } |
|  |  |
|  | user = line; |
|  |  |
|  | file\_in.close(); |
|  |  |
|  | ifstream file\_2in; // Открыли services |
|  | file\_2in.open("services.txt"); |
|  |  |
|  | line = 0; |
|  |  |
|  | while (!file\_2in.eof()) { |
|  |  |
|  | file\_2in >> serv[line].name\_ser >> serv[line].id\_ser >> serv[line].price\_ser >> serv[line].min\_ser; |
|  | line++; |
|  | } |
|  |  |
|  | file\_2in.close(); |
|  |  |
|  | ifstream file\_3in; // Открыли info |
|  | file\_3in.open("info.txt"); |
|  |  |
|  | line = 0; |
|  |  |
|  | while (!file\_3in.eof()) { // Передаём данные в структуру info |
|  |  |
|  | file\_3in >> info[line].phone >> info[line].id >> info[line].date >> info[line].time >> info[line].price; |
|  | line++; |
|  | } |
|  |  |
|  | oper = line; |
|  |  |
|  | file\_3in.close(); // Закрыли info |
|  |  |
|  |  |
|  | ofstream file\_out; // открываем файл для отчётов |
|  | file\_out.open("Report.txt"); |
|  |  |
|  | // Узнаём месяц запуска программы |
|  |  |
|  | time\_t rawtime; |
|  | struct tm\* timeinfo; |
|  | char month\_now[80]; // хранит текущий месяц |
|  | char year\_now[80]; // хранит текущий год |
|  |  |
|  | time(&rawtime); // текущая дата в секундах |
|  | timeinfo = localtime(&rawtime); // текущее локальное время, представленное в структуре |
|  |  |
|  | strftime(month\_now, 80, " %m.", timeinfo); // Получаем месяц #12 |
|  | strftime(year\_now, 80, " %Y.", timeinfo); // Получаем год #2020 |
|  |  |
|  | int k = 0; // переменная нужна для проверки |
|  |  |
|  | for (int i = 0; i < user; i++) { |
|  | // Получения списка задолжности не больше заданной |
|  | if (atoi(people[i].money\_1) < atoi(DANO)) { |
|  |  |
|  | for (int j = 0; j < oper; j++) { |
|  |  |
|  | char month\_oper[6] = { info[j].date[3] , info[j].date[4] }; // хранит месяц операции |
|  | char year\_oper[6] = { info[j].date[6], info[j].date[7], info[j].date[8], info[j].date[9] }; // хранит год операции |
|  |  |
|  | // проверяем пользовались услугами абоненты в прошлом месяце |
|  | if ((atoi(month\_now) - 1) == atoi(month\_oper) and (atoi(year\_now)) == atoi(year\_oper)) { |
|  |  |
|  | file\_out << "Имеет задолжность " << people[i].F << endl; |
|  | k++; |
|  |  |
|  | } |
|  | } |
|  | } |
|  | } |
|  |  |
|  | file\_out.close(); |
|  |  |
|  | if (k != 0) { |
|  | cout << "Программа завершина успешно!" << endl; |
|  | } |
|  | else { |
|  | cout << "Клиентов имеющих задолжность " << DANO << " нету!"; |
|  | } |
|  |  |
|  | return 0; |
|  | } |

Принцип работы программы:

# Individual task | work 8

### Получить список клиентов, которые имеют задолженность не более заданной и не пользовались услугами в течение предыдущего месяца от момента запуска программы

Принцип работы программы:

### 1. Открываем файл Param.txt

* Считываем параметры в DANO
* Закрываем файл Param.txt

### 2. Открываем файл user\_id.txt

* Записываем данные в структуру People
* Закрываем файл user\_id.txt

### 3. Открываем файл services.txt

* Записываем данные в структуру Services
* Закрываем файл services.txt

### 4. Открываем файл info.txt

* Записываем данные в структуру info\_serv
* Закрываем файл Info.txt

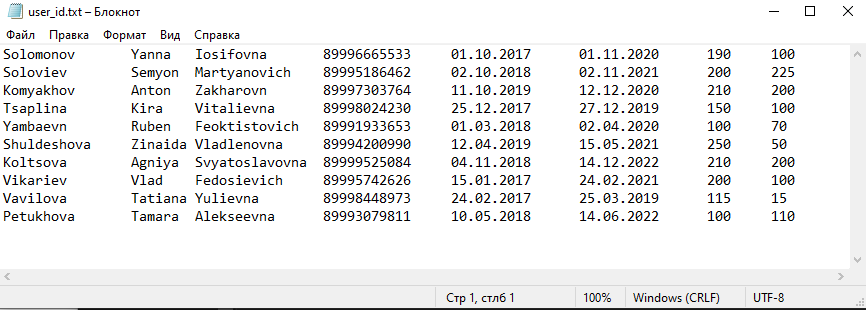
### 5. Считываем год и месяц, когда была запущена программа на PC

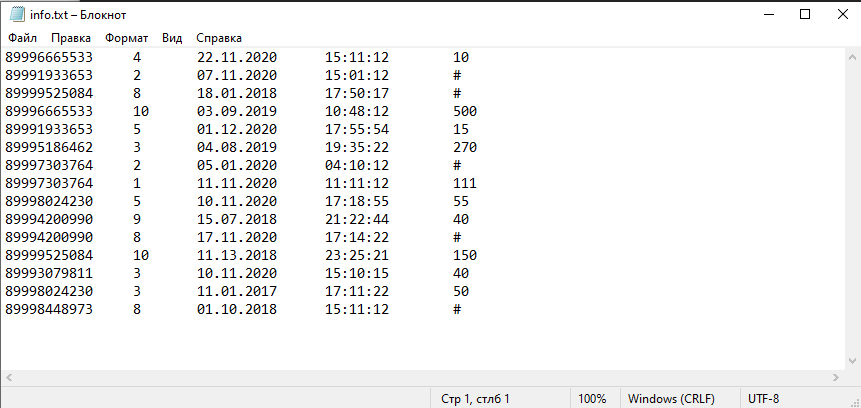
### 6. Смотрим какие операции совершал первый абонент в прошлом году

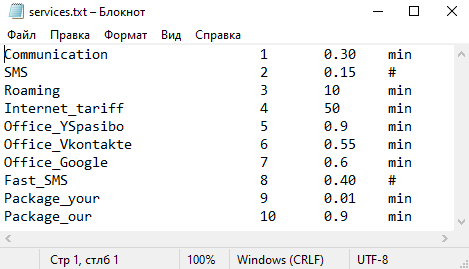
* Открываем файл для записи в отчётов Report.txt
* Гуляем по циклу for, по всем пользователям из user\_id
* Получения списка задолжности не больше заданной значения из DANO
* Проверяем пользовались услугами абоненты в прошлом месяце, а так же сравниваем номера телефонов из info и user\_id
* Записываем результаты в файл Report.txt
* Закрываем файл отчёта Report.txt

Пример:

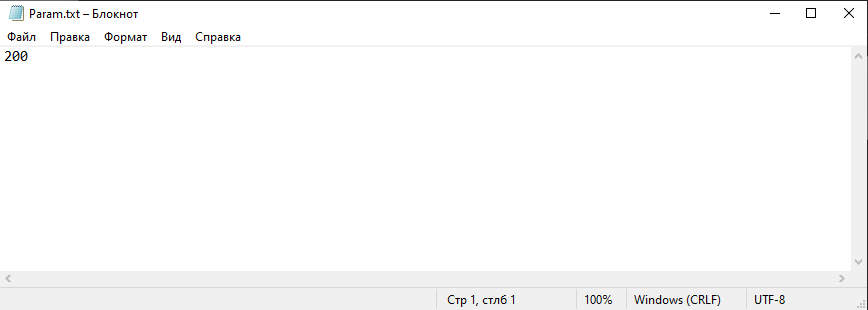
Заполним данные файлов user\_id.txt



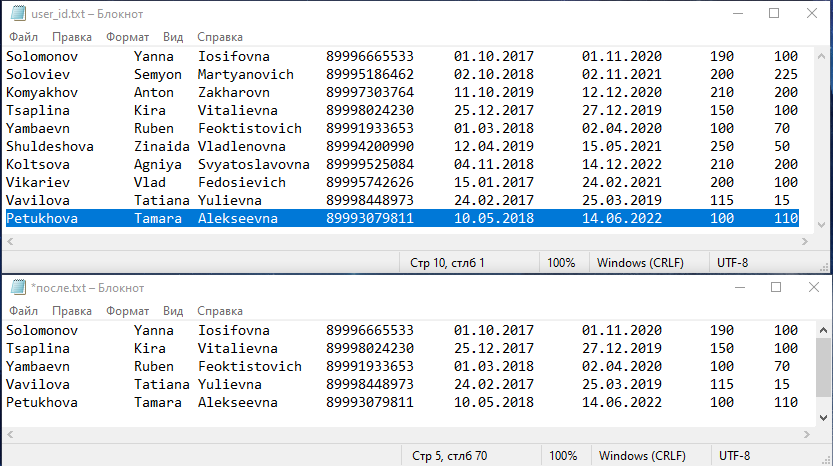
Заполним данные файлов info.txt 

Заполним данные файлов services.txt 

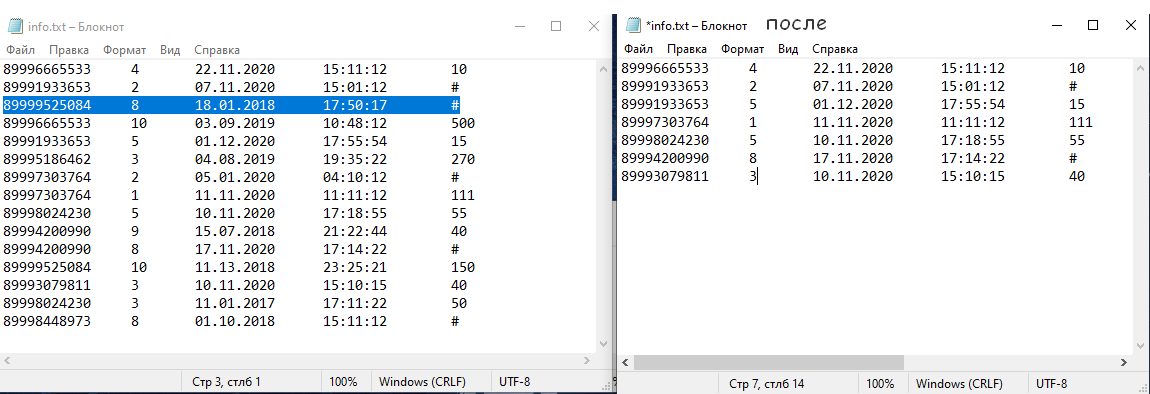
Вводим Параметры в файл Param.txt , для примера возьмём “200”



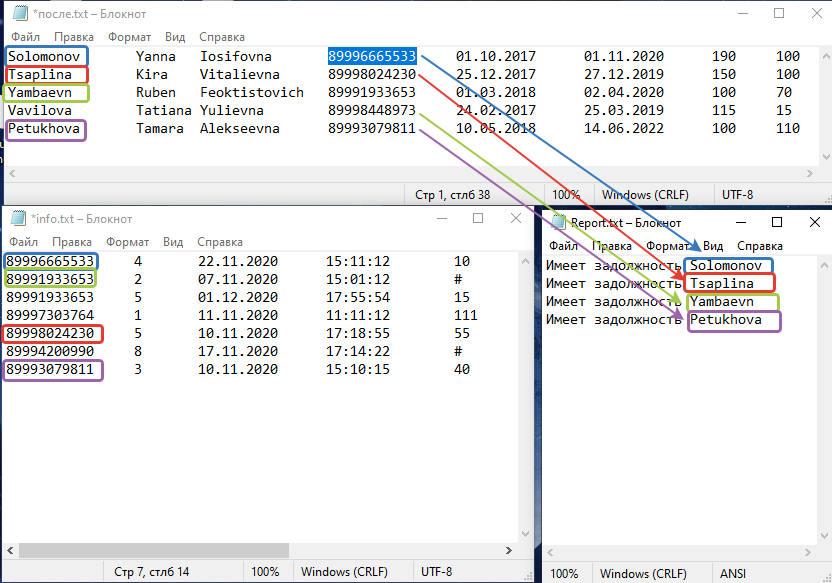
Программа ищет абонентав у которых задолжность ниже 200



Затем она сравнивает время на pc (месяц и год) с совершонной операцией в прошлом месяце этого года



Дальше программа сравнивает каждого пользователя и его операцию



Остальные не вывелись, потому что их задолжность выше 200, либо они пользовались услугой в прошлом месяце

После ввода, видим, что “Программа завершина успешно!”, это говорит о том что данные сохранились в файл Report.txt

Откроем его и увидим результат

