МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
“Национальный исследовательский университет ИТМО”

ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
Кафедра проектирования и безопасности компьютерных систем

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине

"ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА"

по теме:

«Сложение и умножение матриц»

Выполнил: Нгуен Ле Минь
Группа: N3251
Преподаватель: Гришенцев А.Ю.

Санкт-Петербург
2021

Задание

Разработать алгоритм и написать программу реализующую: ввод комплекснозначных матриц из консоли или из файла и следующие операции над матрицами, включая проверку реализуемости операции: сложение, вычитание, умножение на комплексное число, умножение матрицы на матрицу. Оценить вычислительную сложность различных операций.

Краткая теоретическая

+) **Сложение матриц** Суммой матриц A и B одного размера называется матрица $C = A + B$ такого же размера, получаемая из исходных путем сложения соответствующих элементов:

$$A_{m*n} + B_{m*n} = C_{m*n}; c_{ij} = a_{ij} + b_{ij}, i = \overline{1; m}, j = \overline{1; n}$$

+) **Вычитание матриц :** Разностью матрц A и B одного и того же размера называется матрица $C = A - B$ такого же размера, получаемая из исходных путем прибавления к матрице A матрицы B, умноженной на (-1).
Замечение : Складывать и вычитать можно только матрицы одинакового размера

Краткая теоретическая

+) **Произведение матрицы на число :** Пусть $A, C \in M_{m*n}$ и $\alpha \in \mathbb{R}$. Тогда : $C = \alpha * A \Leftrightarrow c_{ij} = \alpha * a_{ij}; i = \overline{1,m}, j = \overline{1,n}$.
Как видно, эта формула определяет умножение матрицы на число поэлементно: произведение A на (α на A) есть матрица C , элементы которой – это соответственные элементы A , умноженные на число α.
+) **Умножение матриц :** Пусть первый множитель A есть $m * n-$ матрица: $A \in M_{m*n}$, а второй множитель B – это $n * p-$ матрица: $B \in N_{n*p}$. Тогда элементы произведения матрицы A на матрицу B вычисляются следующим образом:

$$C = A.B \Leftrightarrow c_{ij} = \sum_{k=1}^{n} a_{ik} * b_{kj}, i = \overline{1,m}, j = \overline{1,p}$$

Как видно, произведение $A * B$ есть $m * p$ - матрица, число строк которой совпадает с числом строк первого множителя, а число столбцов – с числом столбцов второго множителя.

Алгоритмы

Сложение матриц : Сложение двух матриц A_{m*n} и B_{m*n} дает матрицу C_{m*n}. Алгоритм сложения матриц можно записать как:

```
for i in 1 to m
   for j in 1 to n
      c_ij = a_ij + b_ij
```

Сложность алгоритма операции сложения: $O(m*n)$, где $m*n$ - порядок матриц

Вычитание матриц : Вычитание двух матриц A_{m*n} и B_{m*n} дает матрицу C_{m*n}. Алгоритм вычитания матриц можно записать как:

```
for i in 1 to m
   for j in 1 to n
      c_ij = a_ij-b_ij
```

Сложность алгоритма операции вычитания: $O(m*n)$, где $m*n$ - порядок матриц

Алгоритмы

Умножение матриц : Умножение двух матриц A_{m*n} и B_{n*p} дает матрицу C_{m*p}. Это означает, что количество столбцов в A должно быть равно количеству строк в B, чтобы вычислить $C = A * B$. Алгоритм умножения матриц A с порядком $m * n$ и B с порядком $n * p$ можно записать как:

```
for i in 1 to m
   for j in 1 to p
      c_ij = 0
      for k in 1 to n
         c_ij += a_ik*b_kj
```

Сложность алгоритма операции сложения: Сложность операции умножения $(A * B)$ составляет $O(m * n * p)$, где $m * n$ и $n * p$ являются порядком A и B соответственно.

Листинг программы

```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int** new2DArray(const int m, const int n)
{
    int** arr = new int*[m];
    for (int i = 0; i < m; ++i)
        arr[i] = new int[n];
    for (int i = 0; i < m; ++i)
        for (int j = 0; j < n; ++j) {
            cout << "a[" << i << "][" << j << "]:";
            cin >> arr[i][j];
        }
    return arr;
}

void print2DArray(int** arr, const int m, const int n)
{
    for (int i = 0; i < m; ++i)
    {
        cout << "|";
        for (int j = 0; j < n; ++j)
            cout << std::setw( n: 4) << arr[i][j];
        cout << "|\n";
    }
}

void delete2DArray(int** arr, const int m)
{
    for (int i = 0; i < m; ++i)
        delete[] arr[i];
    delete[] arr;
}
```

```cpp
void sumMinusMatrix(int** mA, int** mB,
                    const int m, const int n) {
    int** mSum = new int*[m];
    int** mMinus = new int*[m];
    for (int i = 0; i < m; ++i){
        mSum[i] = new int[n];
        mMinus[i] = new int[n];
    }

    for (int i = 0; i < m; ++i)
        for (int j = 0; j < n; ++j) {
            mSum[i][j] = mA[i][j] + mB[i][j];
            mMinus[i][j] = mA[i][j] - mB[i][j];
        }
    cout << "A plus B:\n";
    print2DArray(mSum, m, n);
    cout << "A subtracts B :\n";
    print2DArray(mMinus, m, n);
    delete2DArray(mSum, m);
    delete2DArray(mMinus,m);
}
void mulNumMatrix(int** mA, const int m, const int n,const int num) {
    for (int i = 0; i < m; ++i) {
        for (int j = 0; j < n; ++j) {
            mA[i][j] *= num;
        }
    }
    print2DArray(mA, m, n);
}
```

Листинг программы

```
void matrixMul(int** mA, int** mB, const int m, const int n){
    if (m != n){
        return;
    }
    int** mMul = new int*[m];
    for (int i = 0; i < m; ++i){
        mMul[i] = new int[n];
        for (int j = 0; j < n; ++j){
            mMul[i][j] = 0;
            for (int k = 0; k < n; ++k){
                mMul[i][j] += mA[i][k] * mB[k][j];
            }
            }
        }
    print2DArray(mMul,m,n);
    delete2DArray(mMul,m);
}
```

```
int main()
{
    // вводим размеры матрицы
    cout << "Input m: ";
    int m;
    cin >> m;

    cout << "Input n: ";
    int n;
    cin >> n;

    cout << "Input matrix A!" << endl;
    int** mtrxA = new2DArray(m, n);
    cout << "Input matrix B!" << endl;
    int** mtrxB = new2DArray(m, n);
    print2DArray(mtrxA,m,n);
    print2DArray(mtrxB,m,n);
    cout << "Sum and Difference between matrix A and B" << endl;
    sumMinusMatrix(mtrxA, mtrxB, m, n);

    cout << "Result matrix A multiply with matrix B:\n";
    matrixMul(mtrxA,mtrxB,m,n);

    int num;
    cout << "Matrix A multiply with number:";
    cin >> num;
    cout << "Result matrix A multiply with number:\n";
    mulNumMatrix(mtrxA,m,n,num);


    // освобождаем выделенную память
    delete2DArray(mtrxA, m);
    delete2DArray(mtrxB, m);
    return 0;
}
```

Результаты программы

```
Input m: 3
Input n: 3
Input matrix A!
a[0][0]:2
a[0][1]:2
a[0][2]:322
a[1][0]:2323
a[1][1]:3
a[1][2]:33
a[2][0]:22
a[2][1]:1
a[2][2]:2
Input matrix B!
a[0][0]:3
a[0][1]:45
a[0][2]:6
a[1][0]:-9
a[1][1]:2
a[1][2]:31
a[2][0]:2
a[2][1]:3
a[2][2]:45
|         2         2       322|
|      2323         3        33|
|        22         1         2|
|         3        45         6|
|        -9         2        31|
|         2         3        45|
Sum and Difference between matrix A and B
A plus B:
|         5        47       328|
|      2314         5        64|
|        24         4        47|
A subtracts B :
|        -1       -43       316|
|      2332         1         2|
|        20        -2       -43|
```

```
Result matrix A multiply with matrix B:
|      632     1060    14564|
|     7008   104640    15516|
|       61      998      253|
Matrix A multiply with number:6
Result matrix A multiply with number:
|       12       12     1932|
|    13938       18      198|
|      132        6       12|
```

Вывод по работе

После этой лабораторной работы, я научился как написать программы с помощью языка C++ для реализирования операций над матрицами : сложения, вычитания, умножение на скаляр и умножение двух матрицы. И также познакомился с оценкой вычислительной сложность различных операций.

СПАСИБО ЗА ВНИМАНИЕ !