Задача №3

Разработать программу, в которой будет организовано меню, выбор функций меню должно быть организовано по функциональной клавише. Вся информация должна храниться в массиве структур, с возможностью их записи в файл. Организовать сортировку данных одним из методов (быстрая, Шелла,

Пузырьковая), вывод результатов сортировки должен быть в табличной форме.

**Функции программы:**

• функция для ввода данных пользователем;

• функция для записи данных в файл;

• чтение данных из файла;

• вывод данных на экран (в виде таблицы);

• дополнение данных;

• удаление данных;

• поиск информации по атрибуту;

• сортировка.

**Состав и структура исходных данных**

Данная программа может применяться в области телефонных услуг. Таблица содержит некоторые коды городов, стоимости одной минуты разговора жителей Москвы с людьми, живущими в других городах России и Белоруссии. Программа позволяет вести учет телефонных разговоров, подсчитывать

стоимость услуг.

**Выходные данные:**

* Начисление, руб. (Стоимость разговора равна произведению стоимости одной минуты разговора на время разговора в минутах.)
* Итого,руб. (Стоимость разговора, учитывая скидку, равна разности стоимости разговора без скидки и скидки, если она есть.)
* Найти код города Псков (вывести город и код города)
* Сортировка городов по стоимости минуты разговора (по убыванию, самый дорогой вверху списка)

// ВитвицкийИндЗад№3.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include<iostream>

#include <string>

#include <cstdlib>

#include<iomanip>

#include<vector>

#include <locale>

#include <fstream>

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

class PhoneCall

**{**

int call\_id**;**

string calldate**;**

string city**;**

int city\_id**;**

char phonenumber**[**128**];**

double priceperminute**;**

int callduration**;**

char callbeginning**[**64**];**

string invoicedate**;**

double charge**;**

double discount**;**

double total**;**

public**:**

PhoneCall**()** **{};**

**~**PhoneCall**()** **{};**

void set\_data**()**

**{**

cout **<<** "Enter data:" **<<** endl**;**

cout **<<** "Enter call\_id: "**;** cin **>>** call\_id**;**

cout **<<** "Enter call date: "**;**cin **>>** calldate**;**

cout **<<** "Enter the city from which the call was made: "**;** cin **>>** city**;**

cout **<<** "Enter the code of the city: "**;** cin **>>** city\_id**;**

cout **<<** "Enter phone number: "**;** cin **>>** phonenumber**;**

cout **<<** "Enter price per minute: "**;** cin **>>** priceperminute**;**

cout **<<** "Enter the call's duration: "**;** cin **>>** callduration**;**

cout **<<** "Enter the time of beginning of the call: "**;** cin **>>** callbeginning**;**

cout **<<** "Enter the day and month of reception of the invoice: "**;**cin **>>** invoicedate**;**

cout **<<** "Enter the discount in percent :"**;** cin **>>** discount**;**

**}**

void Charge**()**

**{**

charge **=** priceperminute **\*** callduration**;**

**}**

void Total**()**

**{**

total **=** charge **-** **(**charge **\*** discount**);**

**}**

void show**()**

**{**

cout **<<** setw**(**4**)** **<<** call\_id **<<** setw**(**12**)** **<<** calldate **<<** setw**(**8**)** **<<** city **<<** setw**(**20**)** **<<**city\_id **<<** setw**(**10**)** **<<** priceperminute **<<** setw**(**12**)** **<<** callduration **<<** setw**(**15**)** **<<** callbeginning **<<** setw**(**15**)** **<<** invoicedate **<<** setw**(**8**)** **<<** charge**<<**setw**(**10**)** **<<** discount **<<**setw**(**8**)** **<<**total**;**

**}**

void display**()**

**{**

cout **<<** setw**(**4**)** **<<** city **<<** setw**(**12**)** **<<** city\_id**;**

**}**

void edit**()**

**{**

char c**[**64**];**

char p**[**128**];**

int num**;**

int d**;**

char key**;**

**while** **(**1**)**

**{**

cout **<<** " what do you want to edit?" **<<** endl

**<<** "1. Call date" **<<** endl

**<<** "2. Phone number" **<<** endl

**<<** "3. Call's duration" **<<** endl

**<<** "4. Discount" **<<** endl

**<<** "5. End edit" **<<** endl**;**

cin **>>** key**;**

**switch** **(**key**)**

**{**

**case** '1'**:**

cout **<<** "call date : "**;**

cin **>>** c**;**

calldate **=** c**;**

**break;**

**case** '2'**:**

cout **<<** "Phone number: "**;**

cin **>>** p**;**

strcpy\_s**(**phonenumber**,**p**);**

**break;**

**case** '3'**:**

cout **<<** "Call's duration: "**;**

cin **>>** num**;**

callduration **=** num**;**

**break;**

**case** '4'**:**

cout **<<** "Discount"**;**

cin **>>** d**;**

discount **=** d**;**

**case** '5'**:**

**return;**

**break;**

**}**

**}**//while

**}**//func

int get\_callid**()** const

**{**

**return** call\_id**;**

**}**

string get\_city**()** const

**{**

**return** city**;**

**}**

**};**

class arr

**{**

vector **<**PhoneCall**>** call**;**

public**:**

void add**(**PhoneCall zvonok**)**

**{**

call**.**push\_back**(**zvonok**);**

**}**

void show**()**

**{**

cout **<<** "Phone call data" **<<** endl**;**

cout **<<** setw**(**4**)** **<<** "call\_id" **<<** setw**(**12**)** **<<** "call date" **<<** setw**(**10**)** **<<** "city" **<<** setw**(**15**)<<**"city\_id" **<<** setw**(**10**)** **<<** " price per min " **<<** setw**(**12**)** **<<** " duration " **<<** setw**(**15**)** **<<** "call beginning" **<<** setw**(**15**)** **<<** "invoice date" **<<** setw**(**8**)** **<<** "charge" **<<** setw**(**10**)** **<<** "discount" **<<** setw**(**8**)** **<<** "total" **<<** endl**;**

cout **<<** "---------------------------------------------------------------------------------------------------------------------" **<<** endl**;**

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

**{**

cout **<<** endl**;**

call**[**i**].**show**();**

**}**

**}**

void showcity**()**

**{**

cout **<<** "Here's some information about the city you entered" **<<** endl**;**

cout **<<** setw**(**4**)** **<<** "city" **<<** setw**(**10**)** **<<** "city\_id" **<<** endl**;**

cout **<<** "-----------------------------------------" **<<** endl**;**

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

**{**

cout **<<** endl**;**

call**[**i**].**display**();**

cout **<<** endl**;**

**}**

**}**

void erase**(**int id\_**)**

**{**

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

**{**

**if** **(**call**[**i**].**get\_callid**()** **==** id\_**)**

**{**

call**.**erase**(**call**.**begin**()** **+** i**);**

**return;**

**}**

**}**

cout **<<** "No id" **<<** endl**;**

**}**

void search**(**int id\_**)**

**{**

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

**{**

**if** **(**call**[**i**].**get\_callid**()** **==** id\_**)**

**{**

call**[**i**].**show**();**

**return;**

**}**

**}**

cout **<<** "No ID" **<<** endl**;**

**}**

void Findcity**(**string gorod**)**

**{**

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

**if** **(**call**[**i**].**get\_city**()** **==** gorod**)**

**{**

call**[**i**].**display**();**

**return;**

cout **<<** endl**;**

**}**

cout **<<** "Incorrect city" **<<** endl**;**

**}**

void find**(**int id\_**)**

**{**

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

**{**

**if** **(**call**[**i**].**get\_callid**()** **==** id\_**)**

**{**

call**[**i**].**edit**();**

call**[**i**].**Charge**();**

call**[**i**].**Total**();**

**return;**

**}**

**}**

cout **<<** "No ID" **<<** endl**;**

**}**

void to\_file**()**

**{**

char file**[**64**]** **=** "data.txt"**;**

ofstream out**(**file**,** ios**::**binary**);**

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

**{**

out**.**write**((**char**\*)&**call**[**i**],** **sizeof(**call**[**i**]));**

**}**

cout **<<** "Saved" **<<** endl**;**

**}**

void from\_file**()**

**{**

char file**[**64**]** **=** "data.txt"**;**

char key**;**

PhoneCall zvonok**;**

ifstream in**(**file**,** ios**::**binary**);**

cout **<<** "Clear array? y/n "**;**

cin **>>** key**;**

**if** **(**key **==** 'y'**)**

**{**

call**.**clear**();**

**}**

**while** **(!**in**.**eof**())**

**{**

in**.**read**((**char**\*)&**zvonok**,** **sizeof(**zvonok**));**

call**.**push\_back**(**zvonok**);**

**}**

call**.**erase**(**call**.**end**()** **-** 1**);**

cout **<<** "Loaded" **<<** endl**;**

**}**

void Sort**()**

**{**

char key**;**

cout **<<** "How sort?" **<<** endl

**<<** "1. bubble" **<<** endl**;**

cin **>>** key**;**

**if** **(**key **==** '1'**)**

**{**

PhoneCall temp**;** // временная переменная для хранения элемента массива

bool exit **=** **false;** // болевая переменная для выхода из цикла, если массив отсортирован

**while** **(!**exit**)** // пока массив не отсортирован

**{**

exit **=** **true;**

**for** **(**int int\_counter **=** 0**;** int\_counter **<** **(**call**.**size**()** **-** 1**);** int\_counter**++)** // внутренний цикл

//сортировка пузырьком по возрастанию - знак >

//сортировка пузырьком по убыванию - знак <

**if** **(**call**[**int\_counter**].**get\_callid**()** **>** call**[**int\_counter **+** 1**].**get\_callid**())** // сравниваем два соседних элемента

**{**

// выполняем перестановку элементов массива

swap**(**call**[**int\_counter**],** call**[**int\_counter **+** 1**]);**

exit **=** **false;** // на очередной итерации была произведена перестановка элементов

**}**

**}**

**}**// if

**}**// Sort

**};**

int main**(**int argc**,** char**\*** argv**[])**

**{**

setlocale**(**LC\_ALL**,** "rus"**);**

char key**;**

arr mas**;**

PhoneCall zvonok**;**

int id\_**;**

string gorod**;**

**while** **(**1**)**

**{**

cout **<<** endl

**<<** "Choose action:"

**<<** endl

**<<** "N New phone call." **<<** endl

**<<** "E Edit data." **<<** endl

**<<** "F Find phone call." **<<** endl

**<<** "G Get city\_id" **<<** endl

**<<** "D Delete data." **<<** endl

**<<** "S Save to file." **<<** endl

**<<** "R Load from file." **<<** endl

**<<** "V Show all phone calls." **<<** endl

**<<** "C Sort data." **<<** endl

**<<** "Q Quit." **<<** endl**;**

cin **>>** key**;**

**switch** **(**key**)**

**{**

**case** 'n'**:**

zvonok**.**set\_data**();**

zvonok**.**Charge**();**

zvonok**.**Total**();**

mas**.**add**(**zvonok**);**

**break;**

**case** 'e'**:**

cout **<<** endl **<<** "Enter call\_id: "**;**

cin **>>** id\_**;**

mas**.**find**(**id\_**);**

**break;**

**case** 'f'**:**

cout **<<** endl **<<** "Enter call\_id: "**;**

cin **>>** id\_**;**

mas**.**search**(**id\_**);**

**break;**

**case** 'g'**:**

cout **<<** endl **<<** " Enter the city:"**;**

cin **>>** gorod**;**

mas**.**Findcity**(**gorod**);**

**break;**

**case** 'd'**:**

cout **<<** endl **<<** "Enter call\_id: "**;**

cin **>>** id\_**;**

mas**.**erase**(**id\_**);**

**break;**

**case** 's'**:**

mas**.**to\_file**();**

**break;**

**case** 'r'**:**

mas**.**from\_file**();**

**break;**

**case** 'v'**:**

mas**.**show**();**

**break;**

**case** 'c'**:**

mas**.**Sort**();**

**break;**

**case** 'q'**:**

exit**(**1**);**

**break;**

**}**

cout **<<** endl**;**

**}**

system**(**"pause"**);**

**return** 0**;**

**}**

// Запуск программы: CTRL+F5 или меню "Отладка" > "Запуск без отладки"

// Отладка программы: F5 или меню "Отладка" > "Запустить отладку"

// Советы по началу работы

// 1. В окне обозревателя решений можно добавлять файлы и управлять ими.

// 2. В окне Team Explorer можно подключиться к системе управления версиями.

// 3. В окне "Выходные данные" можно просматривать выходные данные сборки и другие сообщения.

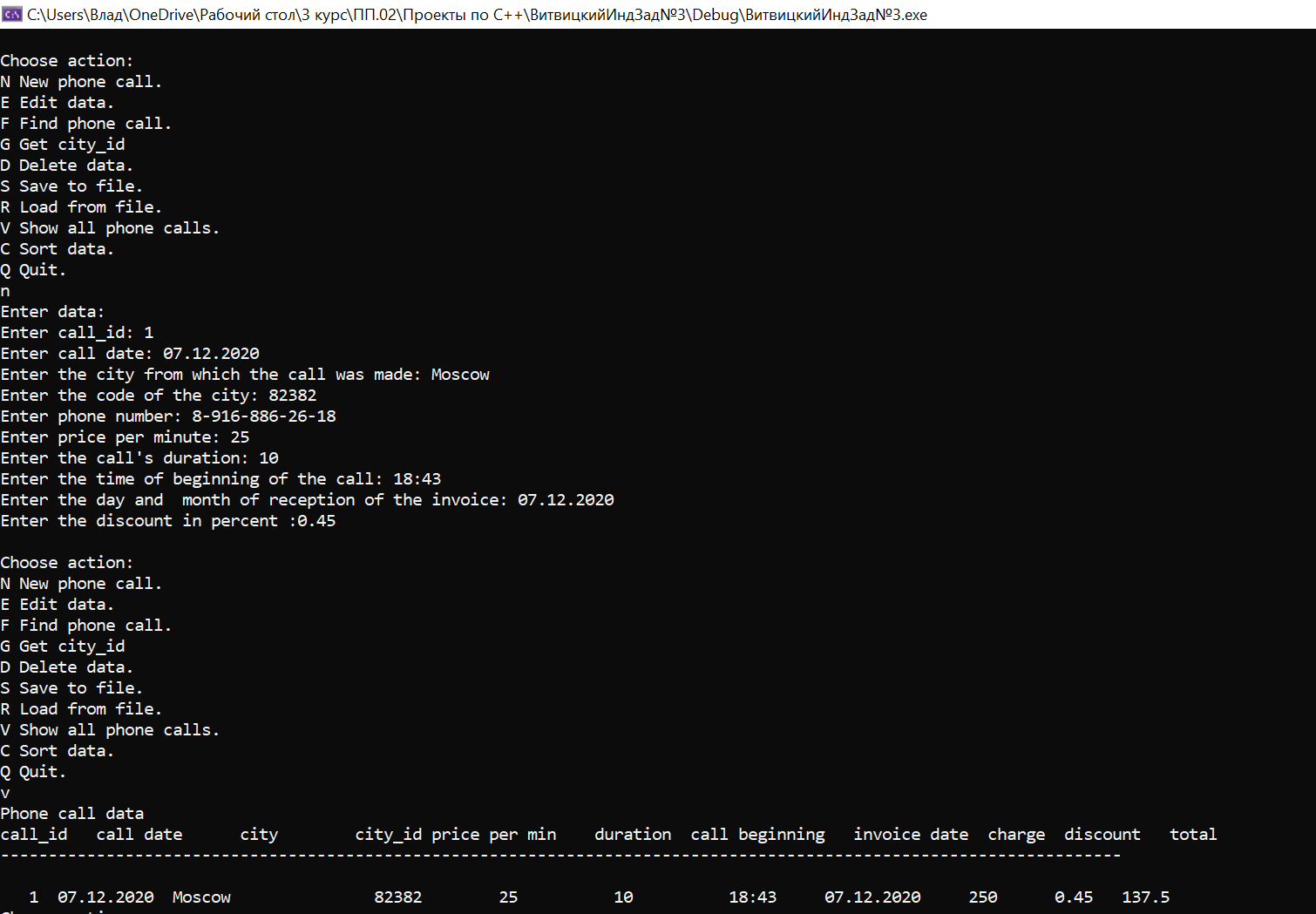
// 4. В окне "Список ошибок" можно просматривать ошибки.

// 5. Последовательно выберите пункты меню "Проект" > "Добавить новый элемент", чтобы создать файлы кода, или "Проект" > "Добавить существующий элемент", чтобы добавить в проект существующие файлы кода.

// 6. Чтобы снова открыть этот проект позже, выберите пункты меню "Файл" > "Открыть" > "Проект" и выберите SLN-файл.

Работа программы

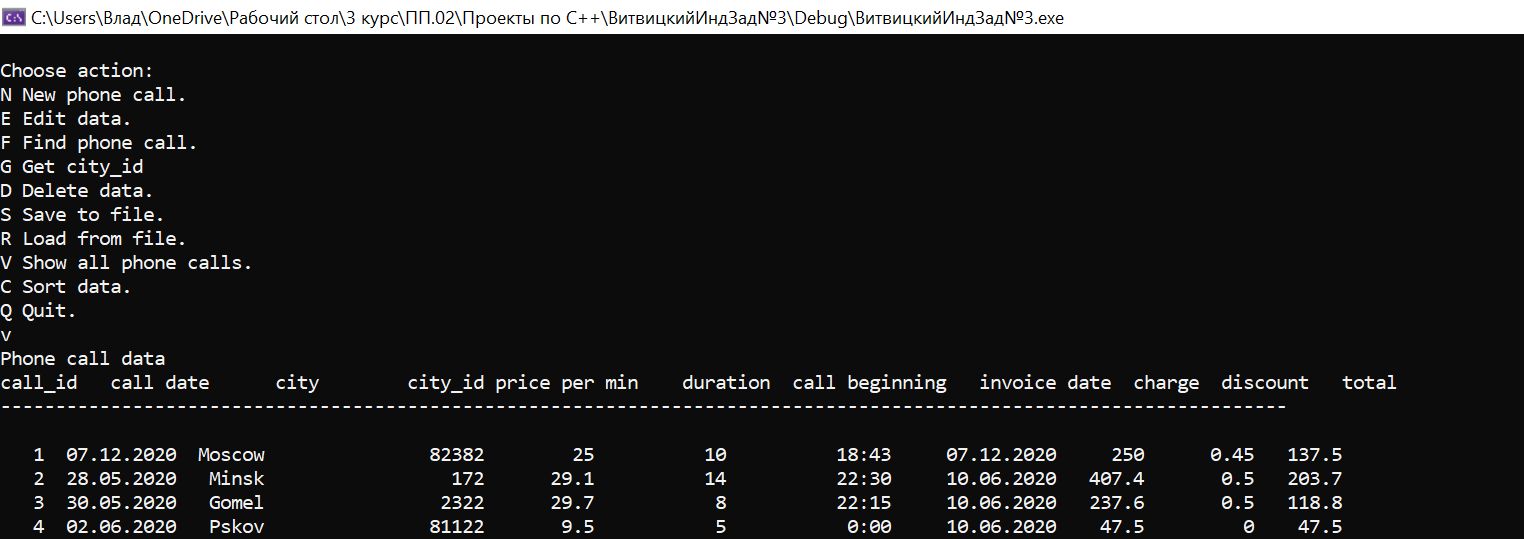
Для начала введем одну строку таблицы для теста.



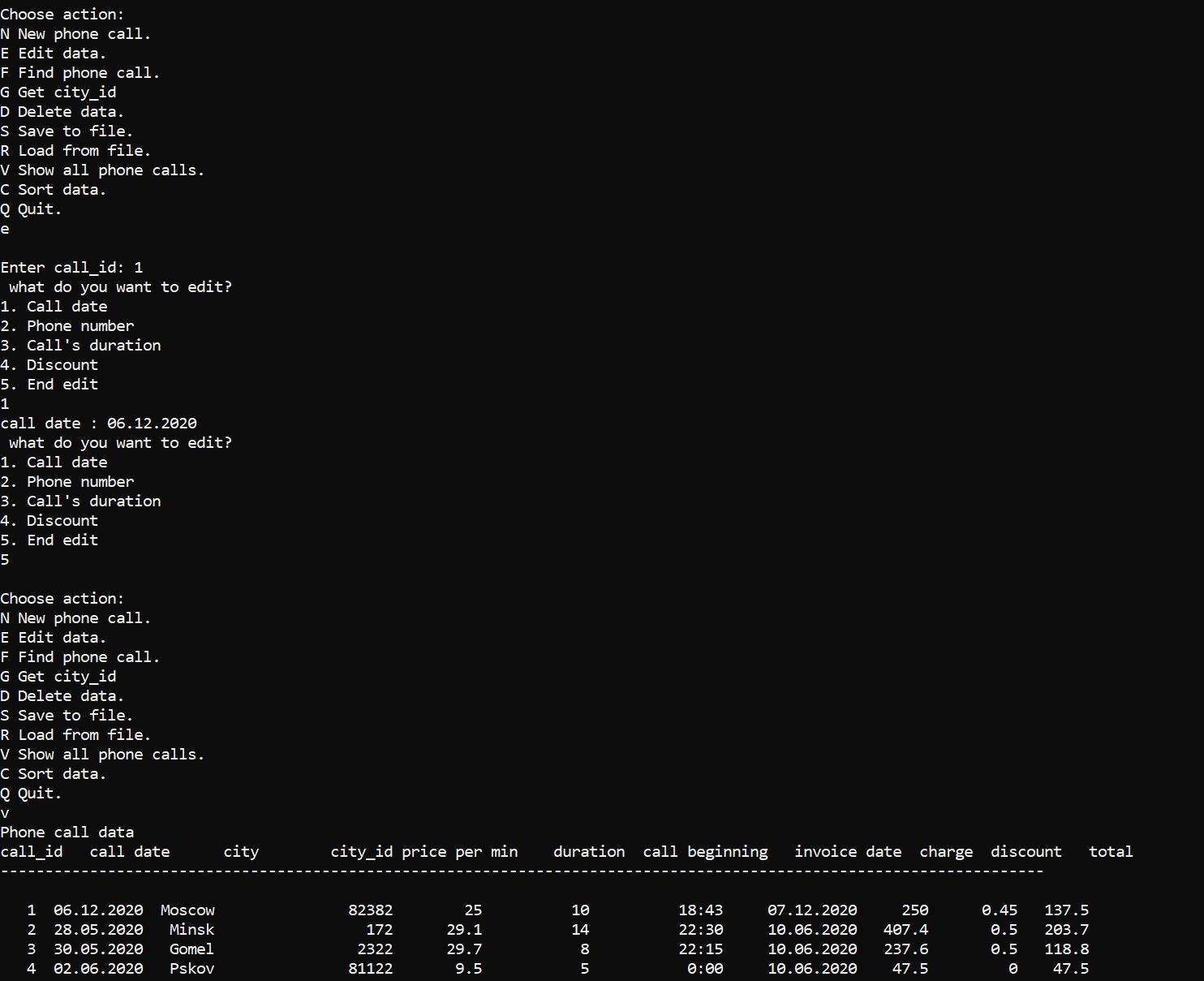
Теперь введем пару строк из таблицы:



Получим такую таблицу при выводе данных:

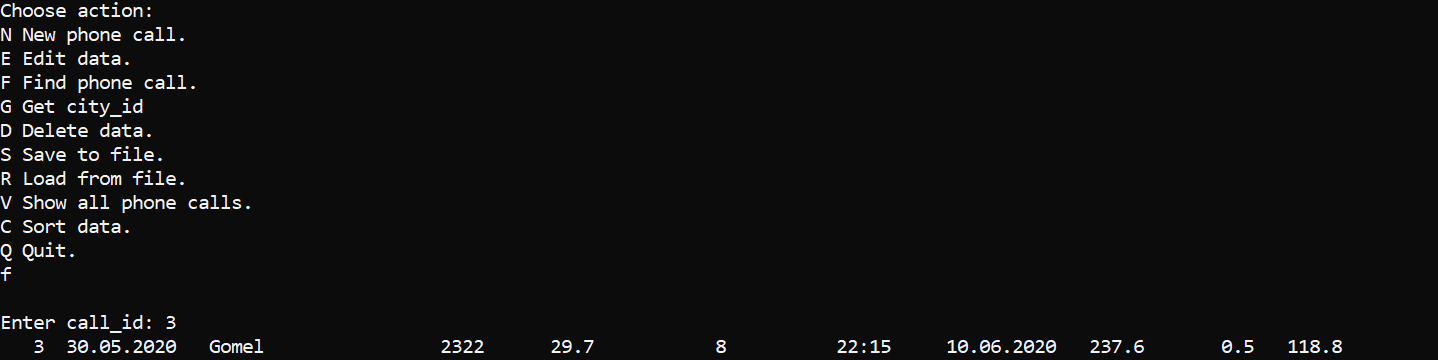


Теперь постараемся изменить дату звонка в первой строке:

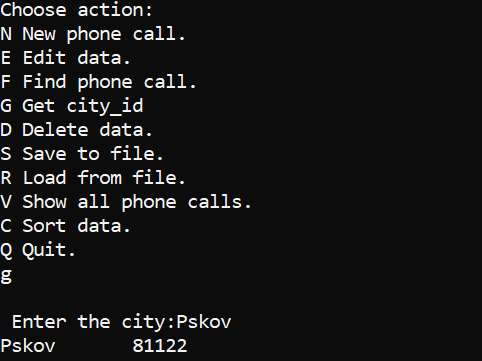


Теперь опробуем другие функции:

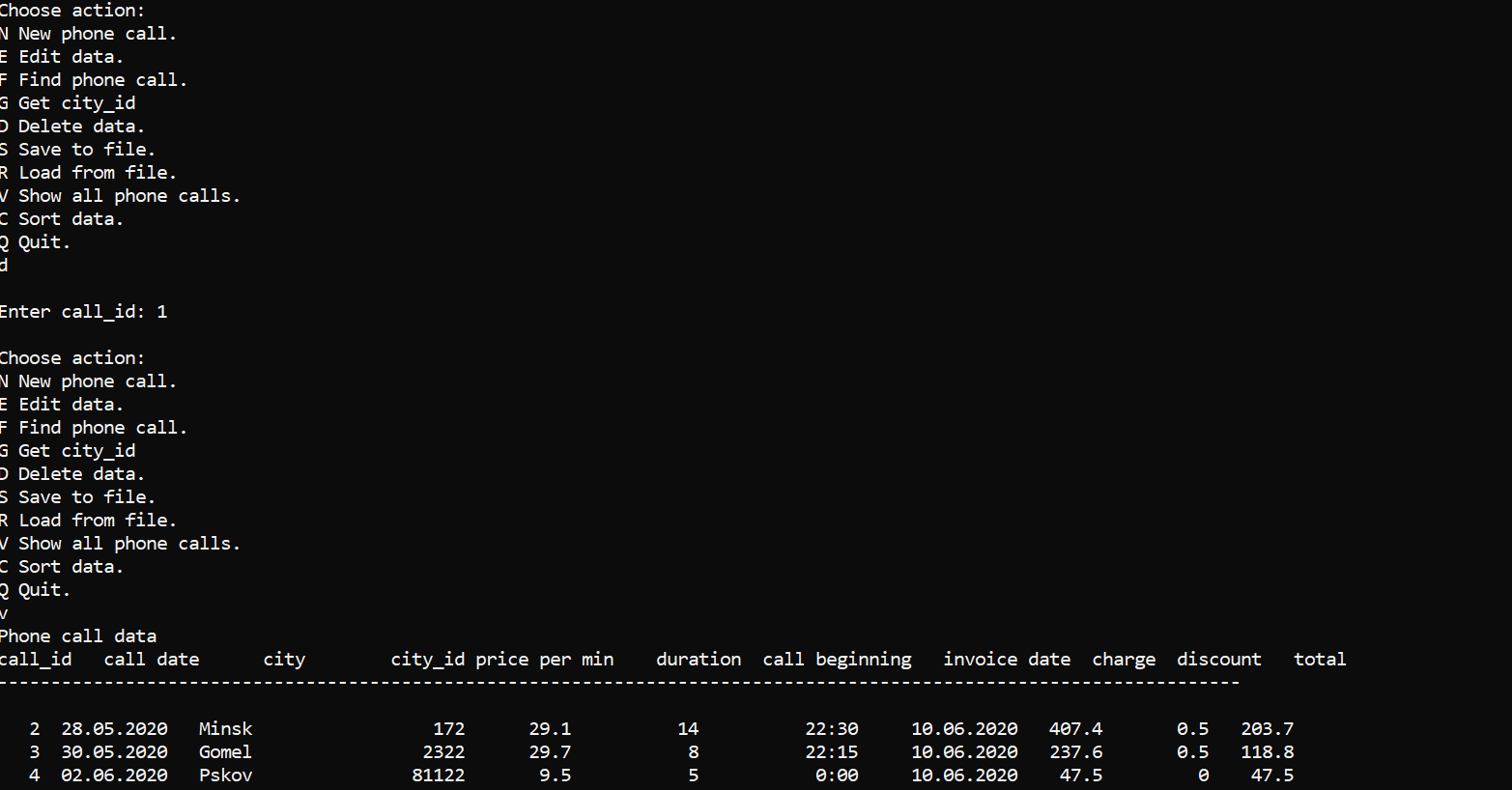
Например, найдем информацию о звонке по его id:



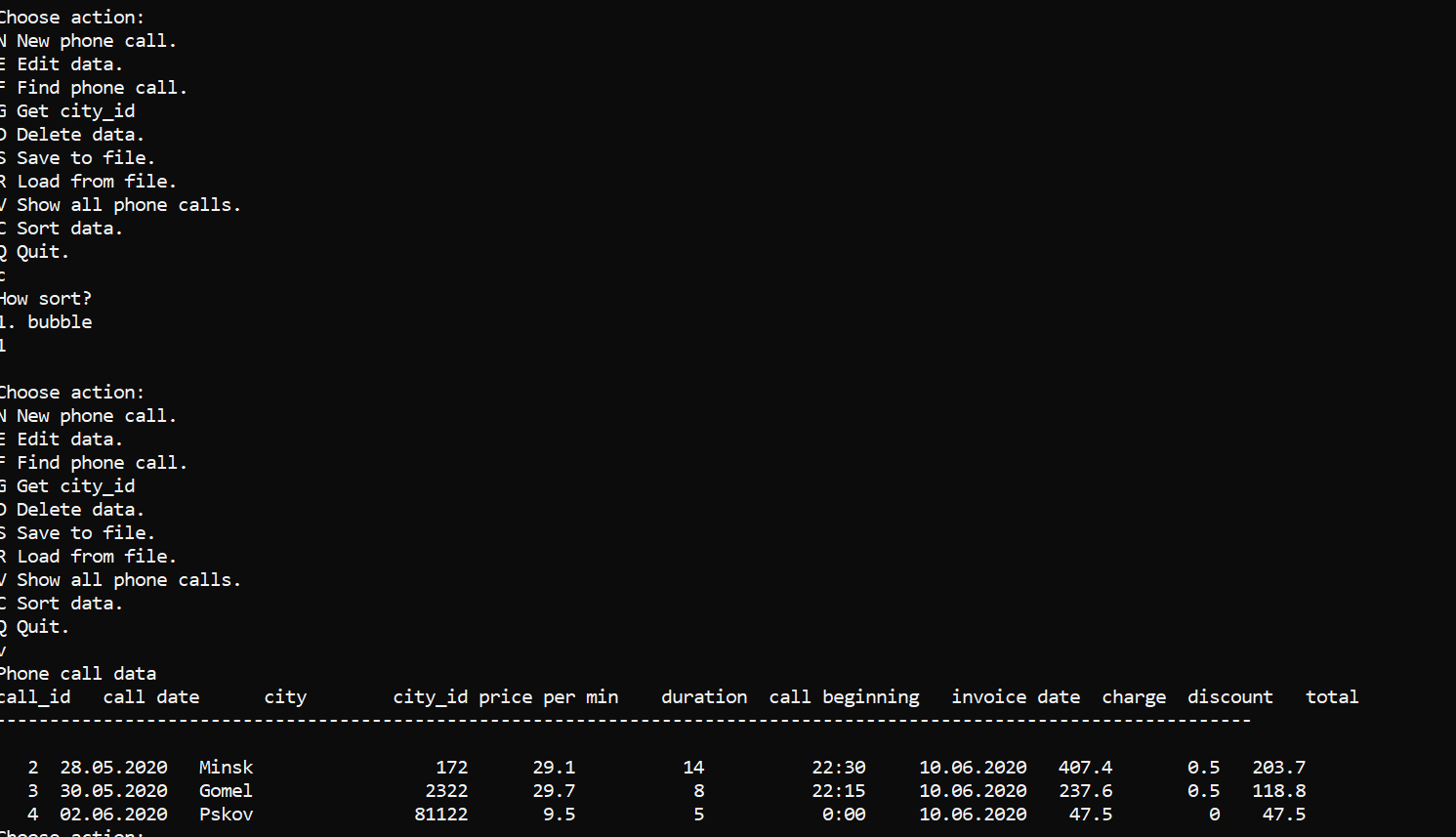
Или найдем код города Псков:



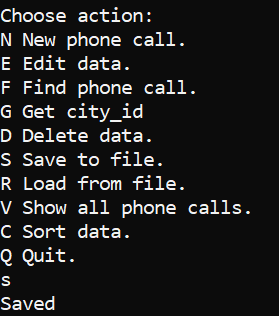
Или удалим строку:



Теперь рассмотрим сортировку данных:



Выгрузку данных в файл:



Если зайти в папку с файлами проекта, там будет лежать следующий файл:

