数据结构实验报告5

学号：XXXXXXXX 姓名：张某某 专业：XXXXXXXX

知识范畴：数组与广义表 完成日期：2023年 月 日

实验题目：基于十字链表的稀疏矩阵加法

|  |  |  |  |
| --- | --- | --- | --- |
| 课程目标1（60%） | 课程目标2（40%） | 得分(100分)  期末成绩占比5% | 批阅人签字 |
| 程序代码及测试 | 写作、排版、代码注释等 |  |
|  |  |  |

实验内容及要求：

从字符文件读入3元组(*i*, *j*, *e*)来建立两个矩阵A和B的十字链表存储结构（矩阵的行、列数*m*, *n*以及非零元素的数目需首先读入）。计算C=A+B，其中矩阵C也采用十字链表存储结构。计算结束后，以三元组形式输出矩阵C至另一个字符文件。矩阵C的十字链表存储空间可以独立于A和B，也可以就用A矩阵的十字链表来存储（则A矩阵的非零元素结点数目可能增加、减少或不变）。

要求实验报告必须写出：输入文件数据格式；输出文件数据格式；加法的实现算法；各子函数的详细说明等。每个同学至少准备一个测试用例（A, B以及期望的C）。每个同学除自己的测试用例外，必须随机选择另外两名同学，用他们的测试用例测试自己的程序。在实验报告中需注明3个测试用例分别来自哪位同学（给出姓名和学号，自己准备的测试用例也需写出姓名和学号）。

实验目的：掌握稀疏矩阵的十字链表存储结构及其基本操作。

数据结构设计简要描述：

typedef struct triTuple

{

    int i, j;

    int e; //数据域

    struct triTuple \*r, \*d; //指针域

} clnode, \*clptr;

class CrossList

{

private:

    int m, n, t; //最大行数, 最大列数, 非零元总数

clptr \*rhead, \*chead;

public:

(见主要函数说明)

};

算法设计简要描述：

通过插入节点的方式, 将矩阵B的所有节点插入A, 若某个节点的位置已有元素, 则将两者的值相加, 若最后结果为0, 则删除该节点.

输入/输出设计简要描述：

从文件中读取数据, 第一行输入矩阵A的最大行数和最大列数和非零元个数t, 接下来t行每行有三个数据, 即元素的所在行, 所在列以及元素值. 第(t + 2) 行输入矩阵B的最大行数最大列数以及非零元个数t, 接下来t行每行输入一个三元组.

输出包含结果矩阵的最大行, 最大列以及非零元个数, 以及每个非零元.

编程语言说明：

使用Visual C++编程。 主要代码采用C语言实现 ；动态存储分配采用C++的new和delete操作符实现；输入与输出采用C++的cin和cout流；程序注释采用C/C++规范。

主要函数说明：

CrossList(); //无参构造函数

    CrossList(CrossList &a, CrossList &b); //通过两个十字链表构造他们的和十字链表

    ~CrossList(); //析构函数

    void assign(); //从字符文件中读取数据为十字链表赋值

    void insert(clptr e); //往十字链表中插入节点

    void remove(int ii, int jj); //删除指定位置节点

    void traverse() const; //遍历链表, 输出到文件中

    void add(CrossList &cl); //将一个十字链表中的所有节点加入到另一个十字链表中

程序测试简要报告：

测试1: 来自2022114467林泽宇

输入:

4 2 5

1 0 4

1 0 -4

1 1 5

1 1 -5

0 0 -9

9 9 4

0 1 4

1 2 5

2 1 7

8 8 9

输出:

9 9 5

(0, 0, -9)

(0, 1, 4)

(1, 2, 5)

(2, 1, 7)

(8, 8, 9)

经过检查, 结果正确.

结论: 程序可以得出正确结果, 并且当非零项出现是时会将其删除, 可以进行不同行列数的矩阵间相加, 从而提高程序鲁棒性.

测试2: 来自2022112893罗溢铭

输入:

4 4 5

1 0 4

0 1 4

1 1 5

3 2 4

3 3 9

3 4 4

0 1 4

1 2 5

2 1 7

2 2 9

输出:

4 4 8

(0, 1, 8)

(1, 0, 4)

(1, 1, 5)

(1, 2, 5)

(2, 1, 7)

(2, 2, 9)

(3, 2, 4)

(3, 3, 9)

经过检查, 结果正确

测试3: 来自 2022114533 汪维诚

输入:

3 4 3

1 1 1

1 3 1

2 2 2

3 4 2

1 2 1

2 2 3

输出:

3 4 4

(1, 1, 1)

(1, 2, 1)

(1, 3, 1)

(2, 2, 5)

经过检查, 结果正确

源程序代码：

#include <iostream>

#include <fstream>

#include <algorithm>

using namespace std;

ifstream iFile;

ofstream oFile;

typedef struct triTuple

{

    int i, j;

    int e; //数据域

    struct triTuple \*r, \*d; //指针域

} clnode, \*clptr;

class CrossList

{

private:

    int m, n, t; //最大行数, 最大列数, 非零元总数

    clptr \*rhead, \*chead;

public:

    CrossList(); //无参构造函数

    CrossList(CrossList &a, CrossList &b); //通过两个十字链表构造他们的和十字链表

    ~CrossList(); //析构函数

    void assign(); //从字符文件中读取数据为十字链表赋值

    void insert(clptr e); //往十字链表中插入节点

    void remove(int ii, int jj); //删除指定位置节点

    void traverse() const; //遍历链表, 输出到文件中

    void add(CrossList &cl); //将一个十字链表中的所有节点加入到另一个十字链表中

};

int main()

{

    iFile.open("input.txt");

    oFile.open("output.txt");

    CrossList cl1;

    CrossList cl2;

    CrossList cl3(cl1, cl2); //通过两个链表构建他们的和

    cl3.traverse(); //输出结果到文件

    iFile.close();

    oFile.close();

}

CrossList::CrossList()

{

    int mm, nn, tt;

    iFile >> mm >> nn >> tt; //读取数据

    rhead = new clptr[mm + 1]; //分配空间

    chead = new clptr[nn + 1];

    for (int i = 0; i < mm; ++i) //置空指针

        rhead[i] = nullptr;

    for (int i = 0; i < nn; ++i)

        chead[i] = nullptr;

    m = mm, n = nn, t = tt;

    assign(); //进行赋值

}

CrossList::CrossList(CrossList &a, CrossList &b)

{

    m = max(a.m, b.m), n = max(a.n, b.n), t = 0; //新十字链表的行数与列数取两个十字链表的最大值

    rhead = new clptr[m + 1];

    chead = new clptr[n + 1];

    for (int i = 0; i < m; ++i)

        rhead[i] = nullptr;

    for (int i = 0; i < n; ++i)

        chead[i] = nullptr;

    this->add(a);

    this->add(b); //分别将两个链表中的所有节点插入新链表中

}

CrossList::~CrossList()

{

    clptr p, q;

    for (int i = 0; i < m; ++i)

    {

        if (rhead[i])

            p = rhead[i]->r;

        while (p)

        {

            q = p->r; //存储下一个位置的信息

            delete p;

            p = q;

        }

    }

    p = q = nullptr;

    delete[] rhead;

    delete[] chead;

    rhead = chead = nullptr; //记得置空

    t = 0;

}

void CrossList::assign()

{

    int ii, jj, ee, tt = t;

    for (int i = 0; i < tt; ++i)

    {

        iFile >> ii >> jj >> ee;

        clptr e = new clnode({ii, jj, ee, nullptr, nullptr});

        insert(e); //循环插入节点

    }

}

void CrossList::insert(clptr e)

{

    clptr p;

    if (!rhead[e->i] || rhead[e->i]->j > e->j) //如果在开头则直接插入

        e->r = rhead[e->i], rhead[e->i] = e;

    else

    {

        p = rhead[e->i];

        while (p->r && p->r->j < e->j)

            p = p->r;

        if (p->r && p->r->j == e->j) //如果存在相同坐标的项, 则直接相加

        {

            p = p->r;

            p->e += e->e;

            if (!p->e) //如果结果为0则删除该节点并返回

                remove(e->i, e->j);

            --t;

            return;

        }

        else if (p->j == e->j)

        {

            p->e += e->e;

            if (!p->e)

                remove(e->i, e->j);

            --t;

            return;

        }

        else

            e->r = p->r, p->r = e;

    }

    if (!chead[e->j] || chead[e->j]->i > e->i)

        e->d = chead[e->j], chead[e->j] = e;

    else

    {

        p = chead[e->j];

        while (p->d && p->d->i < e->i)

            p = p->d;

        e->d = p->d, p->d = e;

    }

    return;

}

void CrossList::remove(int ii, int jj)

{

    --t; //非零元数目减一

    clptr p = rhead[ii], q;

    if (p->j == jj)

        rhead[ii] = nullptr;

    else

    {

        while (p->r->j < jj)

            p = p->r;

        q = p->r, p->r = q->r;

        delete q;

    }

    p = chead[jj];

    if (p->i == ii)

        chead[jj] == nullptr;

    else

    {

        while (p->d->i < ii)

            p = p->d;

        q = p->d, p->d = q->d;

        delete q;

    }

}

void CrossList::traverse() const //不修改对象

{

    oFile << m << " " << n << " " << t << '\n';

    clptr p;

    for (int i = 0; i < m; ++i)

    {

        if (rhead[i])

        {

            p = rhead[i];

            while (p)

            {

                oFile << '(' << p->i << ", " << p->j << ", " << p->e << ")\n";

                p = p->r;

            }

        }

    }

}

void CrossList::add(CrossList &cl)

{

    t += cl.t; //项数增加

    clptr p, q;

    for (int i = 0; i < cl.m; ++i)

    {

        if (cl.rhead[i])

        {

            p = cl.rhead[i];

            while (p)

            {

                q = p->r;

                p = new clnode({p->i, p->j, p->e, nullptr, nullptr});

                this->insert(p); //逐个插入

                p = q;

            }

        }

    }

}