|  |  |
| --- | --- |
|  | **МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**  **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»** |
|  |
|  |
|  |

|  |  |
| --- | --- |
| Институт (факультет) | Институт компьютерных наук |
| Кафедра | Кафедра прикладной математики |

ЛАБОРАТОРНАЯ РАБОТА № 1

По дисциплине: «Программирование»

На тему: «Разработка функциональных модулей обработки агрегатных данных. Работа с текстовыми файлами».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ПМ-24-1 |  |  |  | Сараев А.П. |
|  | группа |  | подпись, дата |  | фамилия, инициалы |
|  |  |  |  |  |  |
| Руководитель |  |  |  |  |  |
| К.т.н., доцент кафедры ПМ | |  |  |  | Мирошников А.И. |
| ученая степень, ученое звание | |  | подпись, дата |  | фамилия, инициалы |

Липецк 2024

**Цель работы**

Освоить способы динамического выделения памяти и разработки функций на примере программирования задач линейной алгебры. Освоить работу с текстовыми файлами данных.

**Задание кафедры**

1. Реализовать вычисление заданного матричного выражения.

2. Реализовать интерфейс для работы с пользователем. Программа должна осуществлять проверку некорректного ввода данных на всех этапах.

3. Реализовать ввод матриц в трех вариантах по выбору пользователя: из файла (значения констант, размеры матриц и их содержимое записаны в файле), консольный ввод (значения констант, размеры матриц и их содержимое определяются пользователем), задание случайными числами (значения констант, размеры матриц определяются пользователем; разброс случайных значений задается пользователем). Имена файлов задаются пользователем. Имена файлов могут быть с пробелом.

4. Реализовать вывод матриц, промежуточных и конечных результатов в двух вариантах по выбору пользователя: в файл, консольный вывод. Имена файлов задаются пользователем. Имена файлов могут быть с пробелом.

5. Операции транспонирования, вычисления суммы, произведения на матрицу и на константу реализовать в виде отдельных функций.

Необходимо контролировать размерность матриц при осуществлении матричных операций. Если размерность матриц не позволяет провести операцию, выдавать сообщение об ошибке.

6. Программа должна представлять результаты вычислений (в том числе промежуточных) в виде матрицы со значениями, выровненными по столбцам, например:

4 5 5

345 54 2

-102 4 85

7. Разработать 2 варианта работы программы:

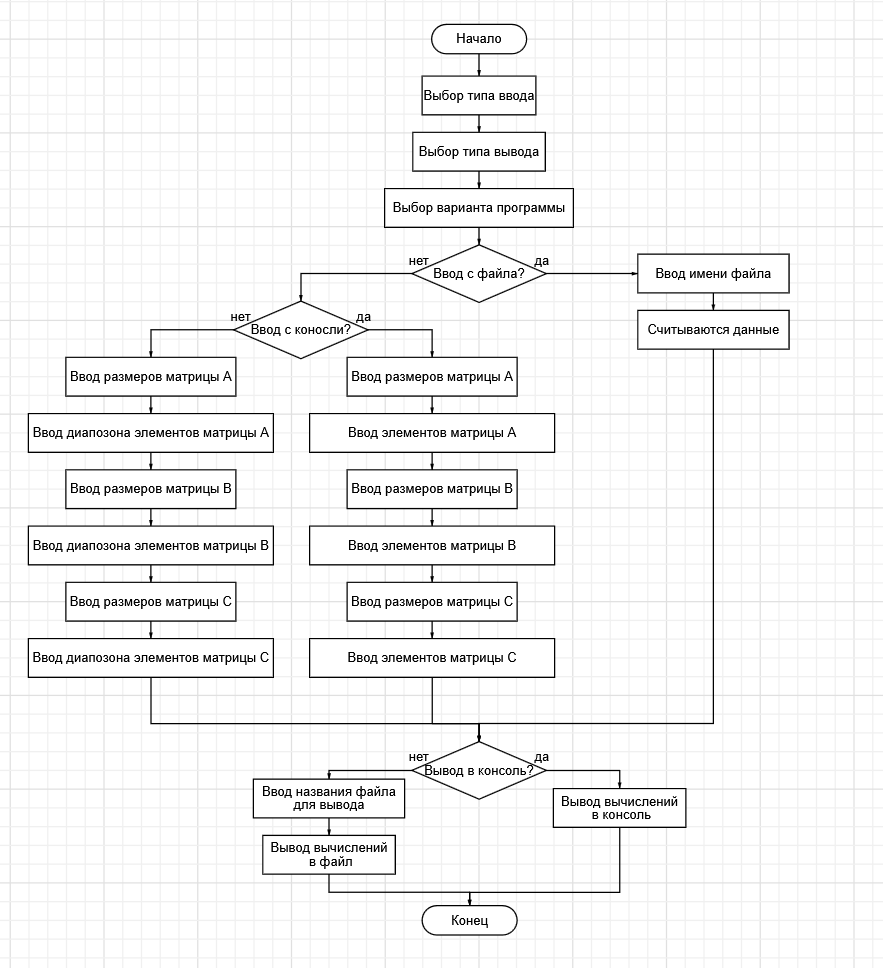
1. Хранить матрицу в виде одномерного массива

2. Хранить матрицу в виде массива указателей

**Задания к лабораторной работе №1**



**Блок-схема**



**Текст программы**

#include <iostream>

#include <fstream>

#include <iomanip>

#include <clocale>

using namespace std;

//////////////////////////////////////////////////////////////////////////

///////////////////////Общие функции//////////////////////////////////////

void output\_file\_name(ofstream &fout) {

if (fout.is\_open()) {

fout.close();

}

cout << "Введите название файла: ";

bool f = 0;

char\* file\_name = new char[256];

while (cin.getline(file\_name, 256)) {

fout.open(file\_name);

if (!fout.is\_open()) {

if (f == 0) {

f = 1;

continue;

}

cout << "Не найден такой файл.\n";

cout << "Введите название файла: ";

}

else

break;

}

}

void input\_file\_name(ifstream &fin) {

if (fin.is\_open()) {

fin.close();

}

cout << "Введите название файла: ";

char\* file\_name = new char[256];

bool f = 0;

while (cin.getline(file\_name, 256)) {

fin.open(file\_name);

if (!fin.is\_open()) {

if (f == 0) {

f = 1;

continue;

}

cout << "Не найден такой файл.\n";

cout << "Введите название файла: ";

}

else

break;

}

}

int input\_type() {

int ch1;

cout << "Выберете тип ввода данных: 1 - из файла, 2 - в консоли, 3 - случайные числа: ";

while (cin >> ch1) {

if (ch1 == 1) {

break;

}

else if (ch1 == 2 || ch1 == 3)

break;

else if (ch1 != 1 && ch1 != 2 && ch1 != 3) {

cout << "Некоректный ввод.\n";

cout << "Выберете тип ввода данных: 1 - из файла, 2 - в консоли, 3 - случайные числа: ";

}

}

return ch1;

}

int output\_type() {

int ch2;

cout << "Выберете способ вывода результатов: 1 - в файл, 2 - в консоль: ";

while (cin >> ch2) {

if (ch2 == 1) {

break;

}

else if (ch2 == 2)

break;

else if (ch2 != 1 && ch2 != 2) {

cout << "Некоректный ввод.\n";

cout << "Выберете способ вывода результатов: 1 - в файл, 2 - в консоль: ";

}

}

return ch2;

}

int var\_type() {

int ch3;

cout << "Выберете вариант программы: 1 - одномерный массив, 2 - массив указателей: ";

while (cin >> ch3) {

if (ch3 == 1 || ch3 == 2) {

break;

}

else{

cout << "Некоректный ввод.\n";

cout << "Выберете вариант программы: 1 - одномерный массив, 2 - массив указателей: ";

}

}

return ch3;

}

int k\_const(int c, ifstream& fin) {

int k;

if (c == 1)

fin >> k;

else {

cout << "Введите значение коснтанты (k != 0, |k| <= 20) k = ";

while (cin >> k) {

if (k != 0 && abs(k) <= 20)

break;

cout << "k != 0, |k| <= 20 !\n";

cout << "Введите значение коснтанты (k != 0, |k| <= 20) k = ";

}

}

return k;

}

void input\_matrix\_size(int& size\_x, int& size\_y, int type, ifstream& fin, char matr\_name) {

int a, b;

if (type == 1) {

fin >> a >> b;

size\_x = a, size\_y = b;

return;

}

else {

cout << "Введите размер матрицы (от 1 до 10)" << matr\_name << ": ";

while (cin >> a >> b) {

if (a >= 1 && a <= 10 && b >= 1 && b <= 10) {

size\_x = a, size\_y = b;

return;

}

else {

cout << "Неправильный размер матрицы!\n";

cout << "Введите размер матрицы (от 1 до 10)" << matr\_name << ": ";

}

}

}

}

bool check\_add\_sizes(int a\_n, int a\_m, int b\_n, int b\_m) {

if (a\_n != b\_n || a\_m != b\_m) {

cerr << "Несовпадающие размеры матриц.\n";

return false;

}

return true;

}

bool check\_mul\_sizes(int a\_n, int a\_m, int b\_n, int b\_m) {

if (a\_m != b\_n) {

cerr << "Несовпадающие размеры матриц.\n";

return false;

}

return true;

}

//////////////////////////////////////////////////////////////////////////

///////////////////////1 ВАРИАНТ (одномерный массив)//////////////////////

void generate\_matrices(double\* a, int a\_n, int a\_m, int l, int r) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++) {

a[i \* a\_m + j] = rand() % (r - l) + l;

}

}

}

bool input\_matrix(double\* matr, int size\_x, int size\_y, int type, ifstream &fin, char matr\_name) {

if (type == 1) {

for (int i = 0; i < size\_x; i++) {

for (int j = 0; j < size\_y; j++) {

fin >> matr[i \* size\_y + j];

if (abs(matr[i \* size\_y + j]) > 20) {

cout << "Числа в матрице должрны быть |a| <= 20 или пропущены некоторые значения!\n";

return false;

}

}

}

}

else if (type == 2) {

cout << "Введите значения в матрице " << matr\_name << ": \n";

for (int i = 0; i < size\_x; i++) {

for (int j = 0; j < size\_y; j++) {

cin >> matr[i \* size\_y + j];

if (abs(matr[i \* size\_y + j]) > 20) {

cout << "Числа в матрице должрны быть |a| <= 20!\n";

return false;

}

}

}

}

else{

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

int l\_a, r\_a;

while (cin >> l\_a >> r\_a) {

if (abs(l\_a) > 20 || abs(r\_a) > 20) {

cout << "Числа не могут быть вне диапозона.\n";

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

continue;

}

if (l\_a > r\_a) {

cout << "Левая граница не может быть больше правой.\n";

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

continue;

}

break;

}

generate\_matrices(matr, size\_x, size\_y, l\_a, r\_a);

}

return true;

}

void add\_matrices(double\* a, int a\_n, int a\_m, double\* b, int b\_n, int b\_m, double\* res) {

if (a\_n != b\_n || a\_m != b\_m) {

cerr << "Несовпадающие размеры матриц.";

exit(0);

}

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++)

res[i \* a\_m + j] = a[i \* a\_m + j] + b[i \* b\_m + j];

}

}

void mul\_matr(double\* a, int a\_n, int a\_m, double\* b, int b\_n, int b\_m, double\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < b\_m; j++) {

for (int l = 0; l < b\_n; l++) {

res[i \* b\_m + j] += a[i \* b\_n + l] \* b[l \* b\_m + j];

}

}

}

}

void div\_matr\_by\_num(double\* a, int a\_n, int a\_m, int k, double\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++)

res[i \* a\_m + j] = a[i \* a\_m + j] / k;

}

}

void transp\_matr(double\* a, int a\_n, int a\_m, double\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++) {

res[j \* a\_n + i] = a[i \* a\_m + j];

}

}

}

void print\_mat(double\* a, int a\_n, int a\_m, int c, ofstream &fout) {

if (c == 1) {

for (int i = 0; i < a\_n; i++) {

fout << setw(2);

for (int j = 0; j < a\_m; j++)

fout << a[i \* a\_m + j] << setw(10);

fout << '\n';

}

}

else {

for (int i = 0; i < a\_n; i++) {

cout << setw(2);

for (int j = 0; j < a\_m; j++)

cout << a[i \* a\_m + j] << setw(10);

cout << '\n';

}

}

}

/////////////////////////////////////////////////////////////////////////

///////////////////////2 ВАРИАНТ (масив указателей)//////////////////////

void generate\_matrices(double\*\* a, int a\_n, int a\_m, int l, int r) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++) {

a[i][j] = rand() % (r - l) + l;

}

}

}

bool input\_matrix(double\*\* matr, int size\_x, int size\_y, int type, ifstream& fin, char matr\_name) {

if (type == 1) {

for (int i = 0; i < size\_x; i++) {

for (int j = 0; j < size\_y; j++) {

fin >> matr[i][j];

if (abs(matr[i][j]) > 20) {

cout << "Числа в матрице должрны быть |a| <= 20 или пропущены некоторые значения!\n";

return false;

}

}

}

}

else if (type == 2) {

cout << "Введите значения в матрице " << matr\_name << ": \n";

for (int i = 0; i < size\_x; i++) {

for (int j = 0; j < size\_y; j++) {

cin >> matr[i][j];

if (abs(matr[i][j]) > 20) {

cout << "Числа в матрице должрны быть |a| <= 20!\n";

return false;

}

}

}

}

else {

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

int l\_a, r\_a;

while (cin >> l\_a >> r\_a) {

if (abs(l\_a) > 20 || abs(r\_a) > 20) {

cout << "Числа не могут быть вне диапозона.\n";

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

continue;

}

if (l\_a > r\_a) {

cout << "Левая граница не может быть больше правой.\n";

cout << "Введите диапозон чисел в матрице (от -20 до 20)" << matr\_name << ": ";

continue;

}

break;

}

generate\_matrices(matr, size\_x, size\_y, l\_a, r\_a);

}

return true;

}

void add\_matrices(double\*\* a, int a\_n, int a\_m, double\*\* b, int b\_n, int b\_m, double\*\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++)

res[i][j] = a[i][j] + b[i][j];

}

}

void mul\_matr(double\*\* a, int a\_n, int a\_m, double\*\* b, int b\_n, int b\_m, double\*\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < b\_m; j++) {

for (int l = 0; l < b\_n; l++) {

res[i][j] += a[i][l] \* b[l][j];

}

}

}

}

void div\_matr\_by\_num(double\*\* a, int a\_n, int a\_m, double k, double\*\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++)

res[i][j] = a[i][j] / k;

}

}

void transp\_matr(double\*\* a, int a\_n, int a\_m, double\*\* res) {

for (int i = 0; i < a\_n; i++) {

for (int j = 0; j < a\_m; j++) {

res[j][i] = a[i][j];

}

}

}

void print\_mat(double\*\* a, int a\_n, int a\_m, int c, ofstream& fout) {

if (c == 1) {

for (int i = 0; i < a\_n; i++) {

fout << setw(2);

for (int j = 0; j < a\_m; j++)

fout << a[i][j] << setw(10);

fout << '\n';

}

}

else {

for (int i = 0; i < a\_n; i++) {

cout << setw(2);

for (int j = 0; j < a\_m; j++)

cout << a[i][j] << setw(10);

cout << '\n';

}

}

}

/////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////

int main() {

setlocale(LC\_ALL, "Russian");

ifstream fin;

ofstream fout;

while (1) {

cout << '\n';

int ch1 = input\_type();

if (ch1 == 1) input\_file\_name(fin);

int ch2 = output\_type();

if (ch2 == 1) output\_file\_name(fout), fout << fixed << setprecision(4);

else cout << fixed << setprecision(4);

int ch3 = var\_type();

double k = k\_const(ch1, fin);

if (k == 0 || abs(k) > 20) {

cout << "k не может быть равен 0 или |k| > 20 !\n";

continue;

}

/////////////////////////////////////////////////////////////////

int a\_n, a\_m, b\_n, b\_m, c\_n, c\_m;

input\_matrix\_size(a\_n, a\_m, ch1, fin, 'A');

if (a\_n < 1 || a\_n > 10 || a\_m < 1 || a\_m > 10) {

cout << "Неправильный размер матрицы (от 1 до 10 столбцов и колонок)!\n";

continue;

}

double\* a1 = new double[a\_n \* a\_m];

double\*\* a2 = new double\* [a\_n];

if (ch3 == 2) {

delete[] a1;

for (int i = 0; i < a\_n; i++) {

a2[i] = new double[a\_m];

}

}

else delete[] a2;

if (ch3 == 1) {

if (!input\_matrix(a1, a\_n, a\_m, ch1, fin, 'A')) continue;

else {

if (ch2 == 1) fout << "Матрица A:\n";

else cout << "Матрица A:\n";

print\_mat(a1, a\_n, a\_m, ch2, fout);

}

}

else { if (!input\_matrix(a2, a\_n, a\_m, ch1, fin, 'A')) continue;

else {

if (ch2 == 1) fout << "Матрица A:\n";

else cout << "Матрица A:\n";

print\_mat(a2, a\_n, a\_m, ch2, fout);

}

}

input\_matrix\_size(b\_n, b\_m, ch1, fin, 'B');

if (b\_n < 1 || b\_n > 10 || b\_m < 1 || b\_m > 10) {

cout << "Неправильный размер матрицы (от 1 до 10 столбцов и колонок)!\n";

continue;

}

double\* b1 = new double[b\_n \* b\_m];

double\*\* b2 = new double\* [b\_n];

if (ch3 == 2) {

delete[] b1;

for (int i = 0; i < b\_n; i++) {

b2[i] = new double[b\_m];

}

}

else delete[] b2;

if (ch3 == 1) { if (!input\_matrix(b1, b\_n, b\_m, ch1, fin, 'B')) continue;

else {

if (ch2 == 1) fout << "Матрица B:\n";

else cout << "Матрица B:\n";

print\_mat(b1, b\_n, b\_m, ch2, fout);

}

}

else { if (!input\_matrix(b2, b\_n, b\_m, ch1, fin, 'B')) continue;

else {

if (ch2 == 1) fout << "Матрица B:\n";

else cout << "Матрица B:\n";

print\_mat(b2, b\_n, b\_m, ch2, fout);

}

}

input\_matrix\_size(c\_n, c\_m, ch1, fin, 'C');

if (c\_n < 1 || c\_n > 10 || c\_m < 1 || c\_m > 10) {

cout << "Неправильный размер матрицы (от 1 до 10 столбцов и колонок)!\n";

continue;

}

double\* c1 = new double[c\_n \* c\_m];

double\*\* c2 = new double\* [c\_n];

if (ch3 == 2) {

delete[] c1;

for (int i = 0; i < c\_n; i++) {

c2[i] = new double[c\_m];

}

}

else delete[] c2;

if (ch3 == 1) { if (!input\_matrix(c1, c\_n, c\_m, ch1, fin, 'C')) continue;

else {

if (ch2 == 1) fout << "Матрица C:\n";

else cout << "Матрица C:\n";

print\_mat(c1, c\_n, c\_m, ch2, fout);

}

}

else { if (!input\_matrix(c2, c\_n, c\_m, ch1, fin, 'C')) continue;

else {

if (ch2 == 1) fout << "Матрица C:\n";

else cout << "Матрица C:\n";

print\_mat(c2, c\_n, c\_m, ch2, fout);

}

}

//////////////////////////////////////////////////////////////

if (ch2 == 1) fout << "B / k:\n";

else cout << "B / k:\n";

double\* d1 = new double[b\_n \* b\_m];

double\*\* d2 = new double\* [b\_n];

if (ch3 == 2) {

delete[] d1;

for (int i = 0; i < b\_n; i++) {

d2[i] = new double[b\_m];

}

}

else delete[] d2;

if (ch3 == 1) {

div\_matr\_by\_num(b1, b\_n, b\_m, k, d1);

print\_mat(d1, b\_n, b\_m, ch2, fout);

}

else {

div\_matr\_by\_num(b2, b\_n, b\_m, k, d2);

print\_mat(d2, b\_n, b\_m, ch2, fout);

}

if (ch2 == 1) fout << "A + B / k:\n";

else cout << "A + B / k:\n";

if (!check\_add\_sizes(a\_n, a\_m, b\_n, b\_m))

continue;

double\* e1 = new double[b\_n \* b\_m];

double\*\* e2 = new double\* [b\_n];

if (ch3 == 2) {

delete[] e1;

for (int i = 0; i < b\_n; i++) {

e2[i] = new double[b\_m];

}

}

else delete[] e2;

if (ch3 == 1) {

add\_matrices(a1, a\_n, a\_m, d1, b\_n, b\_m, e1);

print\_mat(e1, b\_n, b\_m, ch2, fout);

delete[] d1;

}

else {

add\_matrices(a2, a\_n, a\_m, d2, b\_n, b\_m, e2);

print\_mat(e2, b\_n, b\_m, ch2, fout);

for (int i = 0; i < b\_n; i++)

delete[] d2[i];

delete[] d2;

}

if (ch2 == 1) fout << "(A + B / k)T\n";

else cout << "(A + B / k)T\n";

double\* f1 = new double[b\_m \* b\_n];

double\*\* f2 = new double\* [b\_m];

if (ch3 == 2) {

delete[] f1;

for (int i = 0; i < b\_m; i++) {

f2[i] = new double[b\_n];

}

}

else delete[] f2;

if (ch3 == 1) {

transp\_matr(e1, a\_n, a\_m, f1);

print\_mat(f1, b\_m, b\_n, ch2, fout);

delete[] e1;

}

else {

transp\_matr(e2, a\_n, a\_m, f2);

print\_mat(f2, b\_m, b\_n, ch2, fout);

for (int i = 0; i < b\_n; i++)

delete[] e2[i];

delete[] e2;

}

if (ch2 == 1) fout << "(A + B / k)T \* C\n";

else cout << "(A + B / k)T \* C\n";

if (!check\_mul\_sizes(b\_m, b\_n, c\_n, c\_m))

continue;

double\* g1 = new double[b\_m \* c\_m];

double\*\* g2 = new double\* [b\_m];

if (ch3 == 2) {

delete[] g1;

for (int i = 0; i < b\_m; i++) {

g2[i] = new double[c\_m];

for (int j = 0; j < c\_m; j++)

g2[i][j] = 0;

}

}

else {

delete[] g2;

for (int i = 0; i < b\_m; i++) {

for (int j = 0; j < c\_m; j++)

g1[i \* c\_m + j] = 0;

}

}

if (ch3 == 1) {

mul\_matr(f1, b\_m, b\_n, c1, c\_n, c\_m, g1);

print\_mat(g1, b\_m, c\_m, ch2, fout);

delete[] f1;

delete[] g1;

}

else {

mul\_matr(f2, b\_m, b\_n, c2, c\_n, c\_m, g2);

print\_mat(g2, b\_m, c\_m, ch2, fout);

for (int i = 0; i < b\_m; i++)

delete[] f2[i];

delete[] f2;

for (int i = 0; i < b\_m; i++)

delete[] g2[i];

delete[] g2;

}

if (ch3 == 1) { delete[] a1; delete[] b1; delete[] c1; }

else {

for (int i = 0; i < a\_n; i++)

delete[] a2[i];

delete[] a2;

for (int i = 0; i < b\_n; i++)

delete[] b2[i];

delete[] b2;

for (int i = 0; i < c\_n; i++)

delete[] c2[i];

delete[] c2;

}

break;

}

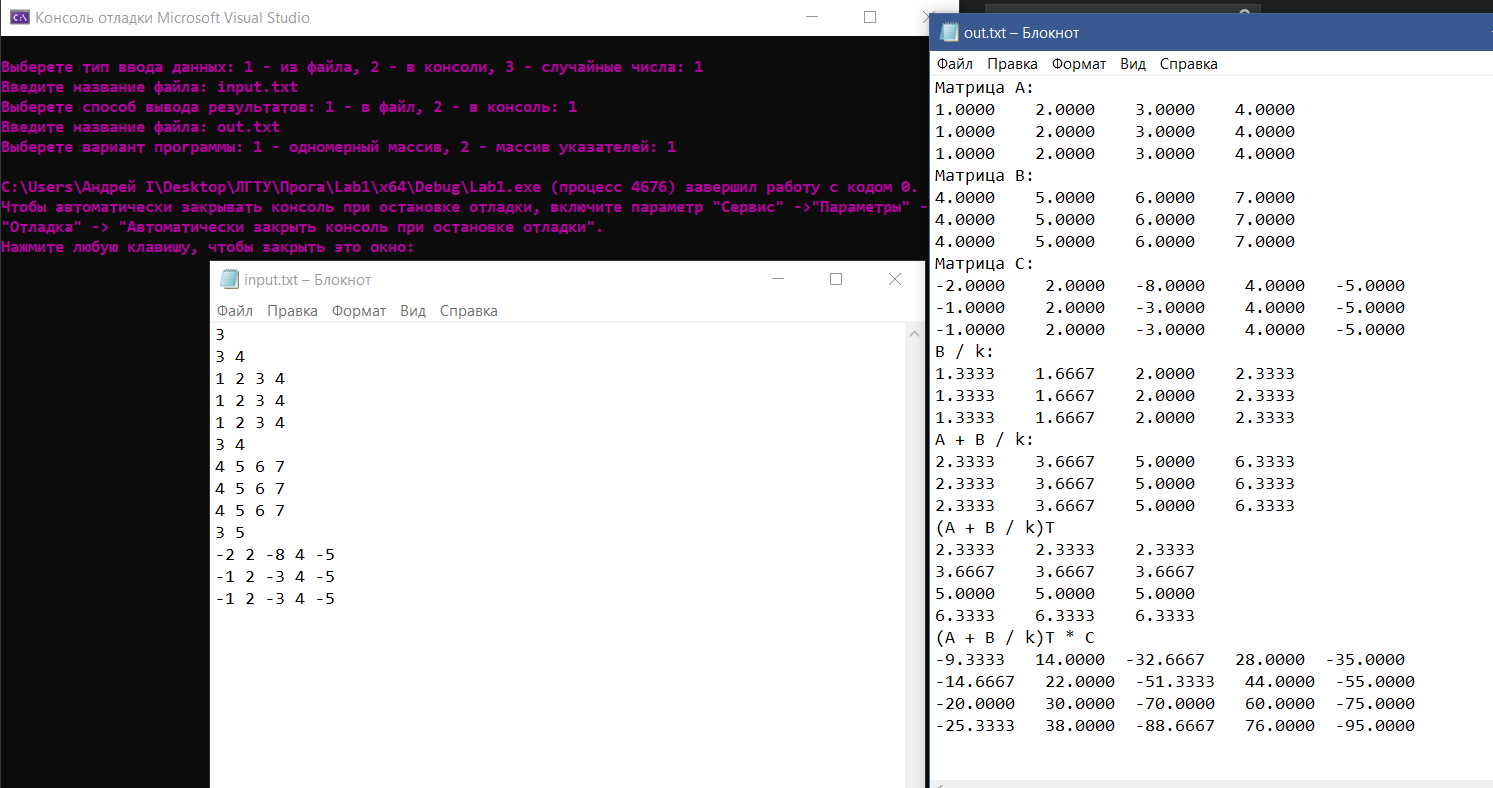
fin.close();

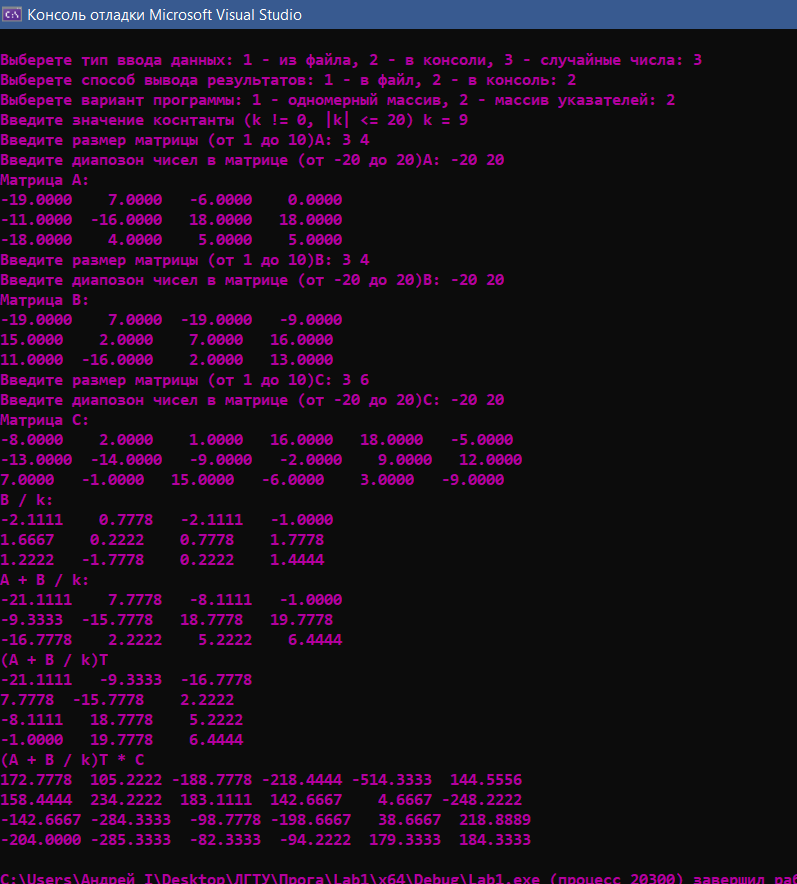
fout.close();

return 0;

}

**Контрольный пример**





**Вывод**

1. Одномерный массив лучше подходит для задач, где важна скорость и минимальное использование памяти, так как в нем все ячейки идут непрерывно.
2. Массив указателей удобен при работе с динамическими или нерегулярными данными, где требуется гибкость, может состоят из массив разных длин, и изменение одного из них остальные не затрагиваются.