**数据结构指导书上算法实现**

1. 循环链表

源代码：

#include<stdio.h>

#include<malloc.h>

typedef struct node

{

int data;

struct node \*next;

}Linklist;

Linklist \*RA,\*RB;

Linklist \*createR()//尾插法创建带头结点的带尾指针的单循环链表

{

Linklist \*L,\*S,\*R;

int x;

L=(Linklist \*)malloc(sizeof(Linklist));

L->next=NULL;

R=L;

printf("请输入一整数序列（以0结束）：\n");

scanf("%d",&x);getchar();

while(x!=0)

{

S=(Linklist \*)malloc(sizeof(Linklist));

S->data=x;

S->next=NULL;

R->next=S;

R=S;

scanf("%d",&x);getchar();

}

R->next=L;

return R;

}

Linklist \*merge(Linklist \*RA,Linklist \*RB)//合并LA和LB

{

Linklist \*u;

u=RA->next;

RA->next=RB->next->next;

free(RB->next);

RB->next=u;

RA=RB;

return RA;

}

void Lprint(Linklist \*R)

{

Linklist \*L,\*p;

L=R->next;

p=L->next;

printf("输出链表：\n");

if(R==L)

printf("This is an empty list!\n");

else

{

while(p!=R)

{

printf("%d ",p->data);

p=p->next;

}

printf("%d ",p->data);

}

printf("\n");

}

int main()

{

printf("创建RA：\n");

RA=createR();

Lprint(RA);

printf("创建RB：\n");

RB=createR();

Lprint(RB);

printf("合并后结果为：\n");

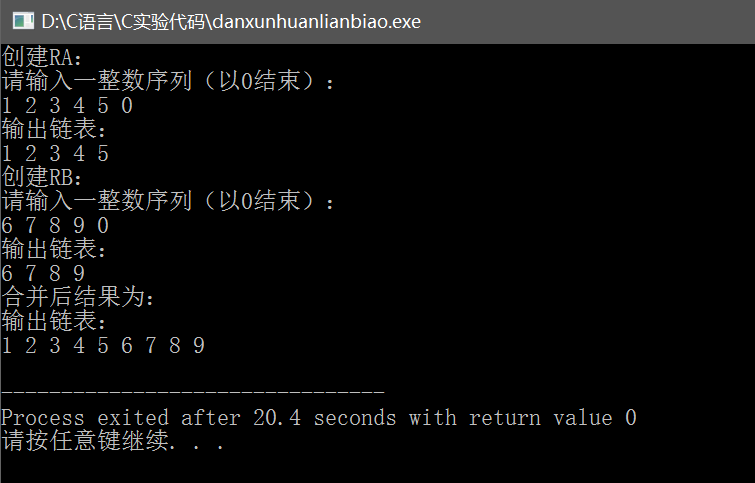
merge(RA,RB);

Lprint(RB);

return 0;

}

测试结果：



1. 稀疏矩阵的顺序存储

源代码：

#include<stdio.h>

#include<malloc.h>

#define smax 10

typedef struct

{

int i,j,v;

}node;//定义结点类型

typedef struct

{

int m,n,t;

node data[smax];

}spmatrix;//定义矩阵类型

spmatrix \*setmatrix()//建立稀疏矩阵的三元组表算法

{

int i;

spmatrix \*A;

A=(spmatrix\*)malloc(sizeof(spmatrix));

printf("请输入稀疏矩阵的行数、列数及非0元素的个数：\n");

scanf("%d%d%d",&(A->m),&(A->n),&(A->t));

printf("建立三元组：\n");

for(i=0;i<A->t;i++)

scanf("%d%d%d",&(A->data[i].i),&(A->data[i].j),&(A->data[i].v));

return A;

}

void spmatrixout(spmatrix \*A)//输出稀疏矩阵

{

int a,b,l=0;

for(a=0;a<A->m;a++)

{

for(b=0;b<A->n;b++)

{

if(A->data[l].i==a&&A->data[l].j==b)

{

printf("%-5d",A->data[l].v);

l++;

}

else

printf("%-5d",0);

}

printf("\n");

}

}

int main()

{

spmatrix \*A;

A=setmatrix();

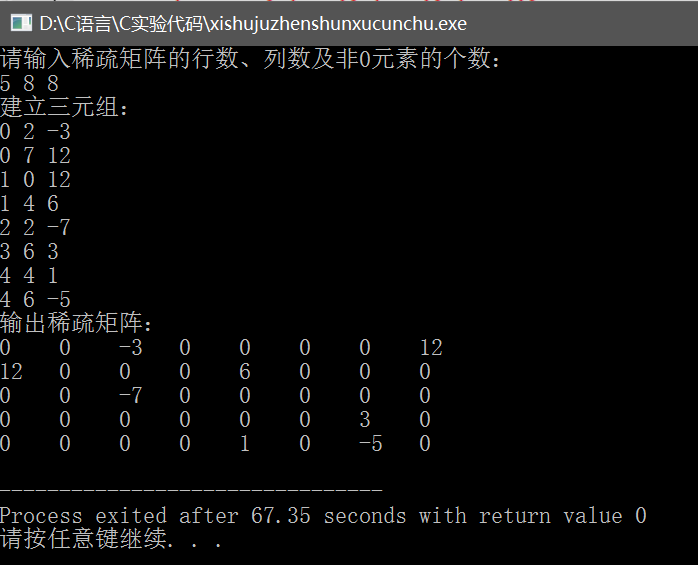
printf("输出稀疏矩阵：\n");

spmatrixout(A);

return 0;

}

测试结果：



1. 矩阵的应用
2. 稀疏矩阵的转置问题

源代码：

#include<stdio.h>

#include<malloc.h>

#define smax 10

typedef struct

{

int i,j,v;

}node;

typedef struct

{

int m,n,t;

node data[smax];

}spmatrix;

spmatrix \*setmatrix()

{

spmatrix \*a;

int i;

a=(spmatrix \*)malloc(sizeof(spmatrix));

printf("请输入稀疏矩阵的行数、列数及非0元素的个数：\n");

scanf("%d%d%d",&(a->m),&(a->n),&(a->t));

printf("建立三元组：\n");

for(i=0;i<a->t;i++)

scanf("%d%d%d",&(a->data[i].i),&(a->data[i].j),&(a->data[i].v));

return a;

}

void spmatrixout(spmatrix \*a)

{

int c,d,n=0;

for(c=0;c<a->m;c++)

{

for(d=0;d<a->n;d++)

{

if(a->data[n].i==c&&a->data[n].j==d)

{

printf("%-5d",a->data[n].v);

n++;

}

else

printf("%-5d",0);

}

printf("\n");

}

}

spmatrix \*trabsmatrix(spmatrix \*a)

{

spmatrix \*b;

int bow,acol,k;

b=(spmatrix \*)malloc(sizeof(spmatrix));

b->m=a->n;

b->n=a->m;

b->t=a->t;

if(b->t>0)

{

bow=0;

for(acol=0;acol<a->n;acol++)

{

for(k=0;k<a->t;k++)

{

if(a->data[k].j==acol)

{

b->data[bow].i=a->data[k].j;

b->data[bow].j=a->data[k].i;

b->data[bow].v=a->data[k].v;

bow++;

}

}

}

}

return b;

}

int main()

{

spmatrix \*a,\*b;

a=setmatrix();

printf("输出稀疏矩阵A：\n");

spmatrixout(a);

b=trabsmatrix(a);

printf("转置后结果为：\n");

spmatrixout(b);

return 0;

}

测试结果：

