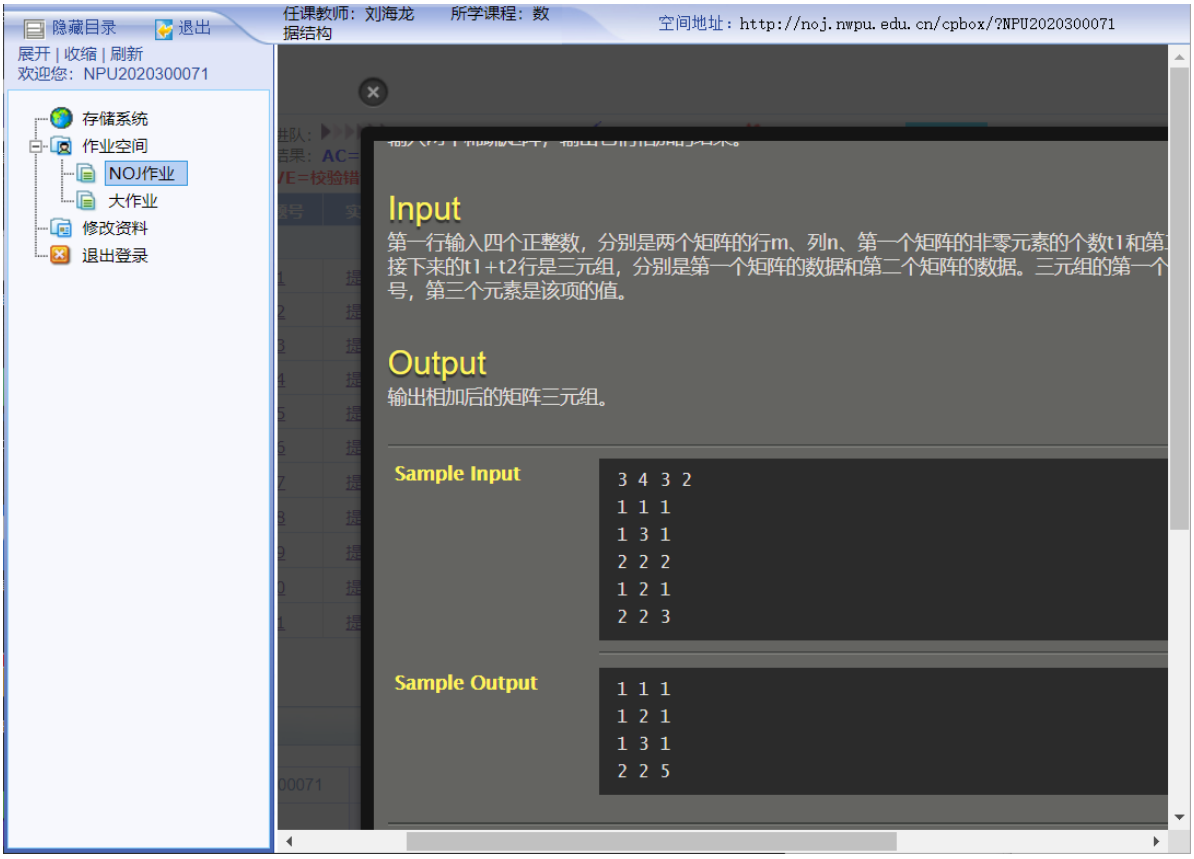
noj实验4&&5报告

0.0 题目//需求分析



需求分析：即输出两个用三元组表示的矩阵的和。

第一行是两个矩阵的行列数m1,n1,m2,n2;

后面的几行就是三元组表示的矩阵

输出的几行也是三元组形式。

ps：我在实验四就用了十字链表，所以就五也直接过了

1.0 实验思路

第一步，初始化结构体，单个位置与矩阵整体（十字链表）

第二步，在主函数中先输入行列值（第一行）并进行判断，把max给大值。并对十字链表初始化为NULL。

第三步 执行加法。

第四步， 输出（这次我是分了一个函数进行输出）

第五步，具体的加法算法，分很多if-else语句，把每种情况描述清楚。

2.0 代码

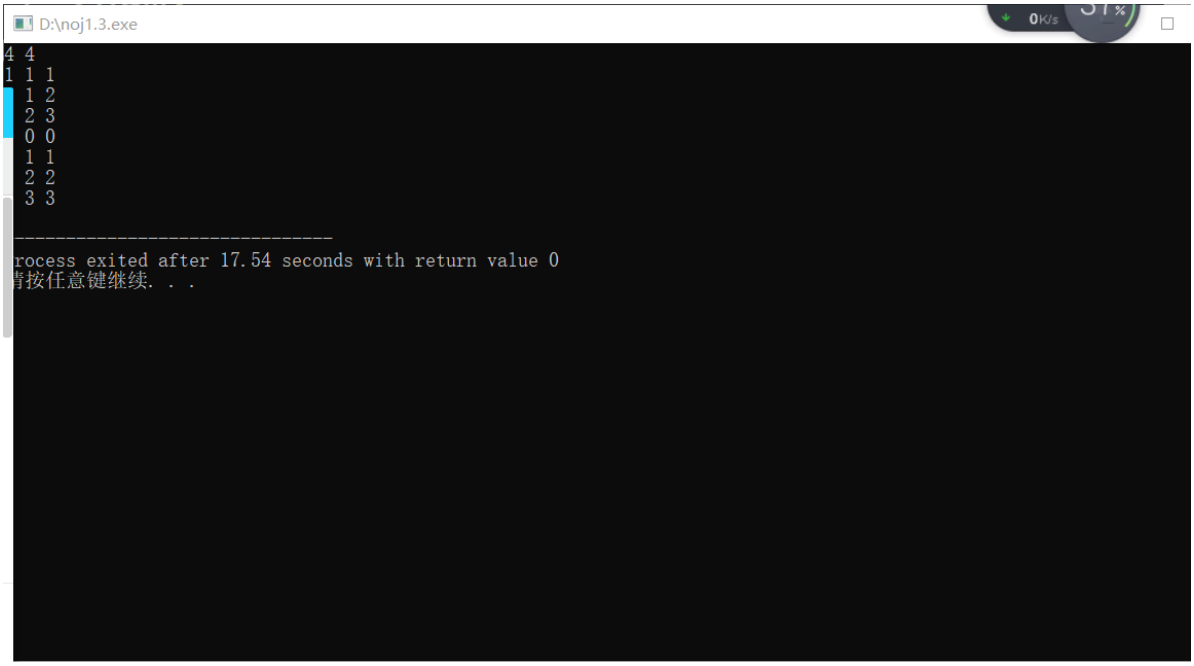
```
#include<stdio.h>
#include<stdlib.h>
```

```c
typedef struct OLNode{
    int row,col;
    int num;
    struct OLNode *right,*down;
}OLNode,*OLink;

typedef struct CrossList{
    OLink rHead[100],cHead[100];
    int ru,cu,tu;
}CrossList;



void init(CrossList *M,int cnt){   //是初始化也算是输入
    int x,y,z;
    OLink r;
    while(cnt--){
        OLink q=(OLink)malloc(sizeof(OLNode));
        scanf("%d%d%d",&x,&y,&z);
        q->row=x;q->col=y;q->num=z;
        if(M->rHead[x]==NULL||M->rHead[x]->col > y){
            q->right=M->rHead[x]; M->rHead[x]=q;
            M->ru++;
        }
        else{
            for(r=M->rHead[x];r->right && r->right->col < y;r=r->right);
            q->right=r->right; r->right=q;
            M->ru++;
        }//完成行插入
        if(M->cHead[y]==NULL||M->cHead[y]->row > x){
            q->down=M->cHead[y]; M->cHead[y]=q;
            M->cu++;
        }
        else{
            for(r=M->cHead[y];r->down && r->down->row < x;r=r->down);
            q->down=r->down; r->down=q;
            M->cu++;
        }//完成列插入
    }
}

void add(CrossList *M,int cnt){  //十字链表加法函数
    OLink l,t,p;
    int flag,flag1;
    int i,x,y,z;
    while(cnt--){
        scanf("%d%d%d",&x,&y,&z);
        OLink r=(OLink)malloc(sizeof(OLNode));
        r->row=x; r->col=y; r->num=z;
        flag=1;  flag1=1;
        if(M->rHead[x] != NULL){
            for(l=M->rHead[x];l;l=l->right){
                if(l->col == r->col){
                    l->num+=r->num;
                    if(l->num == 0){//删除0
                        if(flag1){
                            flag1=0;
                            M->rHead[x] = M->rHead[x]->right;
```

```
                        }
                        else{
                            p = M->rHead[x];
                            while(p->right != l){
                                p = p->right;
                            }
                            p->right = l->right;
                        }
                        for(t = M->cHead[y];t;t=t->down){
                            if(t == M->cHead[y]){
                                M->cHead[y] = t->right;
                                break;
                            }
                            else{
                                p = M->cHead[y];
                                while(p->down != l){
                                    p = p->down;
                                }
                                p->down = l->down;
                                break;
                            }
                        }
                        free(l);
                    }
                    flag=0; break;
                }
            }
        }
        if(flag){
            if(M->rHead[x]==NULL){
                r->right=M->rHead[x];
                M->rHead[x]=r;
                M->ru++;
            }
            else if(M->rHead[x]->col > y){
                r->right=M->rHead[x];
                M->rHead[x]=r;
            }
            else{
                for(l=M->rHead[x];l->right && l->right->col < y;l=l->right);
                r->right=l->right; l->right=r;
            }
            if(M->cHead[y]==NULL){
                r->down=M->cHead[y];
                M->cHead[y]=r;
                M->cu++;
            }
            else if(M->cHead[y]->row > x){
                r->down = M->cHead[y];
                M->cHead[y]=r;
            }
            else{
                for(l=M->cHead[y];l->down && l->down->row < x;l=l->down);
                r->down=l->down; l->down=r;
            }
        }
    }
}
```

```
void output(CrossList *M){  //输出函数
    int i;
    OLink r,t;
    for(i=1;i<=M->ru;i++){
        if(M->rHead[i]){
            for(t=M->rHead[i];t;t=t->right){
                printf("%d %d %d\n",t->row,t->col,t->num);
            }
        }
    }
}

int main(){
    CrossList *M;
    M=(CrossList *)malloc(sizeof(CrossList));
    int m,n,t1,t2,max;
    scanf("%d%d%d%d",&m,&n,&t1,&t2);
    int i;
    M->cu=0;M->ru=0;//初始化
    if(m>n) max=m;
    else max=n;
    for(i=1;i<=max;i++){
        M->cHead[i]=NULL;
        M->rHead[i]=NULL;
    }
    init(M,t1);  //初始化
    add(M,t2);  //执行加法
    output(M);   //输出
    return 0;
}
```

3.0 测试结果



4.0 实验心得

先想思路，再码代码，就有一种按照蓝图做事的感觉，就不会太难。

这也是老师一直教我们的。

注：参考了网上资料思路。