算符重载 上机练习

李宇豪 21305412

任务：

重载*（乘法）算法，实现矩阵与矩阵、矩阵与实数的乘法运算。

思路：

由于* 算法的两个操作数不一定全为矩阵，所以要分情况处理：

Case 1：实数乘矩阵，*先与实数结合，由于实数的*运算不能与矩阵直接结合，且为避免重载 double 类型的*算符，建议通过友元函数实现*的重载；需要传入两个参数

Case 2：矩阵乘实数，可以用与 Case 1 同样的处理方法；

Case 3：矩阵乘矩阵，可直接将*重载为成员数，此时仅需一个参数，另一个参数为本对象（this）

源代码：

1. 头文件中的相关声明：

```
//operator overload
friend Matrix operator*(const Matrix m,double d);
friend Matrix operator*(double d, const Matrix m);
Matrix operator*(const Matrix m);
```

2. 乘法算符（*）重载的实现代码：

```
// 矩阵×实数
Matrix operator*(const Matrix m,double d)
{
   Matrix m2(m);
    for (int i = 0; i < m2.row; i++)
        for (int j = 0; j < m2.col; j++)
            m2.data(i, j) = d * m.getdata(i, j);
   return m2;
}

// 实数×矩阵
Matrix operator*(double d,const Matrix m)
{
   Matrix m2(m);
    for (int i = 0; i < m2.row; i++)
        for (int j = 0; j < m2.col; j++)
            m2.data(i, j) = d * m.getdata(i, j);
   return m2;
}



// 矩阵×矩阵
Matrix Matrix::operator*(const Matrix m)
{
   if (col != m.row) // 判断是否符合矩阵相乘条件
      cerr << "this matrix.col != _ma.row" << endl;
```

```
    int rownew = row, colnew = m.col;
    Matrix c(rownew, colnew);
     for(int i = 0; i < row ; i++)
     {
         for(int j = 0; j < m.col ; j++)
         {
             double sum = 0;
             for(int k = 0; k < col; k++)
             {
                 sum += p_data[i * col + k] * m.p_data[k * m.col + j];
             }
             c.data(i,j) = sum;
         }
     }
    return c;
}
```

3. main.cpp 中 main 函数测试代码：

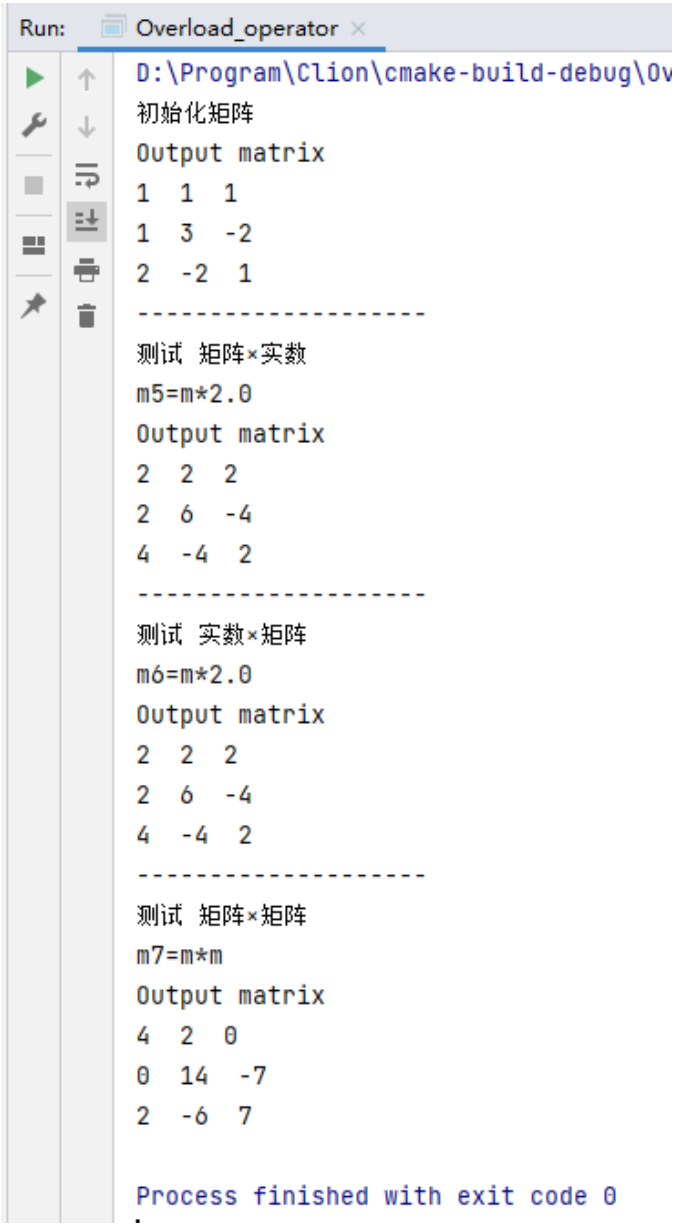
```
int main()
{
    int n = 3;
    cout << "初始化矩阵" << endl;
    double A[3][3] = { {1.,1., 1.},{1.,3.,-2.},{2.,-2.,1.} },
           b[] = { 6., 1., 1. };
    Matrix  m(n,n);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            m.data(i, j) = A[i][j];
    m.print();

    cout << "---------------------" << endl;
    cout << "测试 矩阵×实数" << endl;
    Matrix m5=m*2.0;
    cout<<"m5=m*2.0"<<endl;
    m5.print();

    cout << "---------------------" << endl;
    cout << "测试 实数×矩阵" << endl;
    Matrix m6=2.0*m;
    cout<<"m6=m*2.0"<<endl;
    m6.print();

    cout << "---------------------" << endl;
    cout << "测试 矩阵×矩阵" << endl;
    Matrix m7=m*m;
    cout<<"m7=m*m"<<endl;
    m7.print();
    return 0;
}
```

运行结果：


Run:
Overload_operator
D:\Program\Clion\cmake-build-debug\Ov
初始化矩阵
Output matrix
1 1 1
1 3 -2
2 -2 1

测试 矩阵*实数
m5=m*2.0
Output matrix
2 2 2
2 6 -4
4 -4 2

测试 实数*矩阵
m6=m*2.0
Output matrix
2 2 2
2 6 -4
4 -4 2

测试 矩阵*矩阵
m7=m*m
Output matrix
4 2 0
0 14 -7
2 -6 7
Process finished with exit code 0