Longqing Chen ID: 22010057

I didn’t have time to finish the ARM part of the assignment, and will be working on it in the future. Here is what I have now (the comments will help clarify my codes)

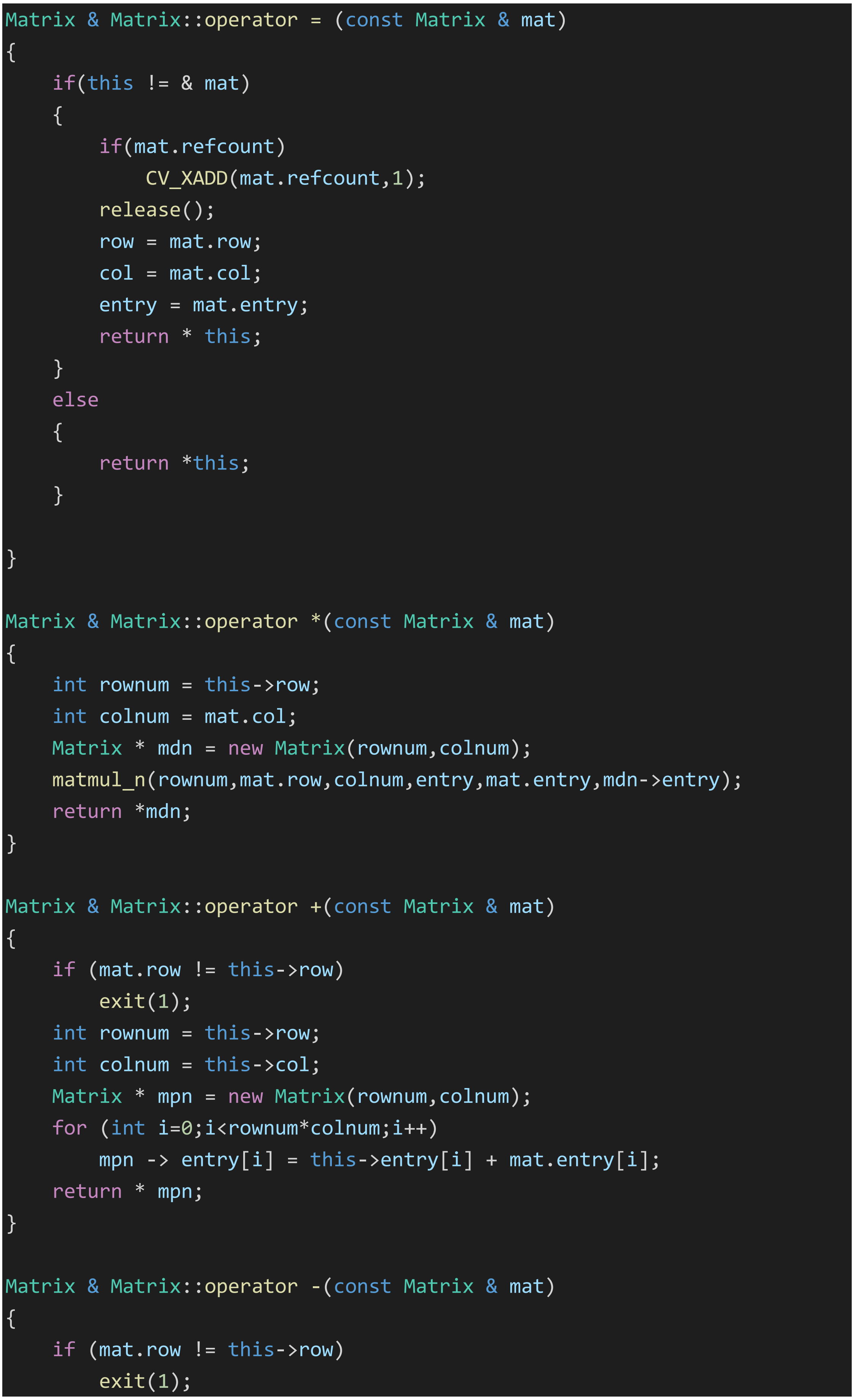
Codes:

matrix.hpp

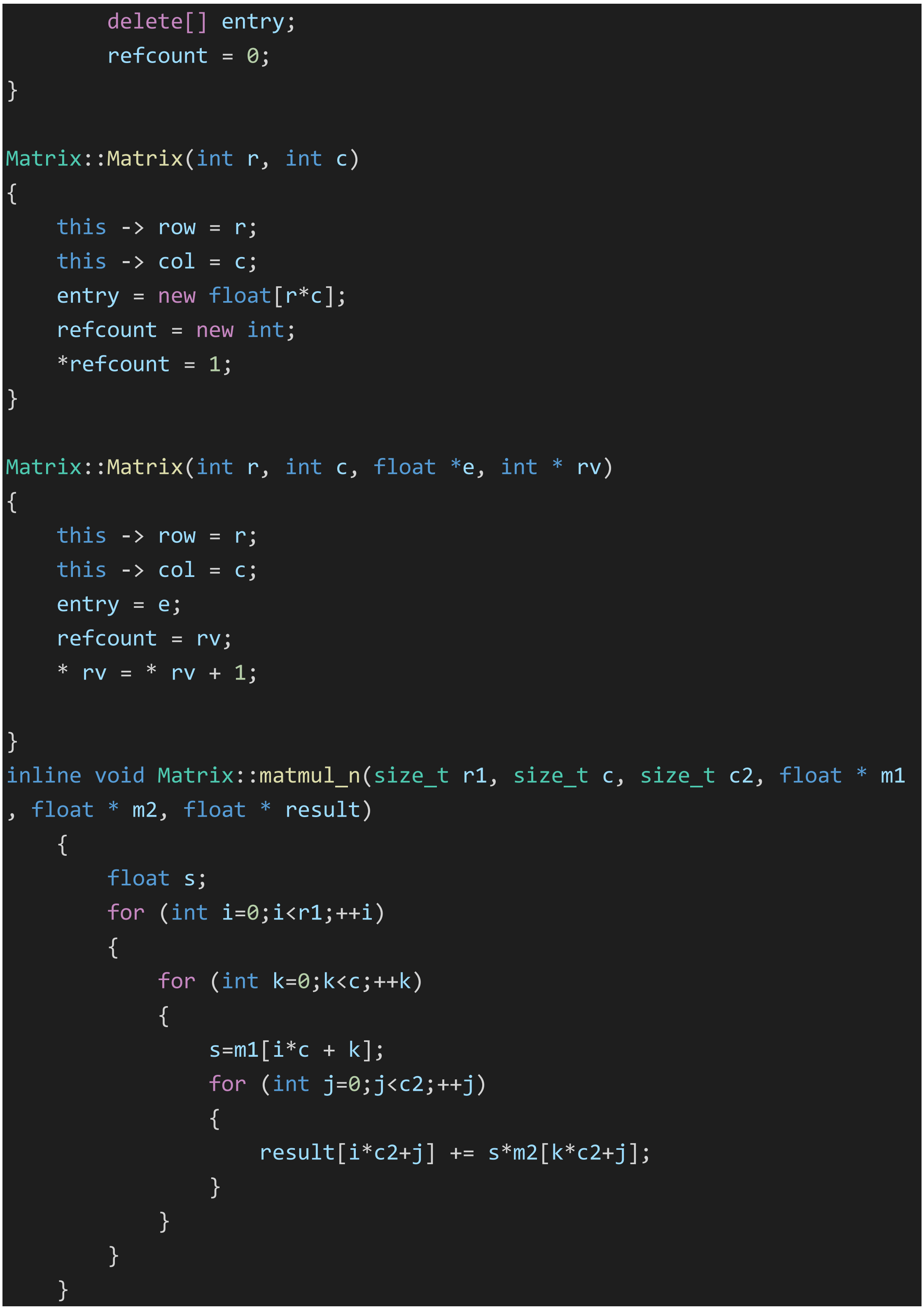
```
#ifndef MATRIX
#define MATRIX
#include <iostream>
using namespace std;

class Matrix
{
private:
    int row;
    int col;
    float *entry;
    int * refcount; //cv, counting the number of matrices
public:
    Matrix() {row = 1; col = 1; *entry = 1.0;}
    Matrix(int r, int c);
    Matrix(int r, int c, float *e, int * rv);
    Matrix(const Matrix &mat) {row = mat.row; col = mat.col; entry = ma
t.entry;} //copying the matrix
    ~Matrix() {release();}
    static inline int CV_XADD(int* addr, int delta){int tmp = * addr; *
 addr += delta; return tmp;} //CV_XADD
    static inline void matmul_n(size_t r1, size_t c, size_t c2, float *
 m1, float * m2, float * result); //matrix multiplication

    Matrix &operator = (const Matrix & mat);
    Matrix &operator + (const Matrix & mat);
    Matrix &operator - (const Matrix & mat);
    Matrix &operator * (const Matrix & mat);

    int GetRow() {return row;}
    int GetCol() {return col;}
    void release();
    friend std::ostream & operator <<(std::ostream &os, const Matrix &
other); //cout
    friend Matrix operator *(double real, const Matrix & mat);//scaling
    friend Matrix operator *(const Matrix & mat,double real);//also
scaling
};
#endif
```

matrix.cpp

```cpp
#include <iostream>
#include "matrix.hpp"
using namespace std;

int main()
{
    Matrix mat1(3,5);
    cout << "Matrix 1 is: ";
    cout << mat1;
    cout << endl;

    Matrix mat3 = mat1;
    cout << "Matrix 3 is the same as Matrix 1: ";
    cout << mat3;
    cout << endl;

    Matrix mat7 = mat3 + mat1;
    cout << "Matrix 3 + Matrix 1: ";
    cout << mat7;
    cout << endl;

    Matrix mat2(5,2);
    cout << "Matrix 2 is: ";
    cout << mat2;
    cout << endl;

    Matrix mat4 = mat1*mat2;
    cout << "Matrix 1 x Matrix 2: ";
    cout << mat4;
    cout << endl;

    Matrix mat5 = 2*mat1;
    cout << "Scale Matrix 1 by 2: ";
    cout << mat5;
    cout << endl;
    Matrix mat6 = mat1*3;
    cout << "Scale Matrix 1 by 3: ";
    cout << mat6;
    cout << endl;

    return 0;
}
```

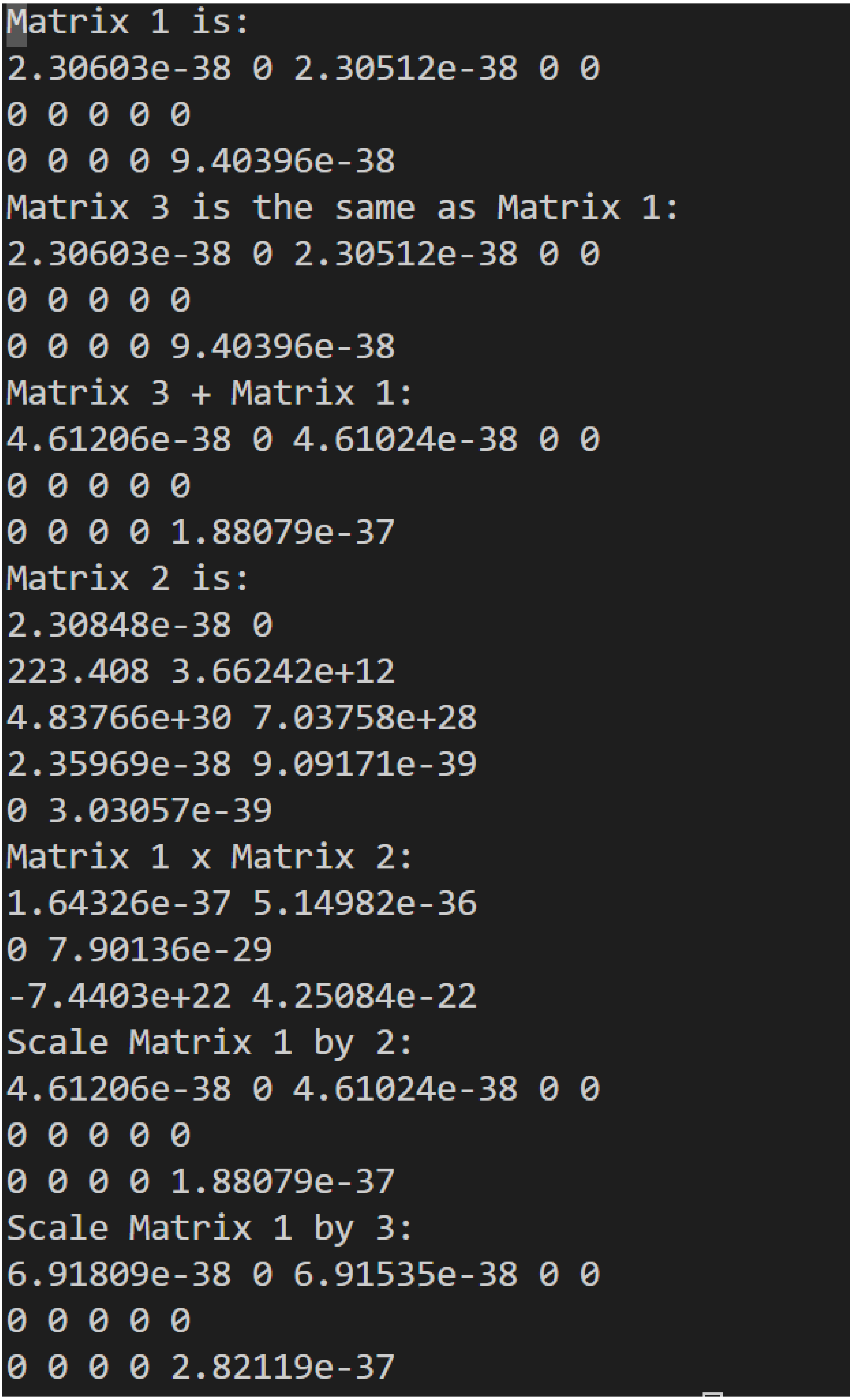


```
Matrix & Matrix::operator = (const Matrix & mat)
{
    if(this != & mat)
    {
        if(mat.refcount)
            CV_XADD(mat.refcount,1);
        release();
        row = mat.row;
        col = mat.col;
        entry = mat.entry;
        return * this;
    }
    else
    {
        return *this;
    }

}

Matrix & Matrix::operator *(const Matrix & mat)
{
    int rownum = this->row;
    int colnum = mat.col;
    Matrix * mdn = new Matrix(rownum,colnum);
    matmul_n(rownum,mat.row,colnum,entry,mat.entry,mdn->entry);
    return *mdn;
}

Matrix & Matrix::operator +(const Matrix & mat)
{
    if (mat.row != this->row)
        exit(1);
    int rownum = this->row;
    int colnum = this->col;
    Matrix * mpn = new Matrix(rownum,colnum);
    for (int i=0;i<rownum*colnum;i++)
        mpn -> entry[i] = this->entry[i] + mat.entry[i];
    return * mpn;
}

Matrix & Matrix::operator -(const Matrix & mat)
{
    if (mat.row != this->row)
        exit(1);
    int rownum = this->row;
    int colnum = this->col;
    Matrix * mpn = new Matrix(rownum,colnum);
    for (int i=0;i<rownum*colnum;i++)
        mpn -> entry[i] = this->entry[i] - mat.entry[i];
    return * mpn;
}

Matrix operator * (double real, const Matrix & other)
{
    int rownum = other.row;
    int colnum = other.col;
    Matrix * mxn = new Matrix (rownum,colnum);
    for(int i=0;i<rownum*colnum;i++)
        mxn -> entry[i] = real*other.entry[i];
    return * mxn;
}

Matrix operator * (const Matrix & other,double real)
{
    int rownum = other.row;
    int colnum = other.col; //I omit the steps that examine whether the
multiplication would produce errors.
    Matrix * mxn = new Matrix (rownum,colnum);
    for(int i=0;i<rownum*colnum;i++)
        mxn -> entry[i] = real*other.entry[i];
    return * mxn;
}

std::ostream & operator << (std::ostream & os, const Matrix & mat)
{
    using namespace std;
    for(int r=0;r<mat.row;r++)
    {
        os << endl;
        for(int c=0;c<mat.col;c++)
            os<<mat.entry[r * mat.row + c] << " ";
    }
    return os;
}

void Matrix::release()
{
    if (refcount && CV_XADD(refcount,-1) ==1)
```



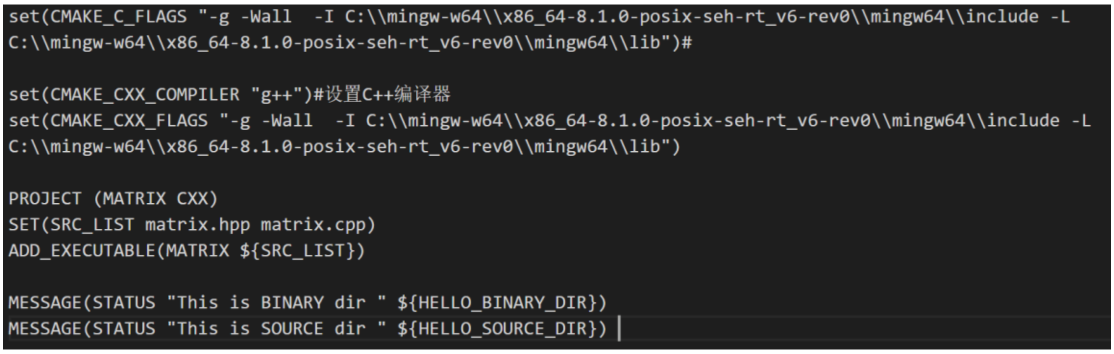
```
        delete[] entry;
        refcount = 0;
}

Matrix::Matrix(int r, int c)
{
    this -> row = r;
    this -> col = c;
    entry = new float[r*c];
    refcount = new int;
    *refcount = 1;
}

Matrix::Matrix(int r, int c, float *e, int * rv)
{
    this -> row = r;
    this -> col = c;
    entry = e;
    refcount = rv;
    * rv = * rv + 1;

}
inline void Matrix::matmul_n(size_t r1, size_t c, size_t c2, float * m1
, float * m2, float * result)
    {
        float s;
        for (int i=0;i<r1;++i)
        {
            for (int k=0;k<c;++k)
            {
                s=m1[i*c + k];
                for (int j=0;j<c2;++j)
                {
                    result[i*c2+j] += s*m2[k*c2+j];
                }
            }
        }
    }
```

Here are the results:



```
Matrix 1 is:
2.30603e-38 0 2.30512e-38 0 0
0 0 0 0 0
0 0 0 0 9.40396e-38
Matrix 3 is the same as Matrix 1:
2.30603e-38 0 2.30512e-38 0 0
0 0 0 0 0
0 0 0 0 9.40396e-38
Matrix 3 + Matrix 1:
4.61206e-38 0 4.61024e-38 0 0
0 0 0 0 0
0 0 0 0 1.88079e-37
Matrix 2 is:
2.30848e-38 0
223.408 3.66242e+12
4.83766e+30 7.03758e+28
2.35969e-38 9.09171e-39
0 3.03057e-39
Matrix 1 x Matrix 2:
1.64326e-37 5.14982e-36
0 7.90136e-29
-7.4403e+22 4.25084e-22
Scale Matrix 1 by 2:
4.61206e-38 0 4.61024e-38 0 0
0 0 0 0 0
0 0 0 0 1.88079e-37
Scale Matrix 1 by 3:
6.91809e-38 0 6.91535e-38 0 0
0 0 0 0 0
0 0 0 0 2.82119e-37
```

This is my CMakeLists file:



```
set(CMAKE_C_FLAGS "-g -Wall  -I C:\\mingw-w64\\x86_64-8.1.0-posix-seh-rt_v6-rev0\\mingw64\\include -L
C:\\mingw-w64\\x86_64-8.1.0-posix-seh-rt_v6-rev0\\mingw64\\lib")#

set(CMAKE_CXX_COMPILER "g++")#设置C++编译器
set(CMAKE_CXX_FLAGS "-g -Wall  -I C:\\mingw-w64\\x86_64-8.1.0-posix-seh-rt_v6-rev0\\mingw64\\include -L
C:\\mingw-w64\\x86_64-8.1.0-posix-seh-rt_v6-rev0\\mingw64\\lib")

PROJECT (MATRIX CXX)
SET(SRC_LIST matrix.hpp matrix.cpp)
ADD_EXECUTABLE(MATRIX ${SRC_LIST})

MESSAGE(STATUS "This is BINARY dir " ${HELLO_BINARY_DIR})
MESSAGE(STATUS "This is SOURCE dir " ${HELLO_SOURCE_DIR})
```

I executed the following commands in CMD;

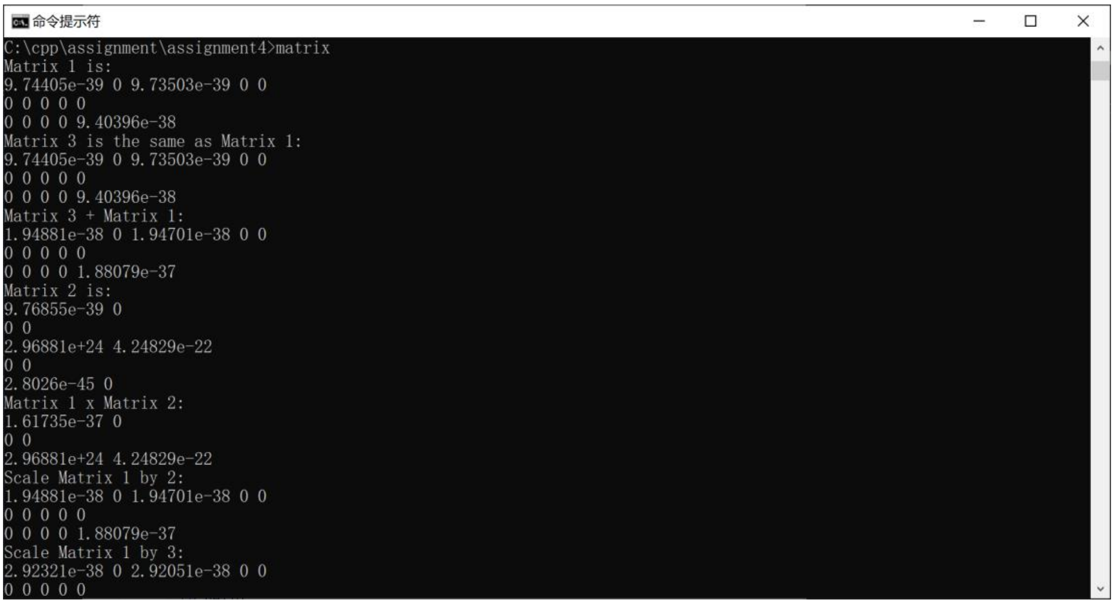
`cmake -G “MinGW Makefiles” .`

`make`

And run the program

`Matrix`

And got the same result:



```
C:\cpp\assignment\assignment4>matrix
Matrix 1 is:
9.74405e-39 0 9.73503e-39 0 0
0 0 0 0 0
0 0 0 0 9.40396e-38
Matrix 3 is the same as Matrix 1:
9.74405e-39 0 9.73503e-39 0 0
0 0 0 0 0
0 0 0 0 9.40396e-38
Matrix 3 + Matrix 1:
1.94881e-38 0 1.94701e-38 0 0
0 0 0 0 0
0 0 0 0 1.88079e-37
Matrix 2 is:
9.76855e-39 0
0 0
2.96881e+24 4.24829e-22
0 0
2.8026e-45 0
Matrix 1 x Matrix 2:
1.61735e-37 0
0 0
2.96881e+24 4.24829e-22
Scale Matrix 1 by 2:
1.94881e-38 0 1.94701e-38 0 0
0 0 0 0 0
0 0 0 0 1.88079e-37
Scale Matrix 1 by 3:
2.92321e-38 0 2.92051e-38 0 0
0 0 0 0 0
```

This is the link to my GitHub where I host this assignment:
https://github.com/lounachen/cpp/tree/master/assignment/assignment4