Министерство образование Республики Беларусь

Учреждение образования «Полоцкий государственный университет»

Информационно-технологический факультет

Кафедра технологий программирования

Лабораторная работа №4

По дисциплина «Основы алгоритмизации и программирования»

Выполнил: Крутилин С.Д.

Группа 19-ИТ-1.

Проверил: Адамовский Е.

Новополоцк

2019 г.

**Структуры и перечисления**

**Вариант 2**

**Цель работы**: научиться работать с векторными данными языка С++: структурами и перечислениями. Изучить правила объявления структур и перечислений, обращение к их полям, построение вложенных векторных типов данных.

**ЗАДАНИЕ 1.** Создать структуру для описания предметной области вашей курсовой работы.

**Описание:** структура содержит контактную информацию(ФИО,ГОРОД,УЛИЦА,ДОМ, КВАРТИРА,НОМЕР ТЕЛЕФОНА).

struct Contact {

char FIO[40];

char city[20];

char street[20];

int house;

int flat;

long int phone;

};

**ЗАДАНИЕ 2.** В функции main() создать указатель на структуру из задания 1. Выделить динамическую память под эту структуру.

**Описание:** создаем указатель на структуру из задания 1 и выделяем динамическую память под эту структуру. Делаем проверка ну выделение памяти.

int main()

{

Contact\* buff = (Contact\*)malloc(sizeof(Contact));

if (buff == NULL) {

cout << "Не удалось выделить память!" << endl;

exit(-1);

}

}

**ЗАДАНИЕ 3.** Написать функцию, которая будет считывать данные с клавиатуры и помещать их в структуру. Выделить дополнительную динамическую память в указателе на структуру под новую запись.

**Кратко:** внутри функции input() создается временная структура, которая хранит контактную информацию; затем заполняем ее каждое поле и для каждого делаем соответствующую проверку на корректность; выделяем дополнительную память для основного массива структур; присваиваем в выделенную память нашу временную структуру.

**Выделение дополнительной динамической памяти:**

void resize() {

Contact\* buff = (Contact\*)realloc(information, sizeof(Contact) \* kol);

if (buff == NULL) {

cout << "Не удалось выделить память!" << endl;

exit(-1);

}

information = buff;

}

**Считывание данных с клавиатуры:**

void input() {

int check = 0;

Contact person;

while (check != 1) {

system("cls");

cout << "Введите ФИО: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(1);

}

}

strcpy\_s(person.FIO, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите город проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(2);

}

}

strcpy\_s(person.city, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите улицу проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(3);

}

}

strcpy\_s(person.street, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер дома проживания: ";

gets\_s(Str);

check = check\_input(4);

}

person.house = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер квартиры проживания: ";

gets\_s(Str);

check = check\_input(5);

}

person.flat = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер телефона: ";

gets\_s(Str);

check = check\_input(6);

if (check != -1) {

check = check\_input(0);

}

}

strcpy\_s(person.phone, Str);

check = 0;

kol++;

resize();

information[kol - 1] = person;

}

**Проверка на корректность ввода:** fied == 0 – проверка ввода длины номера телефона; field == 1 – проверка ввода длины ФИО; field == 2 – проверка ввода длины названия города; field == 3 – проверка ввода длины названия улицы; field == 4 – проверка на корректность ввода номера дома; field == 5 – проверка на корректность ввода номера квартиры; field == 6 – проверка корректности ввода номера телефона; field == 8 – проверка корректности ввода строк;

int check\_input(int field) {

int n = strlen(Str);

int pr = 0;

for (int i = 0; i < n; i++) {

if (Str[i] == ' ') pr++;

}

if (pr == n) {

return -1;

}

if (field == 0) {

if (n != 12) {

return -1;

}

else return 1;

}

if (field == 1) {

if (Str[0] == 0 || n > 40) {

return -1;

}

else return 1;

}

if (field == 2) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 3) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 4) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 5) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 6) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if (Str[i] < '0' || Str[i] > '9')

{

return -1;

}

}

return 1;

}

if (field == 8) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if ((Str[i] < 'A' || Str[i] > 'Z') && (Str[i] < 'a' || Str[i] > 'z') && (Str[i] < 'А' || Str[i] > 'я') && Str[i] != ' ')

{

return -1;

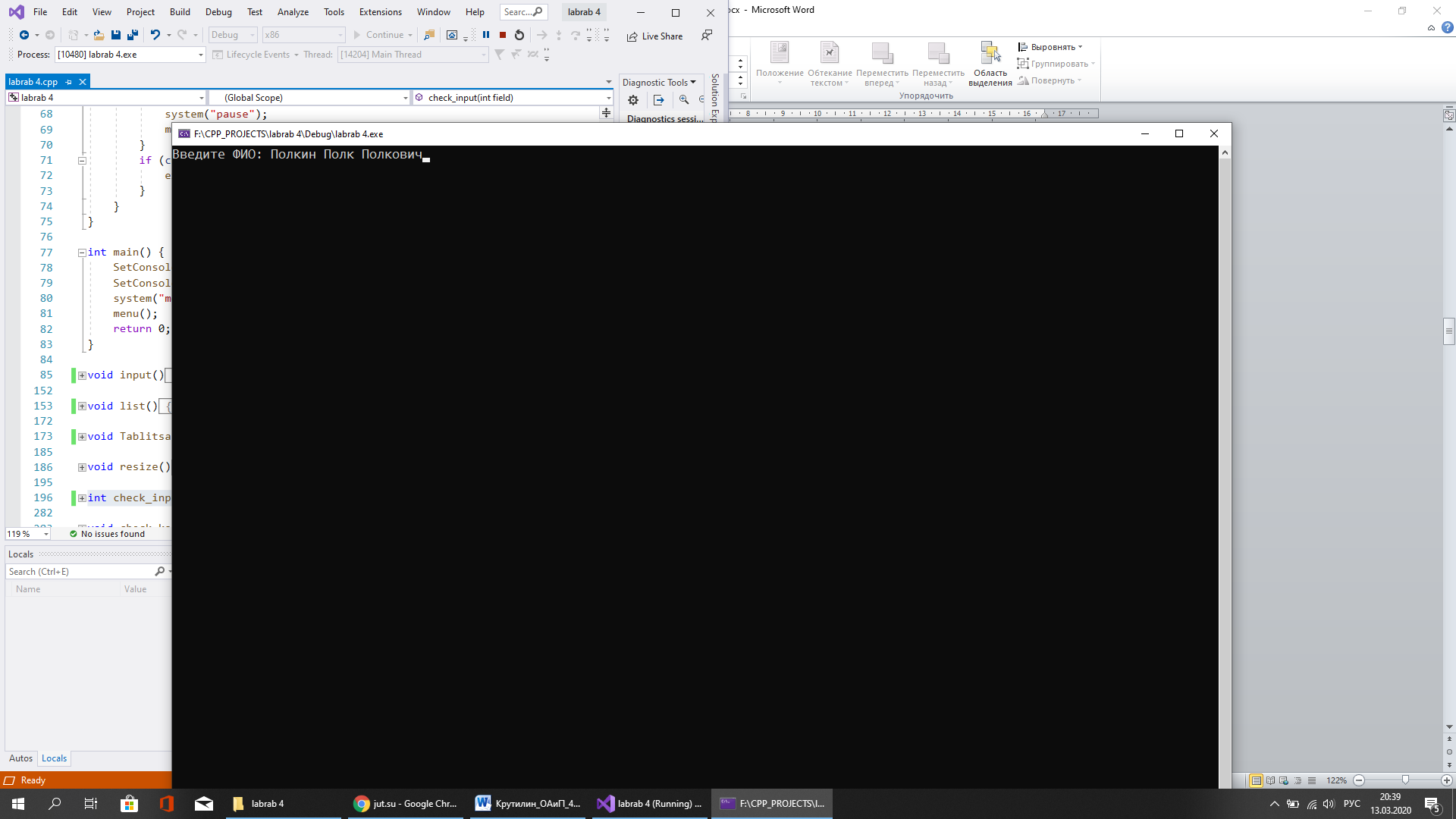
}

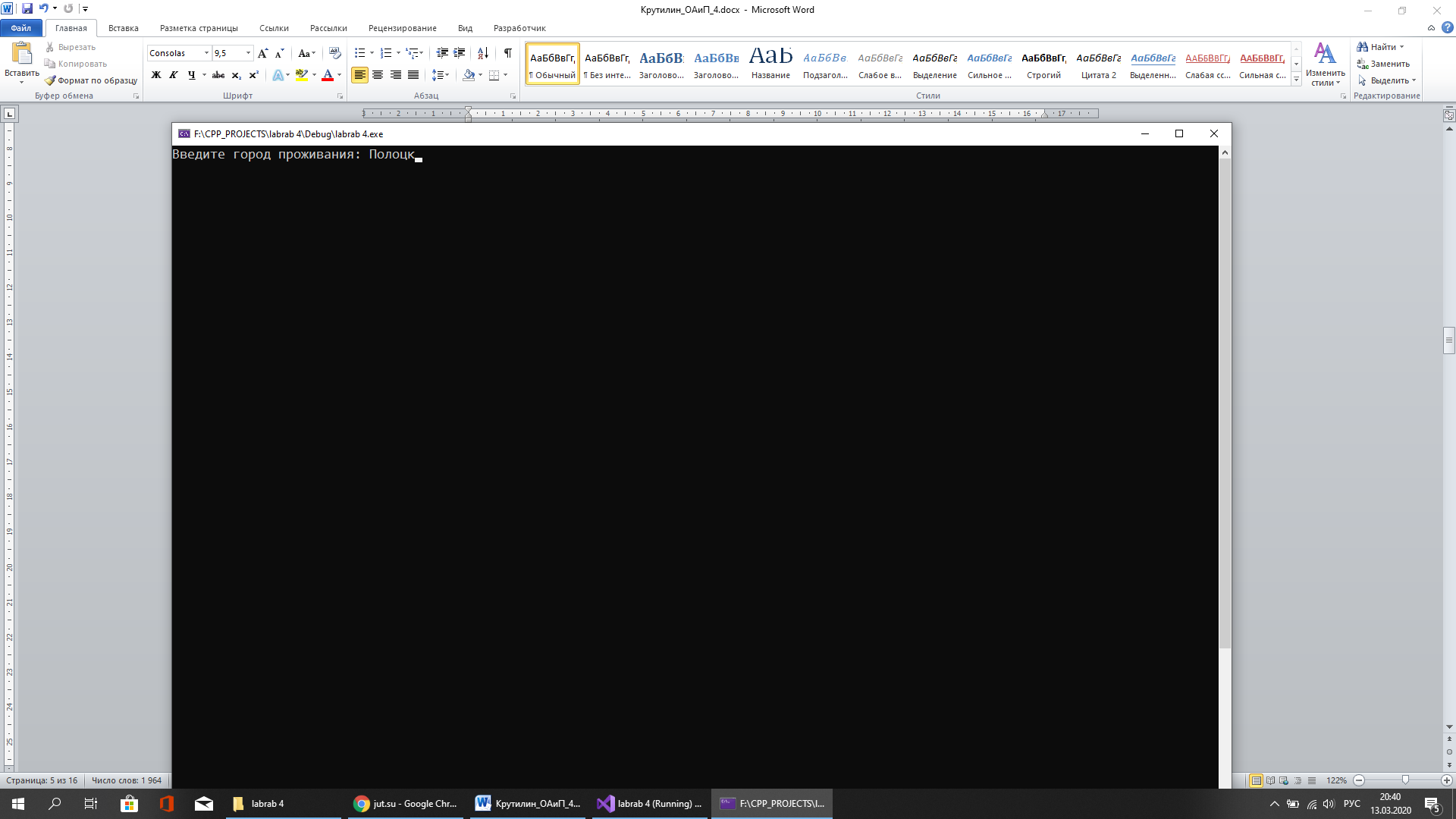
}

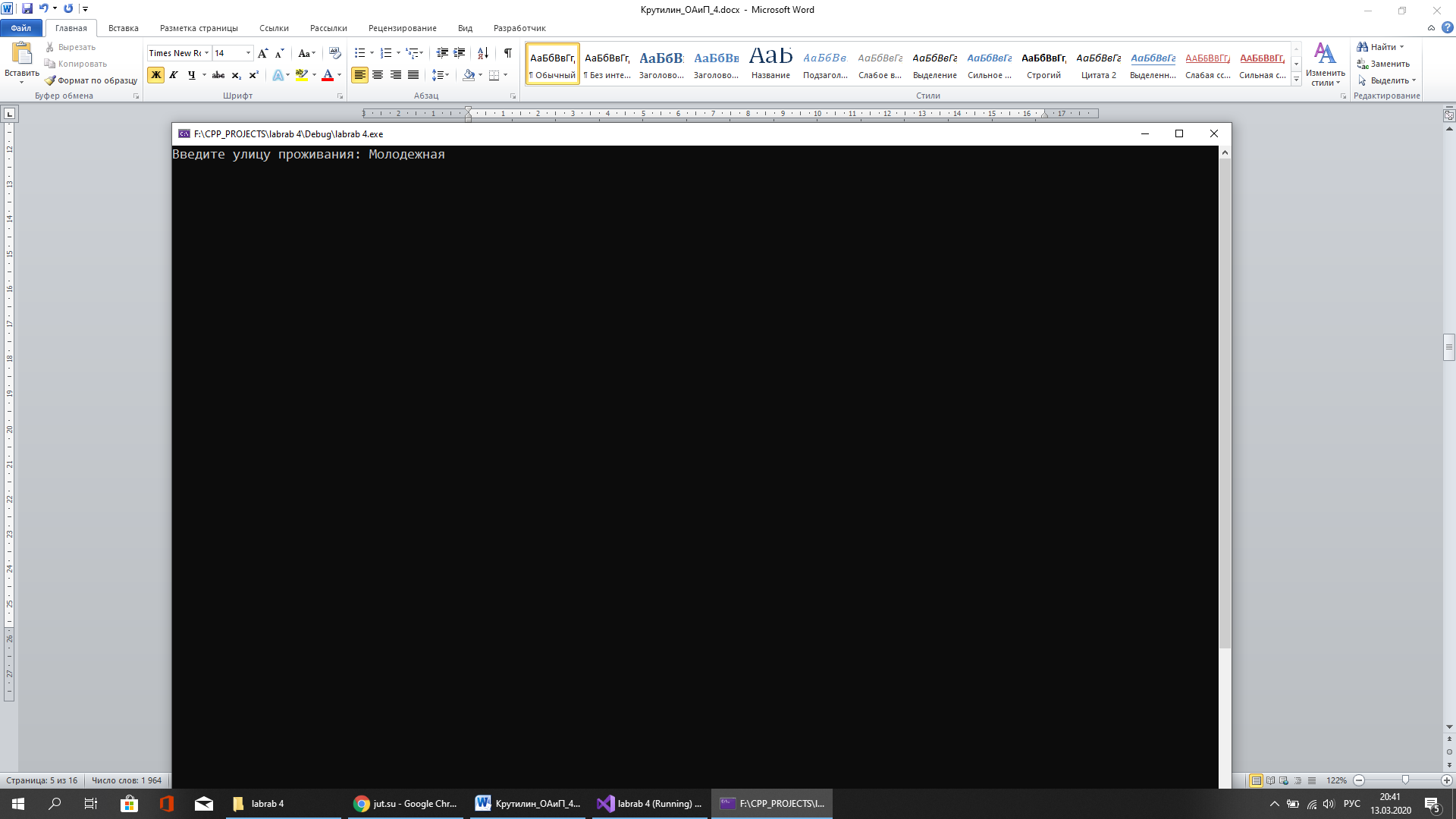
return 1;

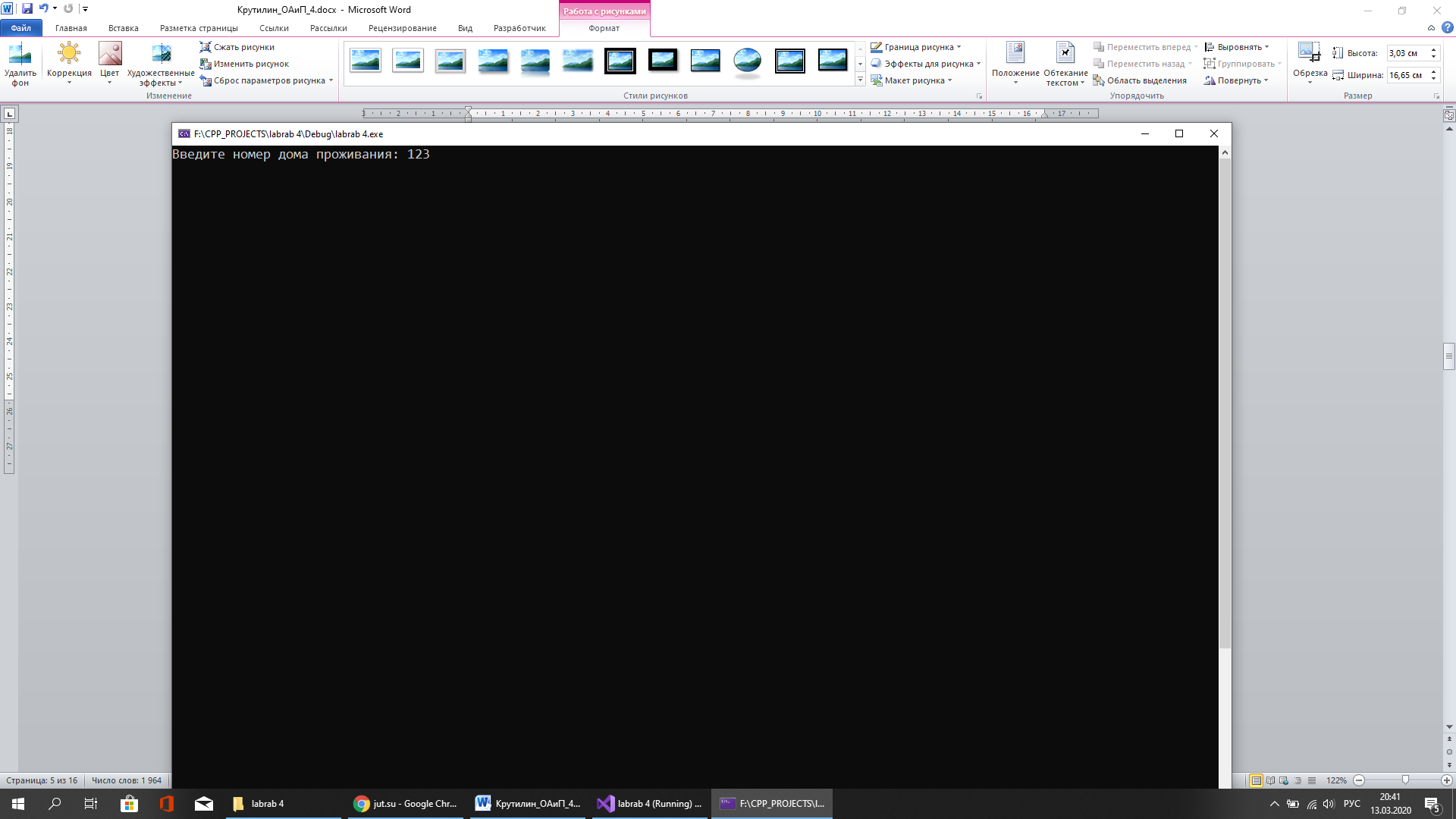
}

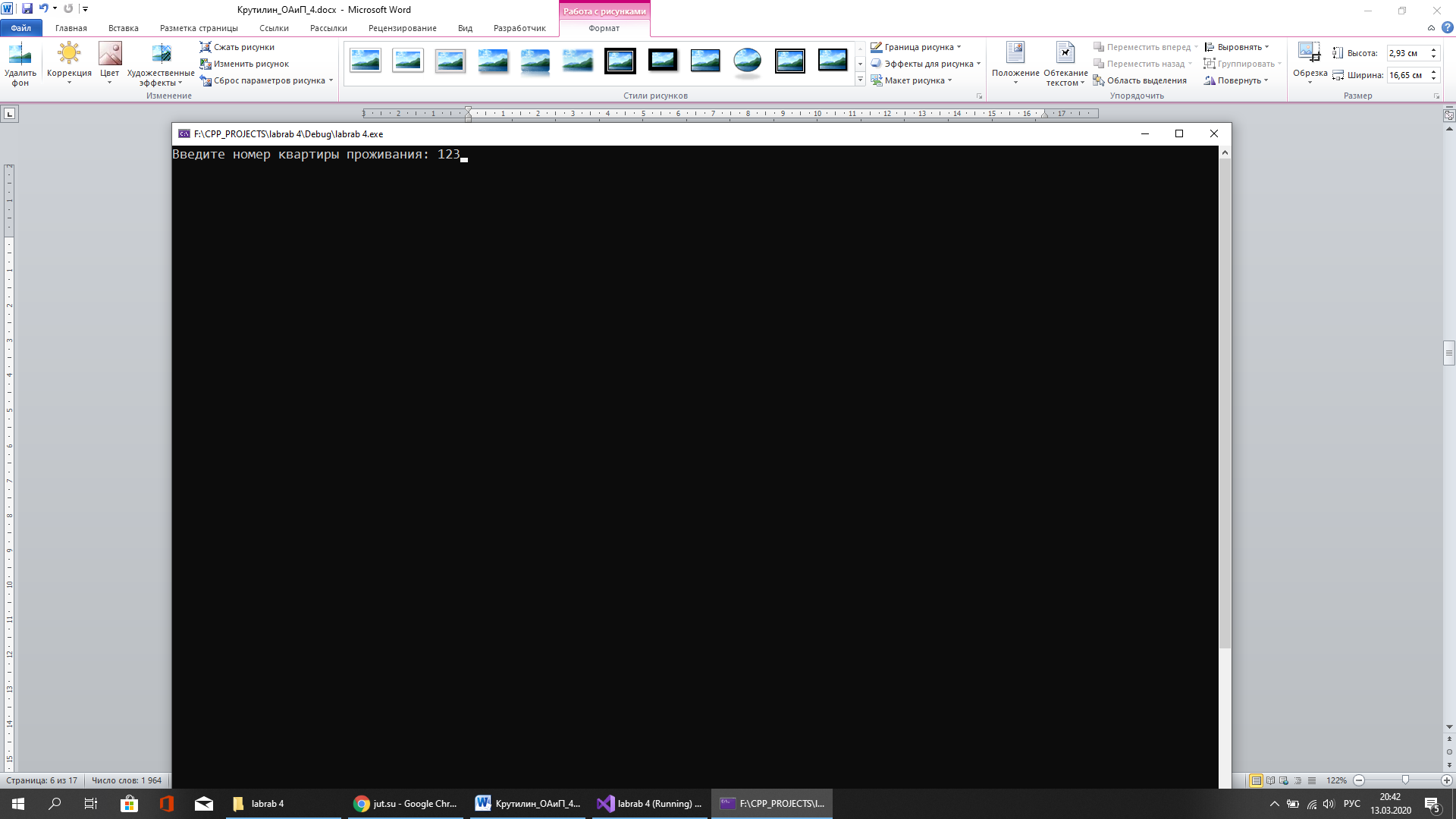
}

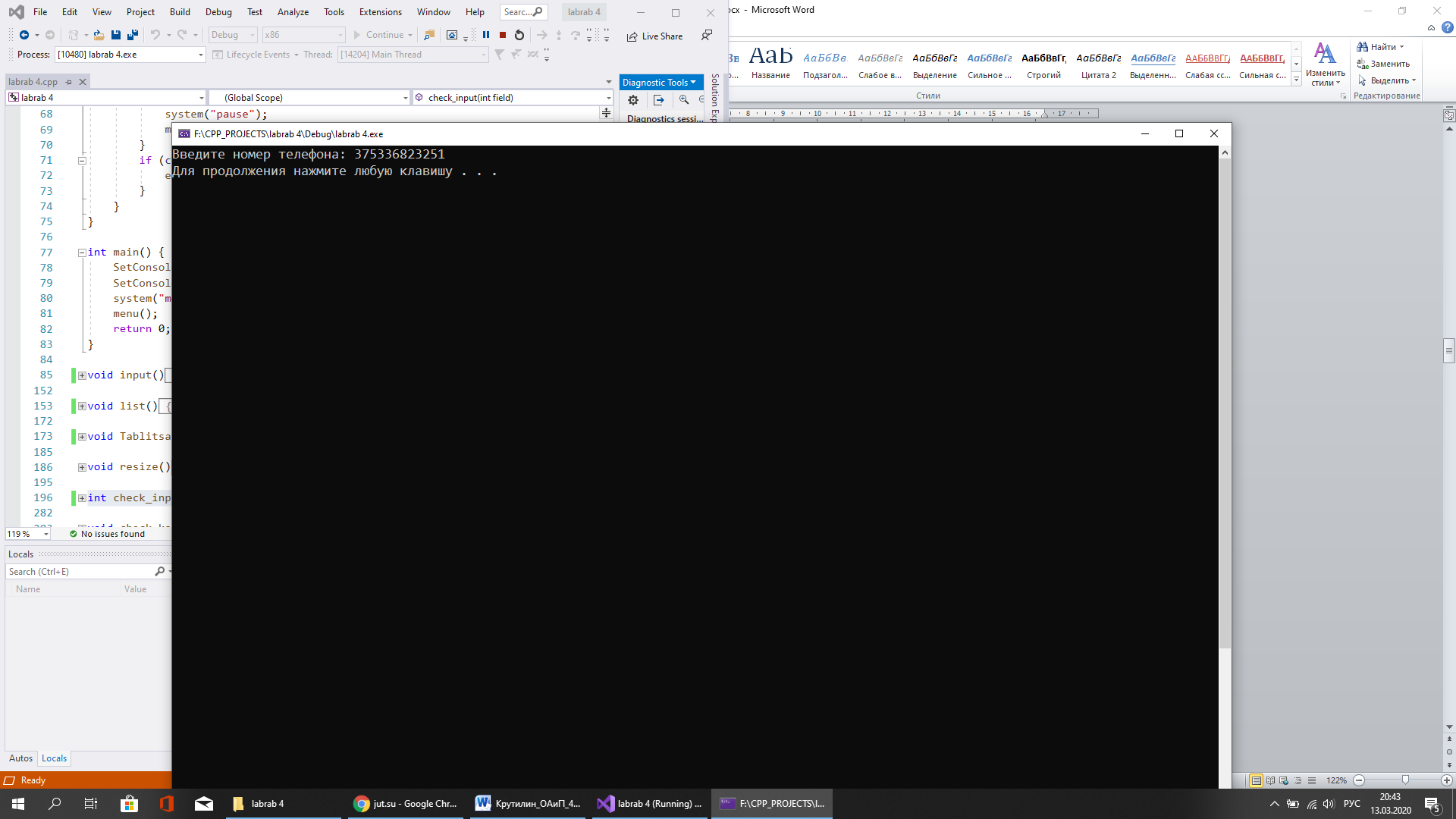








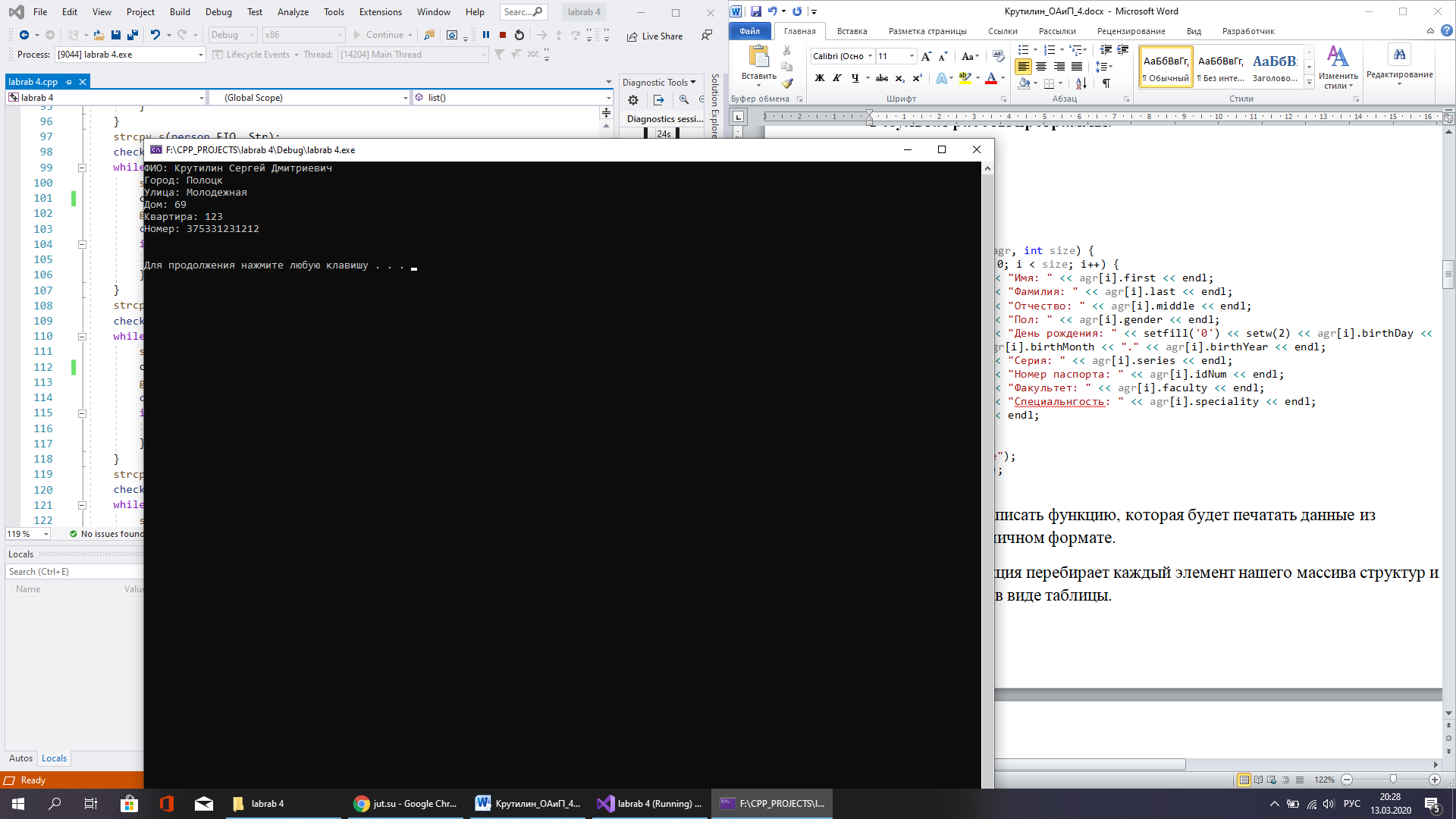




**ЗАДАНИЕ 4.** Написать функцию, которая будет печатать данные из структуры в формате "поле: значение\n".

**Описание:** выводит данные в формате «поле: значение\n». В цикле перебирается каждая структура.

**Результат работы программы:**



void list() {

if (kol == 0) {

cout << "Данные отсутствуют.";

}

else {

for (int i = 0; i < kol; i++) {

cout << "ФИО: " << (\*information).FIO << endl;

cout << "Город: " << (\*information).city << endl;

cout << "Улица: " << (\*information).street << endl;

cout << "Дом: " << (\*information).house << endl;

cout << "Квартира: " << (\*information).flat << endl;

cout << "Номер: " << (\*information).phone << endl;

cout << endl;

}

}

cout << endl;

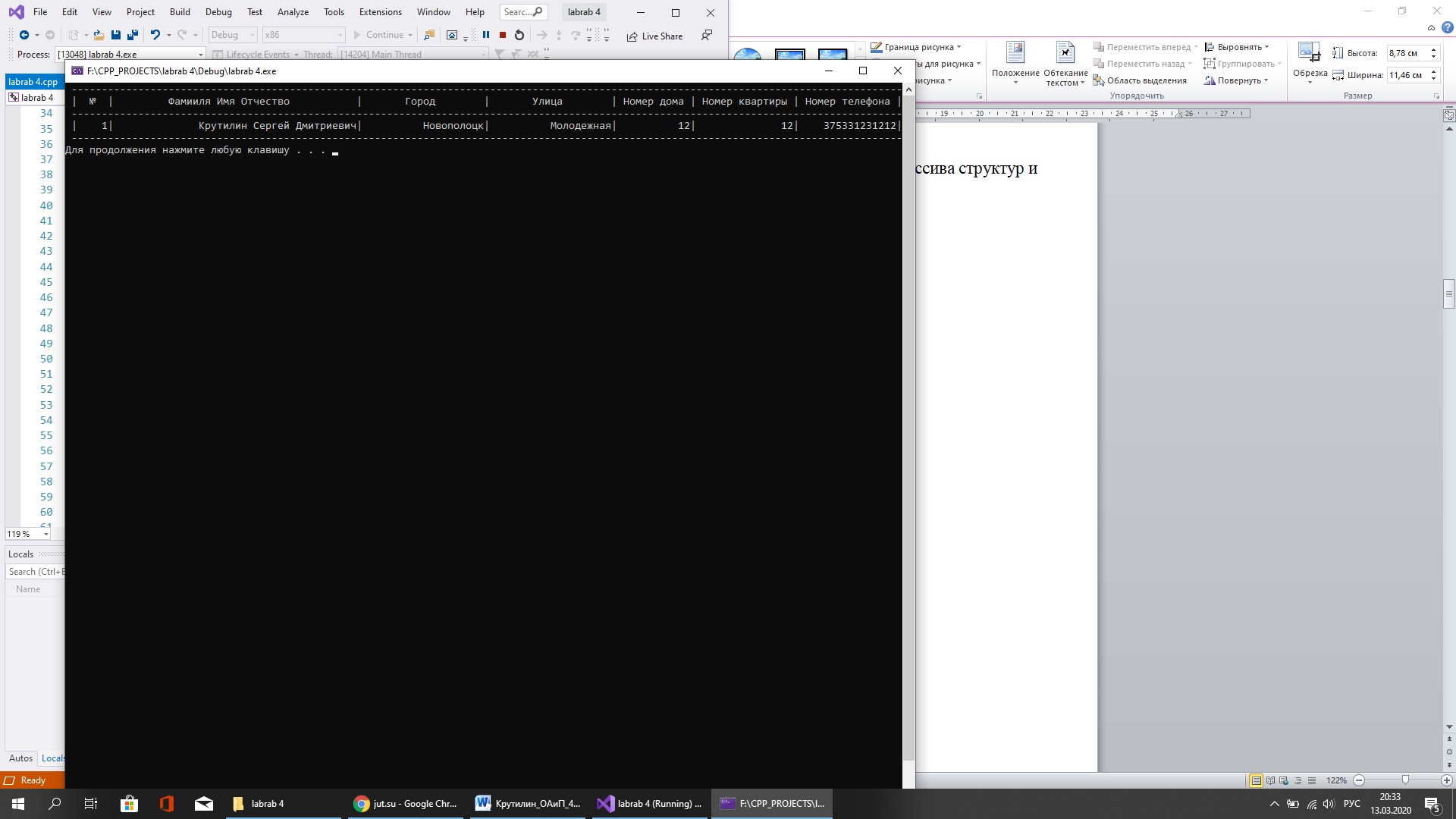
system("pause");

system("cls");

}

**ЗАДАНИЕ 5.** Написать функцию, которая будет печатать данные из структуры в табличном формате.

**Алгоритм:** функция перебирает каждый элемент нашего массива структур и выводит данные в виде таблицы.



**Табличный вывод:**

void Tablitsa()

{

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

printf(" | № | Фамииля Имя Отчество | Город | Улица | Номер дома | Номер квартиры | Номер телефона |\n");

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

for (int i = 0; i < kol; i++)

{

printf(" |%5d|%40s|%20s|%20s|%12d|%16d|%16s|\n", i + 1,

information[i].FIO, information[i].city, information[i].street, information[i].house, information[i].flat, information[i].phone);

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

}

}

**ЗАДАНИЕ \*.** Организовать меню для вызова функций из заданий 3-5

**Описание:** по итогу программа была разделена, а в функции было реализовано меню с вызовом необходимых функций. Меню представляет из себя бесконечный цикл, из которого можно выйти так же с помощью меню. Данные будут вводиться до тех пор, пока пользователь не выберет действие из меню. Затем с помощью if вызывается нужная функция.

void show\_menu() {

system("cls");

cout << " MENU \n" << endl;

cout << "1 - Ввод данных" << endl;

cout << "2 - Просмотр таблицы" << endl;

cout << "3 - Список" << endl;

cout << "0 - Выход" << endl;

}

void menu() {

char ch;

show\_menu();

while (1) {

ch = \_getch();

if (ch == '1') {

system("cls");

input();

system("pause");

menu();

}

if (ch == '2') {

system("cls");

check\_kol();

Tablitsa();

system("pause");

menu();

}

if (ch == '3') {

system("cls");

check\_kol();

list();

system("pause");

menu();

}

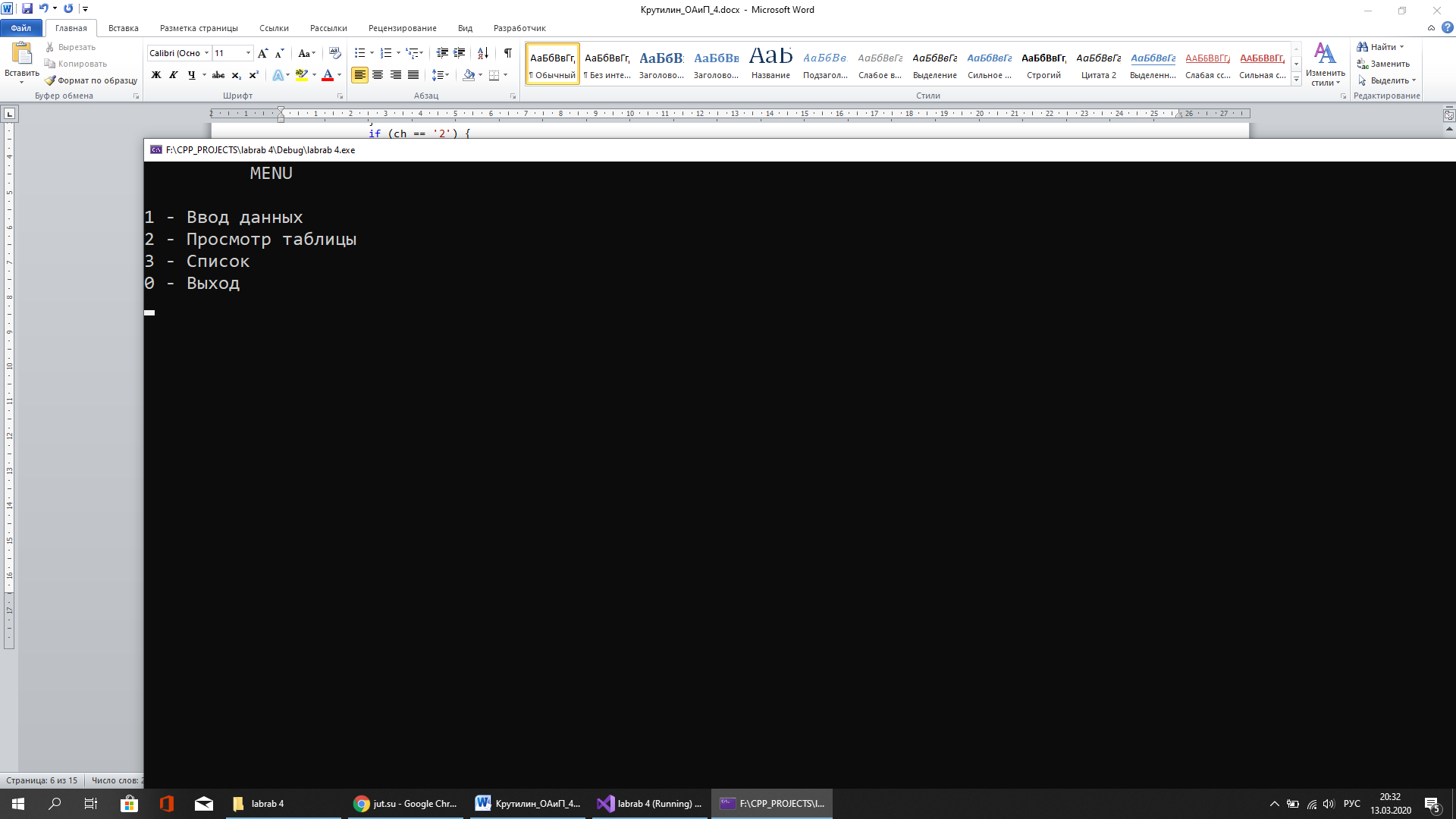
if (ch == '0') {

exit(0);

}

}

}



**Программа целиком:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <fstream>

#include <string>

#include <cstring>

#include <cstdlib>

#include <cstdio>

#include <conio.h>

#include <iomanip>

#include <ctime>

#include <Windows.h>

using namespace std;

struct Contact {

char FIO[40];

char city[20];

char street[20];

int house;

int flat;

char phone[13];

};

Contact\* information = NULL;

int num;

int kol = 0;

char Str[1000000000];

void input();

void Tablitsa();

void list();

void resize();

int check\_input(int field);

void check\_kol();

void show\_menu() {

system("cls");

cout << " MENU \n" << endl;

cout << "1 - Ввод данных" << endl;

cout << "2 - Просмотр таблицы" << endl;

cout << "3 - Список" << endl;

cout << "0 - Выход" << endl;

}

void menu() {

char ch;

show\_menu();

while (1) {

ch = \_getch();

if (ch == '1') {

system("cls");

input();

system("pause");

menu();

}

if (ch == '2') {

system("cls");

check\_kol();

Tablitsa();

system("pause");

menu();

}

if (ch == '3') {

system("cls");

check\_kol();

list();

system("pause");

menu();

}

if (ch == '0') {

exit(0);

}

}

}

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

system("mode con cols=138 lines=70");

menu();

return 0;

}

void input() {

int check = 0;

Contact person;

while (check != 1) {

system("cls");

cout << "Введите ФИО: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(1);

}

}

strcpy\_s(person.FIO, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите город проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(2);

}

}

strcpy\_s(person.city, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите улицу проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(3);

}

}

strcpy\_s(person.street, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер дома проживания: ";

gets\_s(Str);

check = check\_input(4);

}

person.house = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер квартиры проживания: ";

gets\_s(Str);

check = check\_input(5);

}

person.flat = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер телефона: ";

gets\_s(Str);

check = check\_input(6);

if (check != -1) {

check = check\_input(0);

}

}

strcpy\_s(person.phone, Str);

check = 0;

kol++;

resize();

information[kol - 1] = person;

}

void list() {

if (kol == 0) {

cout << "Данные отсутствуют.";

}

else {

for (int i = 0; i < kol; i++) {

cout << "ФИО: " << (\*information).FIO << endl;

cout << "Город: " << (\*information).city << endl;

cout << "Улица: " << (\*information).street << endl;

cout << "Дом: " << (\*information).house << endl;

cout << "Квартира: " << (\*information).flat << endl;

cout << "Номер: " << (\*information).phone << endl;

cout << endl;

}

}

cout << endl;

system("pause");

system("cls");

}

void Tablitsa()

{

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

printf(" | № | Фамииля Имя Отчество | Город | Улица | Номер дома | Номер квартиры | Номер телефона |\n");

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

for (int i = 0; i < kol; i++)

{

printf(" |%5d|%40s|%20s|%20s|%12d|%16d|%16s|\n", i + 1,

information[i].FIO, information[i].city, information[i].street, information[i].house, information[i].flat, information[i].phone);

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

}

}

void resize() {

Contact\* buff = (Contact\*)realloc(information, sizeof(Contact) \* kol);

if (buff == NULL) {

cout << "Не удалось выделить память!" << endl;

exit(-1);

}

information = buff;

}

int check\_input(int field) {

int n = strlen(Str);

int pr = 0;

for (int i = 0; i < n; i++) {

if (Str[i] == ' ') pr++;

}

if (pr == n) {

return -1;

}

if (field == 0) {

if (n != 12) {

return -1;

}

else return 1;

}

if (field == 1) {

if (Str[0] == 0 || n > 40) {

return -1;

}

else return 1;

}

if (field == 2) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 3) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 4) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 5) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 6) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if (Str[i] < '0' || Str[i] > '9')

{

return -1;

}

}

return 1;

}

if (field == 7) {

if (Str[0] == 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

if (num <= 0 || num > kol) {

return -1;

}

else return 1;

}

if (field == 8) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if ((Str[i] < 'A' || Str[i] > 'Z') && (Str[i] < 'a' || Str[i] > 'z') && (Str[i] < 'А' || Str[i] > 'я') && Str[i] != ' ')

{

return -1;

}

}

return 1;

}

}

void check\_kol() {

if (kol == 0) {

cout << "Список пуст!" << endl;

system("pause");

menu();

}

}