数据结构上机报告

题目：稀疏矩阵运算器

班级：04班 姓名： 学号： 完成日期：2020.11.19

一、需求分析

1.问题描述

稀疏矩阵是指那些多数元素为零的矩阵。利用“稀疏”特点进行元素存储和计算可以大大节省存储空间，提高计算效率。实现一个能进行稀疏矩阵基本运算的运算器。

1. 基本要求

以“带行逻辑链接信息”的三元组顺序表表示稀疏矩阵，实现两个矩阵相加、相减和相乘的运算。稀疏矩阵的输入形式采用三元组表示，而运算结果的矩阵则以通常的阵列形式列出。

3.程序功能

对给定参与运算的矩阵数据进行运算处理，以阵列形式输出经过运算的矩阵。

二、概要设计:

程序用结构体表示三元组，利用二维数组保存计算数据并输出。根据用户输入的运算符判定运算类型后输入稀疏矩阵，在输入过程中（加减运算）或过程后（乘除运算）进行结果计算，并储存在数组中。

1. 三元组结构体类型

typedef struct datagroup {

int m = 0; //稀疏矩阵的行数

int n = 0; //稀疏矩阵的列数

int num; //矩阵对应的元素值

}Dg;

3、主程序流程

a.输入矩阵的行列信息和运算方式，对不能进行运算的矩阵数据作报错处理。

b.以三元组的形式输入矩阵，每个矩阵的输入以0结束，在输入过程中根据给定的运算方式做数据处理，将结果保存在二元数组a中。

c.以阵列形式输出矩阵a

三、详细设计

主程序数据处理部分：

1. 加法运算

if(m1 != m2 || n1 != n2) { //输入检查

printf("输入错误，请检查矩阵的行列输入是否正确");

break;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

a:scanf("%d",&t1[0].m); //锚定点，利用goto实现循环输入

if(t1[0].m == 0 && temp == 1) break; //输入完毕检查点,temp用于标记第二个矩阵

else {

if(t1[0].m == 0 && temp == 0) {

temp++; //遇到0进行输入矩阵转换

goto a;

}

scanf("%d %d",&t1[0].n,&t1[0].num); //输入矩阵的列数和元素值

a[t1[0].m][t1[0].n] += t1[0].num; //将运算结果保存在a中

goto a;

1. 减法运算

if(m1 != m2 || n1 != n2) { //输入检查

printf("输入错误，请检查矩阵的行列输入是否正确");

break;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

b:scanf("%d",&t1[0].m); //锚定点，利用goto实现循环输入

if(t1[0].m == 0 && temp == 1) break; //输入完毕检查点,temp用于标记第二个矩阵

else {

if(t1[0].m == 0 && temp == 0) {

temp++; //遇到0进行输入矩阵转换

goto b;

}

scanf("%d %d",&t1[0].n,&t1[0].num); //输入矩阵的列数和元素值

if(temp == 0) a[t1[0].m][t1[0].n] += t1[0].num; //根据矩阵输入的顺序保存运算结果

else a[t1[0].m][t1[0].n] -= t1[0].num;

goto b;

1. 乘法运算

if(n1 != m2) {

printf("输入错误，请检查矩阵的行列输入是否正确");

goto e;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

c:if(temp == 0) scanf("%d",&t1[i].m);

else scanf("%d",&t2[i].m);

if(temp == 1 && t2[i].m == 0) { //若输入完毕，则进行运算

for(i = 0;i < t;i++) {

for(j = 0;j < k;j++) {

if(t1[i].n == t2[j].m) //按照三元组顺序进行乘法运算

a[t1[i].m][t2[j].n] += t1[i].num \* t2[j].num;

}

}

break;

}

else {

if(t1[i].m == 0 && temp == 0) { //利用temp进行两个矩阵的转换

temp++;

i = 0;

goto c;

}

if(temp == 0) { //输入矩阵1

scanf("%d %d",&t1[i].n,&t1[i].num);

i++;

t++;

}

else { //输入矩阵2

scanf("%d %d",&t2[i].n,&t2[i].num);

i++;

k++;

}

goto c;

四、调试分析

1、调试问题

a.初期实现程序时，未对乘法矩阵的输入数据进行存储处理，仅想依靠a数组运算，后发现可行性较低，改为将乘法三元组矩阵保存，并在输入后运算，其余运算均为实时运算。

b.未进行输入数据计数导致后期调试出错，发现后更正

1. 算法的时间复杂度分析和改进设想

算法的时间复杂度为O（m\*n），m、n为生成矩阵的行列数，时间复杂度主要由输出矩阵部分产生，故无法优化。程序还存在健壮性较差，支持的矩阵运算方法较少等问题。后期改进应加强意外输入处理、并补充求逆运算等计算模块。

1. 经验和体会

对数组的存储、运算，以及数组内部数据的处理有了进一步了解。

五、用户使用说明

用户只需运行所给文件夹下exe程序，按照文字提示输入矩阵数据，即可得到输岀结果。

六、测试结果

输入样例

请输入矩阵的运算方式(+、-、\*)：+

请输入两个矩阵的行数和列数：3 3 3 3

请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:

1 1 10

2 3 9

3 1 -1

0

2 3 -1

3 1 1

3 3 -3

0

输出样例

10 0 0

0 0 8

0 0 -3

输入样例

请输入矩阵的运算方式(+、-、\*)：-

请输入两个矩阵的行数和列数：3 2 3 2

请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:

1 1 10

2 2 9

3 1 -1

0

2 2 -1

3 1 1

3 2 -3

0

输出样例

10 0

0 10

-2 3

输入样例

请输入矩阵的运算方式(+、-、\*)：\*

请输入两个矩阵的行数和列数：4 5 5 3

请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:

1 1 4

1 2 -3

1 5 1

2 4 8

3 3 1

4 5 70

0

1 1 3

2 1 4

2 2 2

3 2 1

4 1 1

0

输出样例

0 -6 0

8 0 0

0 1 0

0 0 0

1. 附录

#include<stdio.h>

typedef struct datagroup {

int m = 0;

int n = 0;

int num;

}Dg;

int main() {

char c;

int m1,n1,m2,n2,i = 0,j,k = 0,t = 0,temp = 0,a[21][21] = {0};

Dg t1[100],t2[100];

printf("请输入矩阵的运算方式(+、-、\*)：");

scanf("%c",&c);

printf("请输入两个矩阵的行数和列数：");

scanf("%d %d %d %d",&m1,&n1,&m2,&n2);

switch(c) {

case '+':

if(m1 != m2 || n1 != n2) {

printf("输入错误，请检查矩阵的行列输入是否正确");

break;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

a:scanf("%d",&t1[0].m);

if(t1[0].m == 0 && temp == 1) break;

else {

if(t1[0].m == 0 && temp == 0) {

temp++;

goto a;

}

scanf("%d %d",&t1[0].n,&t1[0].num);

a[t1[0].m][t1[0].n] += t1[0].num;

goto a;

}

case '-':

if(m1 != m2 || n1 != n2) {

printf("输入错误，请检查矩阵的行列输入是否正确");

goto e;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

b:scanf("%d",&t1[0].m);

if(t1[0].m == 0 && temp == 1) break;

else {

if(t1[0].m == 0 && temp == 0) {

temp++;

goto b;

}

scanf("%d %d",&t1[0].n,&t1[0].num);

if(temp == 0) a[t1[0].m][t1[0].n] += t1[0].num;

else a[t1[0].m][t1[0].n] -= t1[0].num;

goto b;

}

case '\*':

if(n1 != m2) {

printf("输入错误，请检查矩阵的行列输入是否正确");

goto e;

}

printf("请依次输入两个稀疏矩阵非零元素所在的行和列以及元素值，以0结束:\n");

c:if(temp == 0) scanf("%d",&t1[i].m);

else scanf("%d",&t2[i].m);

if(temp == 1 && t2[i].m == 0) {

for(i = 0;i < t;i++) {

for(j = 0;j < k;j++) {

if(t1[i].n == t2[j].m) {

a[t1[i].m][t2[j].n] += t1[i].num \* t2[j].num;

}

}

}

break;

}

else {

if(t1[i].m == 0 && temp == 0) {

temp++;

i = 0;

goto c;

}

if(temp == 0) {

scanf("%d %d",&t1[i].n,&t1[i].num);

i++;

t++;

}

else {

scanf("%d %d",&t2[i].n,&t2[i].num);

i++;

k++;

}

goto c;

}

}

for(i = 1;i <= m1;i++) {

for(j = 1;j <= n2;j++) {

printf("%d ",a[i][j]);

}

printf("\n");

}

e:return 0;

}