



河南工业大学

《算法设计与分析》综合性实验 实验报告

题 目： 矩阵乘法

姓 名： 田劲锋

班 级： 软件 1305 班

学 号： 201316920311

指导教师： 靳小波

完成时间： 2014 年 11 月 21 日

一、 实验题目

矩阵乘法

二、 实验目的

1. 掌握二维数组的使用方法。
2. 掌握文件的读入和输出。
3. 掌握复杂循环的设计方法。

三、 实验要求

任给两个矩阵 A 和 B ，计算它们的乘法。

具体要求：

1 每个矩阵输入的格式如下：第一行的两个数字分别表示行 n 和列 m ，随后紧跟包含 m 列的 n 行来表示矩阵。将两个输入矩阵保存在 input.txt 文件中。举例如下：

```
3 3
1 2 3
4 5 6
7 8 9
3 1
1
1
1
```

2 将两个矩阵的运算结果保存在文件 output.txt 中。

四、 程序流程图

设 $A = (a_{ij})_{m \times s}$, $B = (b_{ij})_{s \times n}$ ，矩阵乘法定义为

$$C = A \times B,$$

其中

$$\mathbf{C} = (c_{ij})_{m \times n},$$

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{is}b_{sj} = \sum_{k=1}^s a_{ik}b_{kj}$$

可以得出矩阵乘法的算法，伪代码如图1。

```

MATRIX::×(A, B)
1  C.l = A.l
2  C.c = B.c
3  for i = 1 to A.l
4      for j = 1 to B.c
5          for k = 1 to A.c(= B.l)
6              Cij = Aik × Bkj
7  return C

```

图 1: 矩阵乘法的算法

二维数组的输入输出则是比较简单的，流程图2描述了二维数组的输入过程。

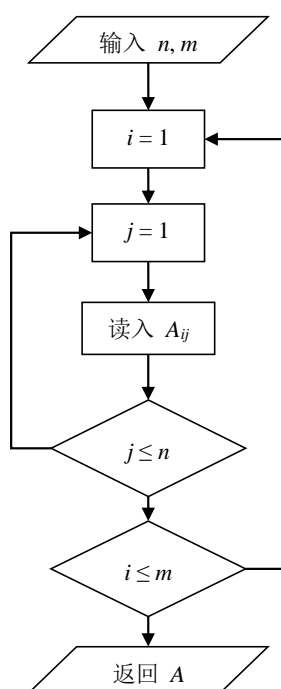


图 2: 矩阵输入过程

输出同理，不再赘述。

五、 程序代码

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4
5  const int MAXN = 128;
6
7  class Matrix
8  {
9  public:
10     int l; // 行 (下标) 1
11     int c; // 列 (下标) 2
12     int A[MAXN][MAXN];
13
14     Matrix(int m, int n)
15     {
16         c = m;
17         l = n;
18         for (int i = 0; i < l; i++) {
19             for (int j = 0; j < c; j++) {
20                 A[i][j] = 0;
21             }
22         }
23     }
24
25     void readin()
26     {
27         for (int i = 0; i < l; i++) {
28             for (int j = 0; j < c; j++) {
29                 cin >> A[i][j];
30             }
31         }
32     }
33
34     void writeout()
35     {
36         for (int i = 0; i < l; i++) {
37             for (int j = 0; j < c; j++) {
38                 cout << setw(4) << A[i][j];
39             }
40             cout << endl;
41         }
42     }
43
44     Matrix operator*(Matrix B)
45     {
46         Matrix C(B.c, l);
47         for (int i = 0; i < l; i++) {
48             for (int j = 0; j < B.c; j++) {
49                 for (int k = 0; k < c; k++) {
50                     C.A[i][j] += A[i][k] * B.A[k][j];
51                 }
52             }
53         }
```

```

53         }
54         return C;
55     }
56 };
57
58 int main(int argc, char **argv)
59 {
60     freopen("input.txt", "r", stdin);
61     freopen("output.txt", "w", stdout);
62
63     int n, m;
64     cin >> n >> m;
65     Matrix A(m, n);
66     A.readin();
67
68     cin >> n >> m;
69     Matrix B(m, n);
70     B.readin();
71
72     if (A.c != B.l) {
73         cerr << "不可相乘! " << endl;
74         return -1;
75     }
76
77     Matrix C = A * B;
78     C.writeout();
79
80     return 0;
81 }

```

六、 实验结果

输出文件output.txt:

```

6
15
24

```

实际上该样例即是此式:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 15 \\ 24 \end{bmatrix}$$

七、 实验体会

这道题目比较适合用于练手，虽然简单但是也有一些细节需要仔细处理。结合最近学习的C++课程，我使用了简单的类来实现这个矩阵乘法的题目。实际上这个程序还不能很好地表现OOP的精髓，比如输入输出逻辑绑定在了类中，应该另外做一个接口的。文件的输入输出则简单的使用了文件重定向，也可在命令行下使用重定向符来重定向输入输出流。

关于程序流程图的问题，我觉得在一定程度上，这个流程图是个累赘，描述算法还是伪代码比较合适。

本来打算用Word随便做做交了的，想到老师实验课上严重的 $\text{T}_{\text{E}}\text{X}$ 痕迹，那就换成 $\text{T}_{\text{E}}\text{X}$ 来排版了。虽然是驾轻就熟的工具，但是改改模板还是费点心思的。