**Отчет по лабораторной работе №VII** по курсу практикум на ЭВМ

Студент группы М8О-101Б-20 Ядров Артем Леонидович, № по списку 28

Контакты www, e-mail, icq, skype temayadrow@gmail.com

Работа выполнена: « » \_\_\_\_\_\_\_\_\_\_\_\_202\_\_\_г.

Преподаватель: доцент каф. 806 Никулин Сергей Петрович

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_202 \_\_ г., итоговая оценка

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Тема:** Разреженные матрицы
2. **Цель работы:** Составить программу на языке Си с процедурами и/или функциями для обработки прямоугольных разреженных матриц с элементами комплексного типов
3. **Задание** (*вариант №* 28 )**:**  Схема размещения: два вектора. Преобразование: Вычислить сумму двух матриц. Проверить, не является ли полученная матрица симметричной. Физическое представление: отображение на массив.
4. **Оборудование** (лабораторное):

ЭВМ Intel Pentium G2140, процессор 3.30 GHz , имя узла сети Cameron с ОП 8096 Мб, НМД 7906 Мб. Терминал ASUS адрес dev/pets/3 Принтер HP Laserjet 6P

Другие устройства

*Оборудование ПЭВМ студента, если использовалось:*

Процессор \_\_Intel core i5-7300HQ 2.50 GHz с ОП 8096 Мб, НМД 131072 Мб. Монитор ASUS

Другие устройства

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Программное обеспечение (лабораторное):**

Операционная система семейства Unix , наименование Ubuntu версия 18.15.0

интерпретатор команд bash версия 4.4.20

Система программирования GNU версия 5.8.13

Редактор текстов emacs версия 25.2.2

Утилиты операционной системы cat

Прикладные системы и программы

Местонахождение и имена файлов программ и данных stud/208104

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства Unix , наименование Fedora версия 33

интерпретатор команд bash версия 5.0.17

Система программирования Clion версия 2020.3

Редактор текстов emacs версия 25.2.2 Утилиты операционной системы cat, gcc

Прикладные системы и программы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных на домашнем компьютере home/Temi4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**6. Идея, метод, алгоритм** решения задачи(в формах:словесной,псевдокода,графической[блок-схема,диаграмма,рисунок,таблица] или формальные спецификации с пред- и постусловиями)

Заведем структуру комплексного числа **complex**, в которой будем хранить рациональную часть (re) и мнимую часть (im).

struct complex {

double re;

double im;

};

Также заведем структуру матрицы **matrix**, в которой будем хранить размеры матрицы, а также два вектора, согласно схеме, и количество ненулевых элементов.

struct matrix {

int LB[10000];

struct complex YE[10000];

int M, N;

int size;

};

Опишем следующие функции:

* struct matrix Read(FILE \*in)

Функция чтения матрицы. Получает входной файл и считывает сначала размер, а затем сами элементы матриц, затем возвращает матрицу ans.

Если элемент ai,j не нулевой (обязательно писать 0, если рациональная/мнимая часть равна 0. Общая формула: c = re + im\*i (в файле пишется как re+imi. Например, 1+1i, 0+1i. Отрицательные мнимые части тоже необходимо писать через знак «+», при чем без скобок. Например, 1+-1i) ), то вычислим значение h = (i-1)\*N+j (т. к. в программе 0-индексация, то h = i\*N+j) и занесем его в конец вектора LB, а само число занесем в конец вектора YE. Увеличим количество ненулевых векторов и перейдем к следующей итерации. После прочтения всех элементов матрицы занесем в конец вектора LB значение «-1», означающее конец вектора.

* void print\_matrix(struct matrix a)

Функция печати матрицы. Получает матрицу, а затем выводит ее размерность, печатает ее во внутреннем представлении и в естественном.

Печать матрицы во внутреннем представлении тривиальна. Для печати матрицы в естественном (человекочитаемом) виде воспользуемся следующим «фокусом»: если элемент матрицы ai,j не нулевой, то значение h = (i-1)\*N+j содержится в массиве LB. Также нетрудно заметить, что если ненулевой элемент ai,j был пройден, то следующий ненулевой элемент при чтении был обработан нами позже, т. к. мы обрабатываем матрицу построчно, а поэтому индекс этого элемента в массиве LB больше текущего. Поэтому будем использовать переменную k, отвечающую за индекс в массиве LB последнего ненулевого элемента. Будем выводить матрицу в «человеческом» виде. Во вложенном цикле будем выводить элемент ai,j матрицы. Для этого попытаемся определить, нулевой это элемент или нет. Вычислим функцию h = (i-1)\*N+j и будем искать ее значение в массиве LB, начиная с индекса k (первоначально равного 0). Если мы нашли этот элемент, то выведем соответствующее значение элемента с этим индексом массиве YE. Если элемент не найден, то выведем 0

* struct matrix sum(struct matrix A, struct matrix B)

Функция суммирования матриц. Получает 2 матрицы, сравнивает размерности и экстренно завершает программу в случае несоответствия размерностей. Если размерности равные, то складывает две матрицы и возвращает сумму.

По формуле суммы матриц cij = aij+bij. С помощью того же «фокуса» будем определять, нулевые ли элементы матриц A и B с индексами i и j. Вычислим cij по указанной выше формуле и определим, нулевой ли это элемент. Если элемент не нулевой, то заносим его в массивы, а также инкрементируем количество ненулевых элементов

* int sym(struct matrix A)

Функция проверки матрицы на симметричность. Получает матрицу и возвращает «1», если матрица симметрична и «0» в противном случае.

Основная идея проверки: если aij — ненулевой элемент, то aji должен быть равен ему. Будем пробегать по массиву LB, определять индексы i и j (исходя из формулы и 0-индексации, i = LB[ind]/N, j = LB[ind]%N), находить значение h = (j-1)\*N+i, а затем искать это значение в массиве LB. Если такого не нашлось, то ненулевому элементу aij при транспонировании соответствует нулевой элемент.

* struct matrix task(struct matrix A, struct matrix B)

Функция выполнения задания — суммирования матриц А и В, а также проверки результата (матрицы С) на симметричность.

Вызывает функции sum и sym, возвращает результат и выводит ответ (симметрична матрица или нет)

В основной части программы будем использовать меню (предварительно проинициализировав входной файл), в котором есть 4 опции:

1. Считывание матриц А и В (Read matrix)

Вызывает функцию Read

1. Печать матрицы (Print matrix)

Вызывает еще одно меню, в котором представляется возможность выбора матрицы, которую необходимо вывести, а затем вызывает функцию print\_matrix

1. Выполнение задания (Task)

Вызов функции Task

1. Выход (Exit)

Выход из меню

1. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию].

**Тесты:**

* 1x1

0

1x1

-1+-1i

* 1x2

0 1+3.14i

1x3

0 0 0

* 6x6

1+-1i 0 0 0 2+4i 0

0 0 0 27+1i 0 0

0 0 0 0 0 0

2+2i 0 0 -1+1i 0 0

0 -3+6i 0 0 0 0

0 0 0 0 0 0+3i

6x6

0 0 0 4+7i 0 0

0 0 0 0 -5+7i 0

0 0 0+45i 0 0 0

2+5i 27+1i 0 0 0 0

0 -2+1i 0 0 4+9i 0

0 0 0 0 0 0

* 6x6

1+-1i 0 0 0 2+4i 0

0 0 0 27+1i 0 0

0 0 0 0 0 0

2+2i 0 0 -1+1i 0 0

2+4i -3+6i 0 0 0 0

0 0 0 0 0 0+3i

6x6

0 0 0 4+7i 0 0

0 0 0 0 -5+7i 0

0 0 0+45i 0 0 0

2+5i 27+1i 0 0 0 0

0 -2+1i 0 0 4+9i 0

0 0 0 0 0 0

* 2x3

1+0i 0 -1+3i

0 37+11i 0

2x3

-1+0i 0 1+-3i

0 -37+-11i 0

*Пункты 1-7 отчета составляются строго до начала лабораторной работы.*

*Допущен к выполнению работы.* **Подпись преподавателя****\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами,подписанныйпреподавателем).

**[Temi4@localhost KP7]$ cat sparse\_matrices.c**

**#include <stdio.h>**

**#include <errno.h>**

**#include <stdlib.h>**

**struct complex {**

**double re;**

**double im;**

**};**

**struct matrix {**

**int LB[100];**

**struct complex YE[100];**

**int M, N;**

**int size;**

**};**

**struct matrix Read(FILE \*in) {**

**struct matrix ans;**

**ans.size = 0;**

**struct complex c;**

**fscanf(in, "%dx%d\n", &ans.M, &ans.N);**

**for (int i = 0; i < ans.M; i++) {**

**for (int j = 0; j < ans.N; j++) {**

**if (fscanf(in, "%lf+%lfi", &c.re, &c.im) == 2) {**

**ans.LB[ans.size] = i \* ans.N + j;**

**ans.YE[ans.size] = c;**

**ans.size++;**

**}**

**}**

**fscanf(in, "\n");**

**}**

**ans.LB[ans.size] = -1;**

**return ans;**

**}**

**void print\_matrix(struct matrix a) {**

**if (!(a.M && a.N)){**

**printf("Matrix doesn't exist\n");**

**return;**

**}**

**printf("Matrix size: %dx%d\n", a.M, a.N);**

**printf("Internal representation:\n");**

**printf("LB\n");**

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

**printf("%d\t", a.LB[i]);**

**}**

**printf("\nYE\n");**

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

**printf("%lf+%lfi\t", a.YE[i].re, a.YE[i].im);**

**}**

**printf("\nHuman readable:\n");**

**int k = 0;**

**struct complex c;**

**for (int i = 0; i < a.M; i++) {**

**for (int j = 0; j < a.N; j++) {**

**c.re = 0;**

**c.im = 0;**

**int h = i \* a.N + j;**

**for (int ind = k; ind < a.size; ind++) {**

**if (a.LB[ind] == h) {**

**c.re = a.YE[ind].re;**

**c.im = a.YE[ind].im;**

**k = ind;**

**}**

**}**

**if (c.re != 0 || c.im != 0) {**

**printf("%lf+%lfi\t", c.re, c.im);**

**} else {**

**printf("0\t");**

**}**

**}**

**printf("\n");**

**}**

**}**

**struct matrix sum(struct matrix A, struct matrix B) {**

**if ((A.M != B.M) || (A.N != B.N)) {**

**perror("Difficult matrix size");**

**\_Exit(1);**

**}**

**struct matrix ans;**

**ans.M = A.M;**

**ans.N = A.N;**

**ans.size = 0;**

**struct complex ca, cb;**

**int ka = 0, kb = 0;**

**for (int i = 0; i < ans.M; i++) {**

**for (int j = 0; j < ans.N; j++) {**

**int h = i \* ans.N + j;**

**ca.im = 0;**

**ca.re = 0;**

**cb.im = 0;**

**cb.re = 0;**

**for (int ind\_a = ka; ind\_a < A.size; ind\_a++) {**

**if (A.LB[ind\_a] == h) {**

**ca = A.YE[ind\_a];**

**ka = ind\_a;**

**break;**

**}**

**}**

**for (int ind\_b = kb; ind\_b < B.size; ind\_b++) {**

**if (B.LB[ind\_b] == h) {**

**cb = B.YE[ind\_b];**

**kb = ind\_b;**

**break;**

**}**

**}**

**struct complex answer;**

**answer.re = ca.re + cb.re;**

**answer.im = ca.im + cb.im;**

**if (answer.re != 0 || answer.im != 0) {**

**ans.LB[ans.size] = h;**

**ans.YE[ans.size] = answer;**

**ans.size++;**

**}**

**}**

**}**

**ans.LB[ans.size] = -1;**

**return ans;**

**}**

**int sym(struct matrix A) {**

**int ans = 1;**

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

**int c = 0;**

**int h = A.LB[i] / A.N + (A.LB[i] % A.N) \* A.N;**

**for (int j = 0; j < A.size; j++) {**

**if ((A.LB[j] == h) && (A.YE[i].re == A.YE[j].re) && (A.YE[i].im == A.YE[j].im)) {**

**c = 1;**

**break;**

**}**

**}**

**if (!c) {**

**ans = 0;**

**break;**

**}**

**}**

**return ans;**

**}**

**struct matrix task(struct matrix A, struct matrix B) {**

**struct matrix C = sum(A, B);**

**int c = sym(C);**

**if (c) {**

**printf("C is symmetric matrix\n");**

**} else {**

**printf("C is not symmetric matrix\n");**

**}**

**return C;**

**}**

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

**struct matrix A, B, C;**

**int g = 1;**

**int c;**

**if (argc != 2) {**

**printf("Use: program\_name input\_file\n");**

**return 0;**

**}**

**FILE \*input = fopen(argv[1], "r");**

**if (!input) {**

**perror("Can't open file");**

**return 1;**

**}**

**A = Read(input);**

**B = Read(input);**

**C.size = 0;**

**int choose;**

**while (g) {**

**printf("1.Print matrix 2. Task 3. Exit\n");**

**scanf("%d", &c);**

**switch (c) {**

**case 1: {**

**printf("Choose matrix: 1.A 2.B 3.C\n");**

**scanf("%d", &choose);**

**switch (choose) {**

**case 1: {**

**print\_matrix(A);**

**break;**

**}**

**case 2: {**

**print\_matrix(B);**

**break;**

**}**

**case 3: {**

**print\_matrix(C);**

**break;**

**}**

**default: {**

**printf("Wrong answer\n");**

**break;**

**}**

**}**

**break;**

**}**

**case 2: {**

**C = task(A, B);**

**break;**

**}**

**case 3: {**

**g = 0;**

**break;**

**}**

**default: {**

**printf("Wrong answer\n");**

**}**

**}**

**}**

**return 0;**

**}**

**[Temi4@localhost KP7]$ gcc sparse\_matrices.c**

**[Temi4@localhost KP7]$ ./a.out**

**Use: program\_name input\_file**

**[Temi4@localhost KP7]$ cat input**

**1x1**

**0**

**1x1**

**-1+-1i**

**[Temi4@localhost KP7]$ ./a.out input**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**1**

**Matrix size: 1x1**

**Internal representation:**

**LB**

**YE**

**Human readable:**

**0**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**2**

**Matrix size: 1x1**

**Internal representation:**

**LB**

**0**

**YE**

**-1.000000+-1.000000i**

**Human readable:**

**-1.000000+-1.000000i**

**1.Print matrix 2. Task 3. Exit**

**2**

**C is symmetric matrix**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**3**

**Matrix size: 1x1**

**Internal representation:**

**LB**

**0**

**YE**

**-1.000000+-1.000000i**

**Human readable:**

**-1.000000+-1.000000i**

**1.Print matrix 2. Task 3. Exit**

**3**

**[Temi4@localhost KP7]$ cat input**

**1x2**

**0 1+3.14i**

**1x3**

**0 0 0**

**[Temi4@localhost KP7]$ ./a.out input**

**1.Print matrix 2. Task 3. Exit**

**2**

**Difficult matrix size: Success**

**[Temi4@localhost KP7]$ cat input**

**6x6**

**1+-1i 0 0 0 2+4i 0**

**0 0 0 27+1i 0 0**

**0 0 0 0 0 0**

**2+2i 0 0 -1+1i 0 0**

**0 -3+6i 0 0 0 0**

**0 0 0 0 0 0+3i**

**6x6**

**0 0 0 4+7i 0 0**

**0 0 0 0 -5+7i 0**

**0 0 0+45i 0 0 0**

**2+5i 27+1i 0 0 0 0**

**0 -2+1i 0 0 4+9i 0**

**0 0 0 0 0 0**

**[Temi4@localhost KP7]$ ./a.out input**

**1.Print matrix 2. Task 3. Exit**

**2**

**C is not symmetric matrix**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**3**

**Matrix size: 6x6**

**Internal representation:**

**LB**

**0 3 4 9 10 14 18 19 21 25 28 35**

**YE**

**1.000000+-1.000000i 4.000000+7.000000i 2.000000+4.000000i 27.000000+1.000000i -5.000000+7.000000i 0.000000+45.000000i 4.000000+7.000000i 27.000000+1.000000i -1.000000+1.000000i -5.000000+7.000000i 4.000000+9.000000i 0.000000+3.000000i**

**Human readable:**

**1.000000+-1.000000i 0 0 4.000000+7.000000i 2.000000+4.000000i 0**

**0 0 0 27.000000+1.000000i -5.000000+7.000000i 0**

**0 0 0.000000+45.000000i 0 0 0**

**4.000000+7.000000i 27.000000+1.000000i 0 -1.000000+1.000000i 0 0**

**0 -5.000000+7.000000i 0 0 4.000000+9.000000i 0**

**0 0 0 0 0 0.000000+3.000000i**

**1.Print matrix 2. Task 3. Exit**

**3**

**[Temi4@localhost KP7]$ cat input**

**6x6**

**1+-1i 0 0 0 2+4i 0**

**0 0 0 27+1i 0 0**

**0 0 0 0 0 0**

**2+2i 0 0 -1+1i 0 0**

**2+4i -3+6i 0 0 0 0**

**0 0 0 0 0 0+3i**

**6x6**

**0 0 0 4+7i 0 0**

**0 0 0 0 -5+7i 0**

**0 0 0+45i 0 0 0**

**2+5i 27+1i 0 0 0 0**

**0 -2+1i 0 0 4+9i 0**

**0 0 0 0 0 0**

**[Temi4@localhost KP7]$ ./a.out input**

**1.Print matrix 2. Task 3. Exit**

**2**

**C is symmetric matrix**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**3**

**Matrix size: 6x6**

**Internal representation:**

**LB**

**0 3 4 9 10 14 18 19 21 24 25 28 35**

**YE**

**1.000000+-1.000000i 4.000000+7.000000i 2.000000+4.000000i 27.000000+1.000000i -5.000000+7.000000i 0.000000+45.000000i 4.000000+7.000000i 27.000000+1.000000i -1.000000+1.000000i 2.000000+4.000000i -5.000000+7.000000i 4.000000+9.000000i 0.000000+3.000000i**

**Human readable:**

**1.000000+-1.000000i 0 0 4.000000+7.000000i 2.000000+4.000000i 0**

**0 0 0 27.000000+1.000000i -5.000000+7.000000i 0**

**0 0 0.000000+45.000000i 0 0 0**

**4.000000+7.000000i 27.000000+1.000000i 0 -1.000000+1.000000i 0 0**

**2.000000+4.000000i -5.000000+7.000000i 0 0 4.000000+9.000000i 0**

**0 0 0 0 0 0.000000+3.000000i**

**1.Print matrix 2. Task 3. Exit**

**3**

**[Temi4@localhost KP7]$ cat input**

**2x3**

**1+0i 0 -1+3i**

**0 37+11i 0**

**2x3**

**-1+0i 0 1+-3i**

**0 -37+-11i 0**

**[Temi4@localhost KP7]$ ./a.out input**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**1**

**Matrix size: 2x3**

**Internal representation:**

**LB**

**0 2 4**

**YE**

**1.000000+0.000000i -1.000000+3.000000i 37.000000+11.000000i**

**Human readable:**

**1.000000+0.000000i 0 -1.000000+3.000000i**

**0 37.000000+11.000000i 0**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**2**

**Matrix size: 2x3**

**Internal representation:**

**LB**

**0 2 4**

**YE**

**-1.000000+0.000000i 1.000000+-3.000000i -37.000000+-11.000000i**

**Human readable:**

**-1.000000+0.000000i 0 1.000000+-3.000000i**

**0 -37.000000+-11.000000i 0**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**3**

**Matrix doesn't exist**

**1.Print matrix 2. Task 3. Exit**

**2**

**C is symmetric matrix**

**1.Print matrix 2. Task 3. Exit**

**1**

**Choose matrix: 1.A 2.B 3.C**

**3**

**Matrix size: 2x3**

**Internal representation:**

**LB**

**YE**

**Human readable:**

**0 0 0**

**0 0 0**

**1.Print matrix 2. Task 3. Exit**

**3**

1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные события(ошибки в сценарии и программе,нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  | или |  |  |  |  |  |
|  | дом. |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы

1. **Выводы**

Я научился обрабатывать разреженные матрицы на Си.

Недочёты при выполнении задания могут быть устранены следующим образом: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_