# 三元组实验报告

**班级：2015211306 姓名：李航 学号：2015211284**

**班级：2015211306 姓名：魏晓 学号：2015211301**

**由于代码量不大，我们的代码都是各自独立编写。**

1. **需求分析**

**要用三元组来实现矩阵的乘法和加法，输入稀疏矩阵是通过键盘来实现，要判断矩阵不能相加相乘的原因输入是两个数组，一个是数组A，一个是数组B输出是他们的和和他们的积，如果他们能相称的话。程序所能达到的功能就是计算矩阵，正确的输入就是依次输入他们的行，列，值，最后用-1结束。**

1. **概要设计**

**要实现矩阵的加和乘，首先要能正确的接受信息，然后能成功的转化为答案就需要正确的函数，函数的输入格式，输出格式要正确**

**typedef struct{**

**int i,j;//非零元素的行下标和列下表**

**int num;//操作对象**

**}triple;**

**typedef struct{**

**triple data[Max];**

**int mu,nu,tu;**

**}matrix;**

**这就是用到的数据的结构，用一个结构存储三元组，另一个结构存储整个数组首先输入，然后判断，然后计算，然后输出，最后释放空间**

1. **祥细设计**

**这个程序设计中，要注意数组的边界。用到几个函数**

**void mullt(matrix p,matrix q,matrix &t)**

**{**

**hangyouxian(p);**

**lieyouxian(q);**

**triple ansp,anwq;**

**int sun=0;**

**int qw=1;**

**for(int a=1;a<=p.mu;a++)//控制p行变化**

**{**

**for(int b=1;b<=q.nu;b++)//控制q列变化**

**{**

**for(int c=1;c<=p.nu;c++)**

**{**

**ansp=searh(p,a,c);**

**anwq=searh(q,c,b);**

**if(ansp.i==1&&anwq.i==1)**

**{**

**sun=sun+p.data[ansp.num].num\*q.data[anwq.num].num;**

**}**

**}**

**if(sun!=0)**

**{**

**t.data[qw].i=a;**

**t.data[qw].j=b;**

**t.data[qw].num=sun;**

**qw++;**

**}**

**sun=0;**

**}**

**}**

**t.tu=qw-1;**

**}这个函数实现乘法运算**

**void hangyouxian(matrix &q)**

**{**

**triple tex;**

**for(int a=1;a<q.tu;a++)**

**{**

**for(int b=1;b<q.tu;b++)**

**{**

**if(q.data[b].i>q.data[b+1].i)**

**{**

**tex.i=q.data[b].i;**

**q.data[b].i=q.data[b+1].i;**

**q.data[b+1].i=tex.i;**

**tex.j=q.data[b].j;**

**q.data[b].j=q.data[b+1].j;**

**q.data[b+1].j=tex.j;**

**tex.num=q.data[b].num;**

**q.data[b].num=q.data[b+1].num;**

**q.data[b+1].num=tex.num;**

**}**

**}**

**}**

**}把一个普通的矩阵转化为行优先的矩阵**

**triple searh(matrix p,int x,int y)**

**{**

**triple wer;**

**wer.i=0;**

**wer.j=0;**

**wer.num=0;**

**for(int a=1;a<=p.tu;a++)**

**{**

**if(p.data[a].i==x&&p.data[a].j==y)**

**{**

**wer.i=1;**

**wer.j=1;**

**wer.num=a;**

**return wer;**

**}**

**}**

**return wer;**

**}这是在矩阵中搜寻需要的一行数的**

**void display(matrix p)**

**{**

**cout<<"the matrix is :"<<endl;**

**for(int a=1;a<=p.tu;a++)**

**{**

**cout<<p.data[a].i<<" ";**

**cout<<p.data[a].j<<" ";**

**cout<<p.data[a].num<<endl;**

**}**

**cout<<"display function have finish perfectly!"<<endl;**

**}**

**这**

**是把三元组输出的矩阵**

**void inpput(matrix &q)**

**{**

**int a=0,b=0,c=0;**

**int d,f,g;**

**cin>>d;**

**while(d!=-1)**

**{**

**c++;**

**q.data[c].i=d;**

**cin>>f;**

**q.data[c].j=f;**

**cin>>g;**

**q.data[c].num=g;**

**cin>>d;**

**}**

**q.tu=c;**

**for(int h=1;h<=c;h++)**

**{**

**if(q.data[h].i>a)**

**a=q.data[h].i;**

**}**

**q.mu=a;**

**for(int m=1;m<=c;m++)**

**{**

**if(q.data[m].j>b)**

**b=q.data[m].j;**

**}**

**q.nu=b;**

**}这是输入三元组的函数**

1. **调试分析报告**

**这个函数在调试的过程中主要问题是看清楚每一层循环的边界条件，有时候产生的新矩阵总是没有把大小保存下来，所以经常内存泄漏，算法的时间复杂度是O（n^5）的，空间复杂度是Ω（n2）的**

1. **用户使用说明**

**按照行，列，值的顺序依次输入，用-1结束，分别输入即可**

1. **测试结果**



