МИНЕСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доц., канд. техн. наук |  |  |  | В. А. Галанина |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Программирование задач линейной алгебры. |
| по курсу: ИНФОРМАТИКА |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | М023 |  |  |  | Д.А.Трегуб |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2020

1. Описание задания:  
   19. Ввести с клавиатуры размерность и элементы квадратной матрицы *A(n,n)*. Матрица *B(n,n)* получена из матрицы **A**по формуле

*aij* 2 *, вне диагоналей*

*bij* = *,на главной диагонали*

*,на побочной диагонали*

Найти определитель D=∣**A⋅B**∣ .

Вывести на экран **B,** D

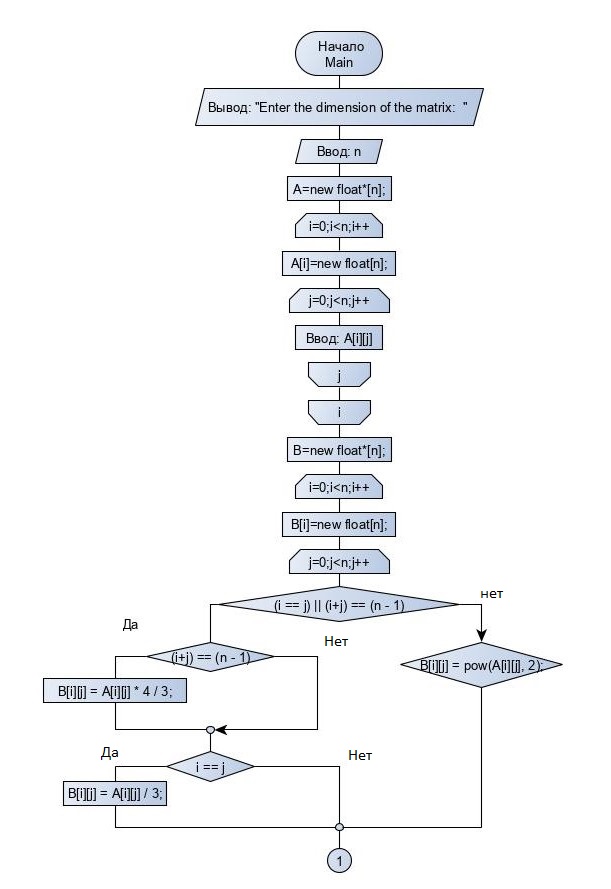
2. Математическая модель решения задачи:

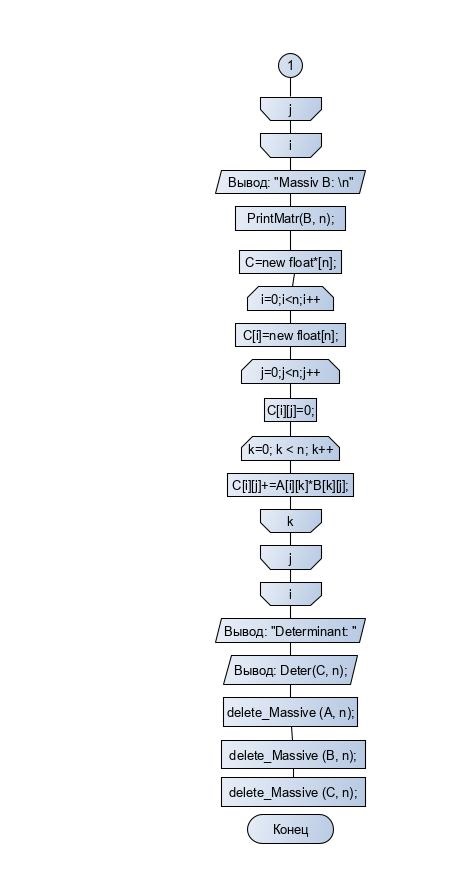
1. Массив B создаётся исходя из трёх условий. Условие для определения главной диагонали: I = j, где i – номер элемента в столбце, j – номер элемента в строке. Условие для определения побочной диагонали: I +j = n – 1, где n – количество строк в матрице.   
Остальные элементы находятся исключением этих двух условий.

2. В программе рассмотрено три способа нахождения определителя в зависимости от размерности матрицы. Для матрицы из одного элемента определитель будет равен её элементу. Для матрицы второго порядка вычисляется как разность произведения элементов главной и побочной диагонали. Для матрицы более высокого порядка существует рекурсивная формула: , где — дополнительный минор квадратной матрицы — определитель матрицы, полученной из исходной вычеркиванием i-ой строки и j-ого столбца.

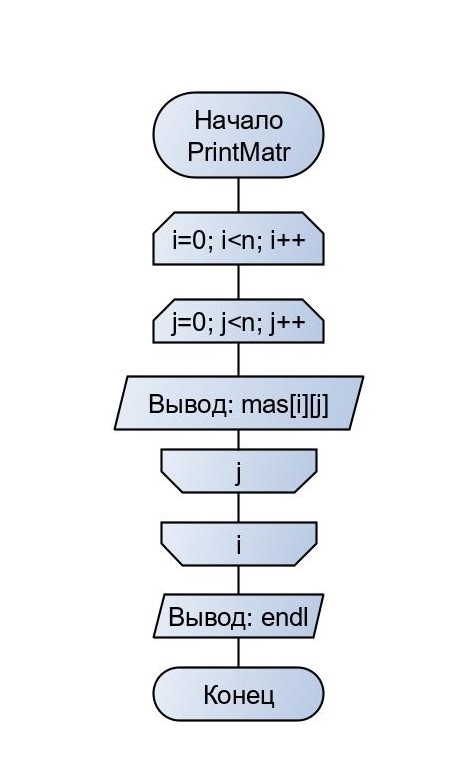
3. Алгоритм:

3.1. Основной алгоритм:

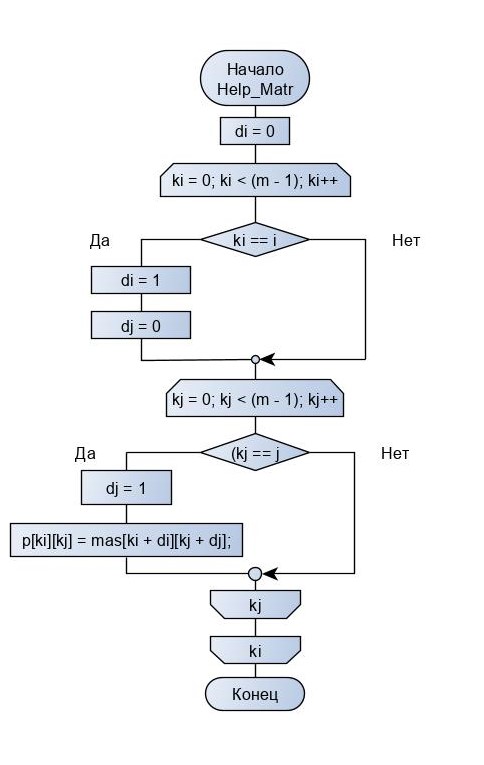




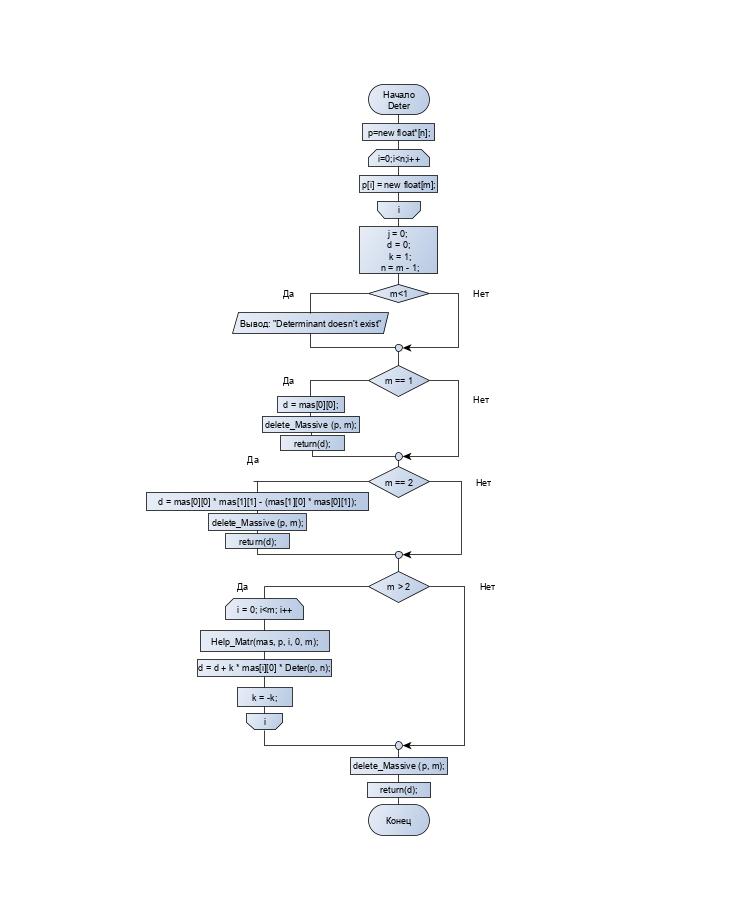
3.2. PrintMatr:



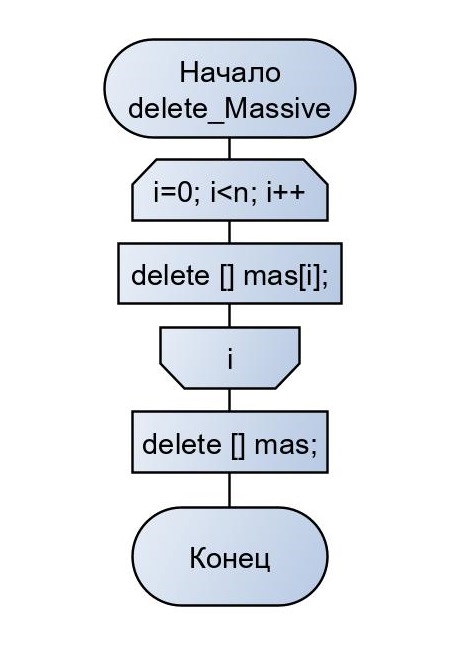
3.3. Help\_matr



3.4. Deter



3.5. Delete\_Massive



4. Программа:  
  
#include <iostream>

#include <math.h>

void delete\_Massive (float \*\*mas, int m);

void PrintMatr(float \*\*mas, int n);

void Help\_Matr(float \*\*mas, float \*\*p, int i, int j, int m);

float Deter(float \*\*mas, int m);

int main() {

int i, j, n, k, d;

float \*\*A; //массив А

std::cout<<"Enter the dimension of the matrix: ";

std::cin>>n;

A=new float\*[n];

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

A[i]=new float[n];

for(j=0;j<n;j++)

std::cin>>A[i][j];

}

float \*\*B; //Массив B ввод и вывод

B=new float\*[n];

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

B[i]=new float[n];

for(j=0;j<n;j++) {

if ((i == j) || (i+j) == (n - 1)){ //определение принадлежности диагоналям

if ((i+j) == (n - 1)) //побочная диагональ

B[i][j] = A[i][j] \* 4 / 3;

if (i == j) //главная диагональ

B[i][j] = A[i][j] / 3;

}

else // элементы вне диагоналей

B[i][j] = pow(A[i][j], 2);

}

}

std::cout<<"Massiv B: \n";

PrintMatr(B, n);

float \*\*C; //Массив с

C = new float\*[n];

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

C[i]=new float[n];

for(j=0;j<n;j++) {

C[i][j]=0;

for(k=0; k < n; k++)

C[i][j]+=A[i][k]\*B[k][j];

}

}

std::cout<<"Determinant: ";

std::cout<<Deter(C, n);

delete\_Massive (A, n);

delete\_Massive (B, n);

delete\_Massive (C, n);

return 0;

}

void delete\_Massive (float \*\*mas, int m){

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

delete [] mas[i];

}

delete [] mas;

}

void PrintMatr(float \*\*mas, int n) {

int i, j;

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

for (j=0; j<n; j++)

std::cout<<mas[i][j]<<" ";

std::cout<<std::endl;

}

}

void Help\_Matr(float \*\*mas, float \*\*p, int i, int j, int m) {

int ki, kj, di, dj;

di = 0;

for (ki = 0; ki < (m - 1); ki++) { // проверка индекса строки

if (ki == i) di = 1;

dj = 0;

for (kj = 0; kj < (m - 1); kj++) { // проверка индекса столбца

if (kj == j) dj = 1;

p[ki][kj] = mas[ki + di][kj + dj];

}

}

}

float Deter(float \*\*mas, int m) {

int i, j = 0, d = 0, k = 1, n;

float \*\*p;

p = new float\*[m];

for (i = 0; i<m; i++)

p[i] = new float[m];

j = 0; d = 0; k = 1; n = m - 1;

if (m<1) std::cout << "Determinant doesn't exist";

if (m == 1) {

d = mas[0][0];

delete\_Massive (p, m);

return(d);

}

if (m == 2) {

d = mas[0][0] \* mas[1][1] - (mas[1][0] \* mas[0][1]);

delete\_Massive (p, m);

return(d);

}

if (m>2) {

for (i = 0; i<m; i++) {

Help\_Matr(mas, p, i, 0, m);

d = d + k \* mas[i][0] \* Deter(p, n - 1);

k = -k;

}

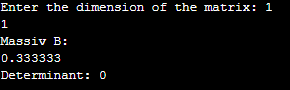
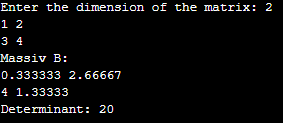
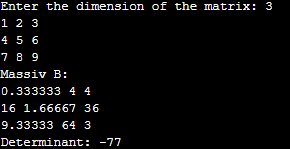
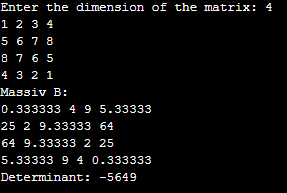
}

delete\_Massive (p, m);

return(d);

}

5.Скриншот результатов:


6. Выводы: В процессе работы над лабораторной я изучил работу динамических массивов в c++ и повторил основы программирования.