Пензенский государственный университет

Кафедра ”Вычислительной техники ”

**Отчет**

по лабораторной работе №1

по дисциплине: “ Логика и основы алгоритмизации в инженерных задачах ”

## по теме: “Простые структуры данных”

## *Выполнил студент группы 19ВВ1:*

## Васильев Л.Р

## *Приняли:*

## Митрохин М.А

ПЕНЗА 2020

**Общие сведения.**

К простым структурам данных языка Си относятся массивы, строки (массивы символов). Структуры (struct) – простейшая составная структура данных.

**Массив** – это конечная совокупность данных одного типа.

Массивы могут состоять из целых чисел, чисел с плавающей запятой, символов и других типов данных. Существуют даже массивы массивов, – многомерные массивы.

Любой массив в программе должен быть описан: вначале указывается тип массива. затем его имя, после имени массива добавляют квадратные скобки ([ ]) внутри которых обычно стоит число, показывающее количество элементов массива.

**Динамический массив**

Для использования функций динамического распределения памяти необходимо подключение библиотеки <malloc.h> или <stdlib.h> (в зависимости от компилятора и используемого стандарта языка).

Функции динамического выделения памяти

void\* malloc(*РазмерМассиваВБайтах*);  
 void\* calloc(*ЧислоЭлементов*, *РазмерЭлементаВБайтах*);

выделяют блок памяти, размером *РазмерМассиваВБайтах* или *ЧислоЭлементов*\**РазмерЭлементаВБайтах* байт, и возвращает указатель на начало блока.

Элементами массива могут быть как данные простых типов, так и составных типов (например, структуры), поэтому для точного определения размера элемента массива в общем случае используется функция

int sizeof(*ТипЭлемента*);

Эта функция возвращает количество байт, занимаемое элементом указанного типа.

Память, динамически выделенная с использованием функций calloc(), malloc(), должна быть освобождена после окончания использования функцией

 free(*указатель*);

Для примера, код

**int** \*a; // указатель на массив

**int** n;

...

n = 10;

// Выделение памяти

a = (**int**\*)malloc(n \* **sizeof**(**int**));

...

a[1] = a[1] + a[3];

...

free(a); // Освобождение памяти

создаёт массив размером 10 элементов и после некоторых преобразований высвобождает занимаемую им память.

Для **генерации случайных чисел** в языке Си используется функция rand(), которая содержится в библиотеке <stdlib.h>.

Функция возвращает случайное целое число в диапазоне от нуля до RAND\_MAX. Глобальная константа RAND\_MAX является максимальным значением, которое может возвращаться функцией rand, это значение определено в заголовочном файле stdlib.h.

Чтобы ограничить сверху случайные числа, можно воспользоваться операцией получения остатка от деления. Остаток от деления на число всегда меньше этого числа. Поэтому для ограничения сверху случайных чисел возьмём остаток от деления на граничное значение.

int k;

k = rand()%100;

В приведенном примере значение k не превысит 99. Т.е. для rand()%n возвращается любое число в диапазоне [0, n). Если необходимо получить числа в диапазоне [0, n], то необходимо изменить выражение на rand()%(n+1).

Для получения чисел из некоторого отрезка [min, max] необходимо найти остаток от деления на длину этого отрезка и выполнить сдвиг возвращаемого остатка на нижнюю границу отрезка

int k;

k = min + rand()%(max-min+1);

При повторных запусках программы функция rand() будет возвращать одинаковые последовательности чисел. Для получения уникальных последовательностей необходимо инициализировать генератор случайных чисел различными значениями. Для этого используется функция srand(), аргументом которой является инициализирующее целое число.

В качестве уникального для каждого запуска программы числа чаще всего используется текущее время, возвращаемое функцией time(), прототип которой описан в библиотеке time.h. Передав в time() в качестве параметра NULL, мы получим целое число, которое можно передать в srand(), установив, таким образом уникальное инициализирующее значение генератора

srand(time(NULL));

int k;

k = rand();

теперь генерируемая последовательность чисел k будет уникальной.

**Структура** - это совокупность переменных одного или нескольких типов, сгруппированных в один элемент.

Объявление структуры осуществляется с помощью ключевого слова struct, за которым идет ее тип и далее список элементов, заключенных в фигурные скобки:

struct тип { тип элемента\_1 имя элемента\_1;

.........

тип элемента\_n имя элемента\_n;

};

Именем элемента может быть любой идентификатор.

Определение структуры:

struct student

{

int Nomzach;

char famil[20];

char name[20], facult[20];

};

После фигурной скобки, заканчивающей список элементов, могут инициализироваться структуры заданного типа, например:

struct student{...} Ivanov, Petrov;

Доступ к отдельным полям структуры осуществляется с помощью оператора . (оператор точка). Например, доступ к полю name в структуре Ivanov осуществляется так:

Ivanov.name;

Код:

ЗАДАНИЕ 1-3

#include "pch.h"

#include <iostream>

#include <stdio.h>

#include <locale>

#include <malloc.h>

#include <stdlib.h>

#include <time.h>

/\* Задание 1-3

1) выводит разницу между максимальным и минимальным числом

2) Устанавливаем размер массива в консоле

3) В массив выводит рандомные числа

\*/

int main()

{

srand(time(NULL));

int arr;

int max = 0;

int min = 0;

int sum = 0;

int \*b;

// Вводим размер массива

printf("Введите размер массива ");

scanf("%d", &arr);

// Выделяем память под массив

b = (int \*)malloc(arr \* sizeof(int));

// Интелезируем массив

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

printf("%d\n", b[i] = rand() % 100);

// Вычисляем разницу

for (int j = 0; j < arr; j++) {

if (b[j] > max) {

max = b[j];

min = b[j] + 1;

}

}

for (int j = 0; j < arr; j++) {

if (b[j] < min) {

min = b[j];

}

}

sum = max - min;

// Выводим ответ

printf("Max = %d\n", max);

printf("Min = %d\n", min);

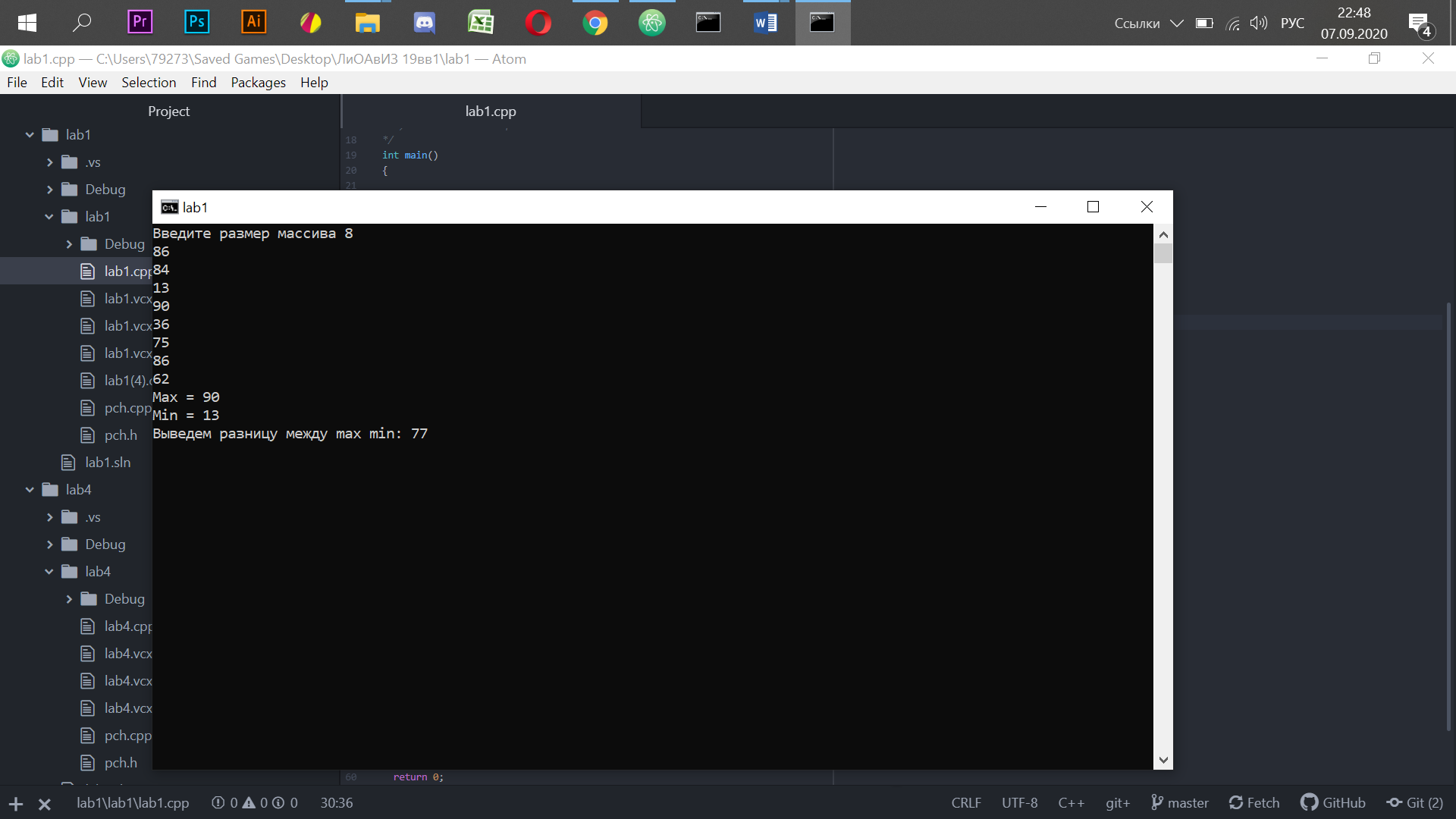
printf("Выведем разницу между max min: %d", sum);

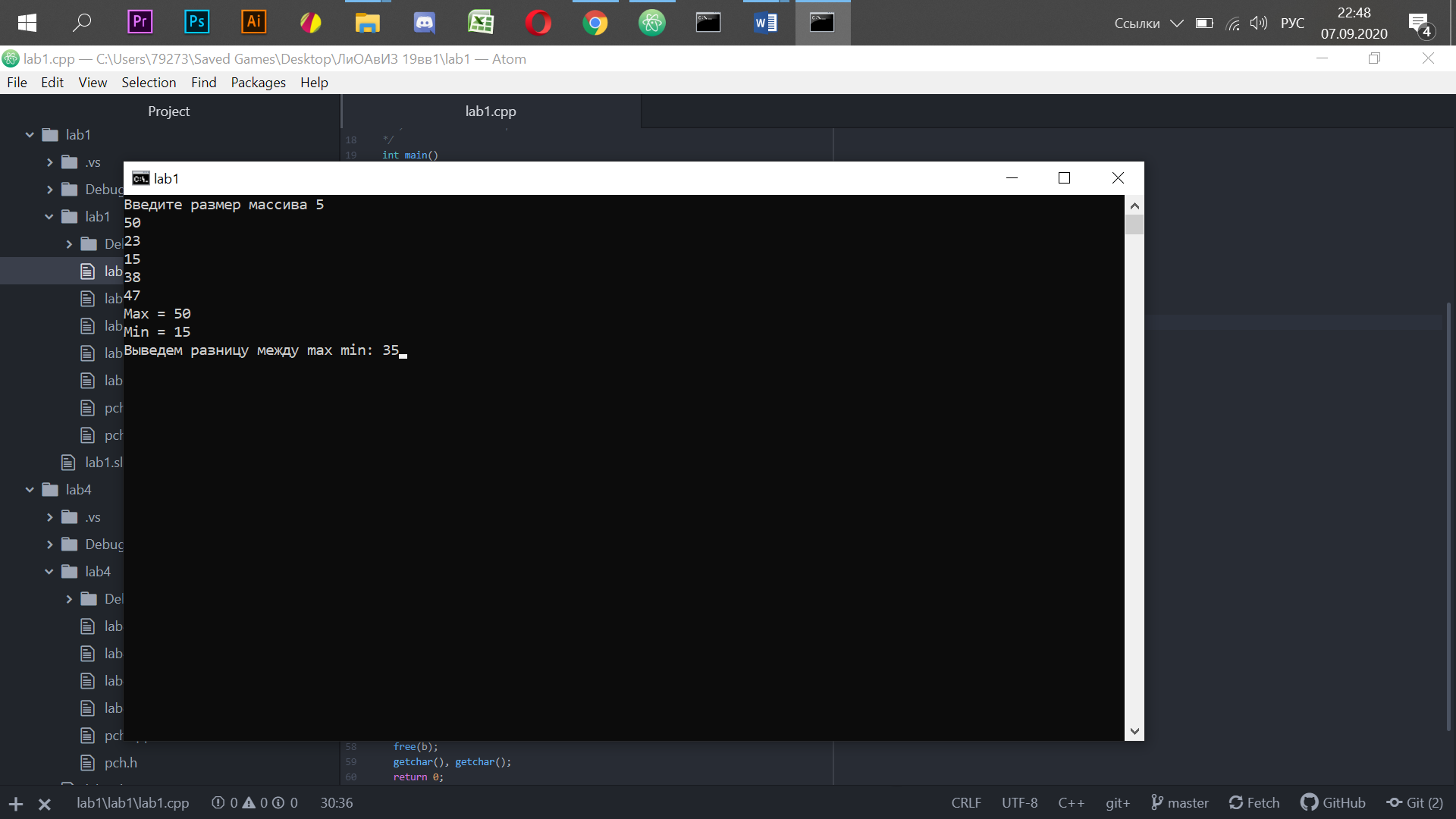
free(b);

getchar(), getchar();

return 0;

}





Задание 4

#include "pch.h"

#include <iostream>

#include <stdio.h>

#include <locale>

#include <malloc.h>

#include <stdlib.h>

#include <time.h>

// Задание 4

// Считаем сумму строк

int main() {

srand(time(NULL));

int \*\*b;

int \*sum;

int arr2;

int arr3;

// Устанавливаем размер матрицы

printf("Введем количество строк в матрице\n: ");

scanf("%d", &arr2);

printf("Введем количество столбцов в матрице\n: ");

scanf("%d", &arr3);

// выделяем память под массив и матрицу

sum = (int \*)malloc(arr2 \* sizeof(int));

b = (int \*\*)malloc(arr2 \* sizeof(int));

for (int i = 0; i < arr2; i++) {

b[i] = (int\*)malloc(arr3 \* sizeof(int));

}

// Интелезируем матрицу и вычисляем сумму строк

for (int i = 0; i < arr2; i++) {

sum[i] = 0;

for (int j = 0; j < arr3; j++) {

printf("%4d", b[i][j] = rand() % 100);

sum[i] += b[i][j];

}

printf("\n\n");

}

// Выводим ответ

for (int i = 0; i < arr2; i++) {

printf("\n%d", sum[i]);

}

for (int i = 0; i < arr2; i++) {

free(b[i]);

}

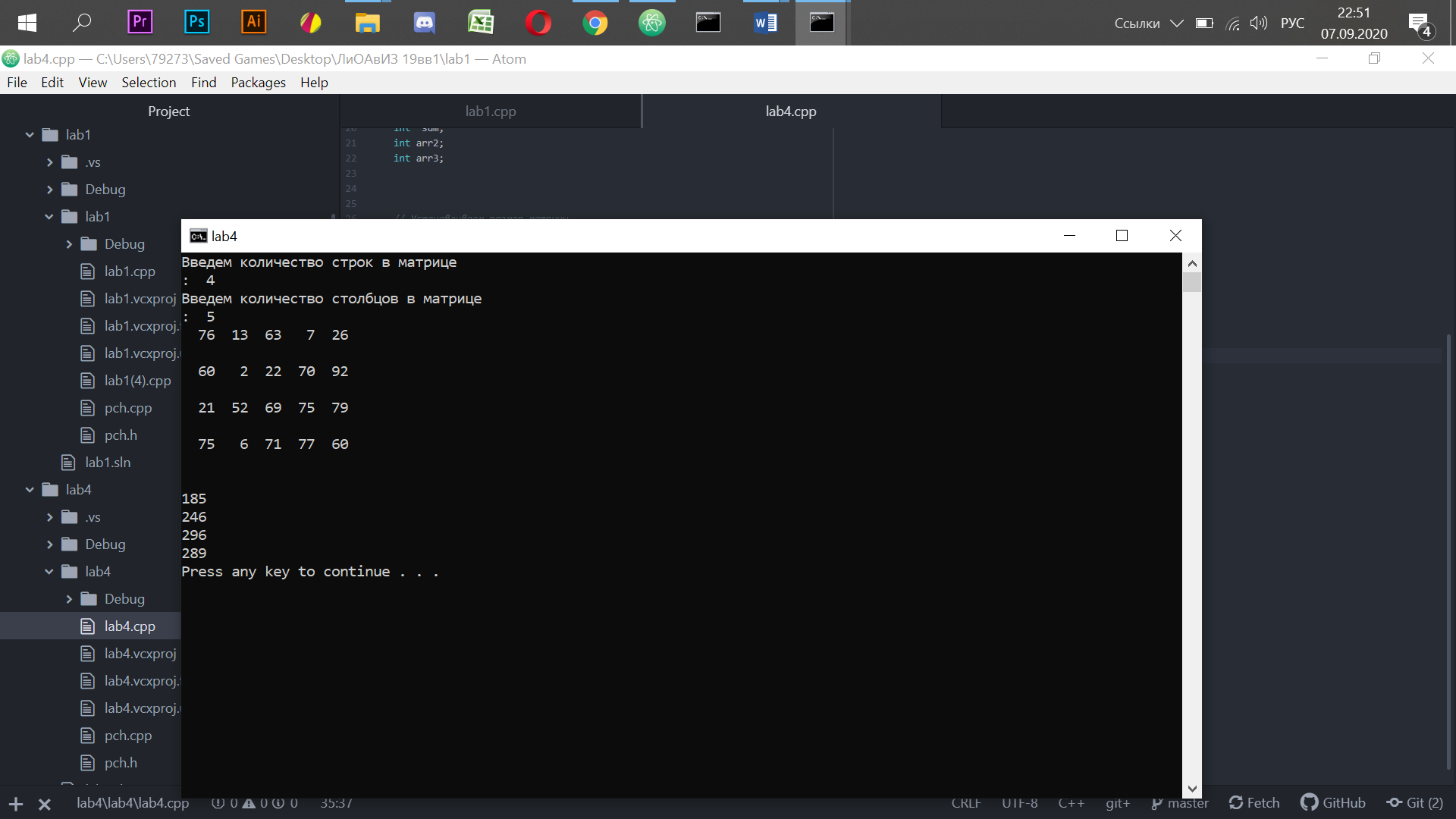
free(b);

free(sum);

getchar();

return 0;

}



Задание 5

#include "pch.h"

#include <iostream>

#include <stdio.h>

#include <locale>

#include <malloc.h>

#include <stdlib.h>

// задание 5

// реализует поиск по фамилии, имени, факультету, номеру зачетной книжки

int main()

{

struct student{

char family[20];

char name[20];

char facult[20];

int Nomzach;

} stud[3], poisk;

printf("Введем фамилию студента:\n");

for (int i = 0; i < 3; i++) {

scanf("%s", stud[i].family);

}

printf("Введем имя студента\n" );

for (int i = 0; i < 3; i++) {

scanf("%s", stud[i].name);

}

printf("Введем название факультета\n" );

for (int i = 0; i < 3; i++) {

scanf("%s", stud[i].facult);

}

printf("Введем номер зачетной книжки\n" );

for (int i = 0; i < 3; i++) {

scanf("%d", &stud[i].Nomzach);

}

for(int i = 0; i<3;i++){

printf("Ученик %s %s обучается на факультете %s, номер зачетной книжки %d \n", stud[i].family, stud[i].name, stud[i].facult, stud[i].Nomzach );

}

do{

printf("\nПоиск: " );

scanf("%s %s %s %d", poisk.family, poisk.name,poisk.facult,&poisk.Nomzach );

for(int i = 0; i<3;i++){

if(strcmp(stud[i].family, poisk.family)==0 && strcmp(stud[i].name, poisk.name)==0 && strcmp(stud[i].facult, poisk.facult)==0 && stud[i].Nomzach == poisk.Nomzach){

printf("\nУченик %s %s обучается на факультете %s, номер зачетной книжки %d\n", poisk.family, poisk.name, poisk.facult,poisk.Nomzach );

}

}

}while(true);

getchar();

getchar();

return 0;

}

