## 习题4-19

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 100

typedef struct{

char ch[MAXSIZE];

int len;

}seqstring;

seqstring \*S;

//建立串

seqstring \*Create(){

int i=0;

char ch1;

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

S->len=0;

printf("请输入串中的元素，以#结束\n");

while((ch1=getche())!='#'){

S->ch[i++]=ch1;

S->len++;

}

S->ch[i]='\0';

return S;

}

//打印

void Print(seqstring \*S){

int i=0;

printf("\n");

while(i<S->len){

printf("%c",S->ch[i]);

i++;

}

printf("\n");

}

//删除

void Del(seqstring \*S,int i,int j){

int m;

if(i>S->len || i+j>S->len+1) printf("Error.\n");

else{

for(m=i-1;m+j<S->len;m++){

S->ch[m]=S->ch[m+j];

}

S->ch[m]='\0';

S->len=S->len-j;

}

}

void main(){

S=Create();

Print(S);

Del(S,2,3);

Print(S);

}

## 习题4-20

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

typedef struct linknode{

char data;

struct linknode \*next;

}linkstring;

linkstring \*S;

//建立一个串

linkstring \*Create(){

char ch;

linkstring \*head,\*s,\*r;

head=(linkstring\*)malloc(sizeof(linkstring));

r=head;

printf("请输入串中的元素，以#结束\n ");

while((ch=getche())!='#'){

s=(linkstring\*)malloc(sizeof(linkstring));

s->data=ch;

r->next=s;

r=s;

}

s=(linkstring\*)malloc(sizeof(linkstring));

s->data='\n';

r->next=s;

r=s;

r->next=NULL;

return head;

}

//打印

void Output(linkstring \*head)

{

linkstring \*t;

t=head->next;

printf("\n");

while(t->next!=NULL)

{

printf("%c",t->data);

t=t->next;

}

printf("%c\n",t->data);

}

//替换

void Replace(linkstring \*S,char s,char c){

linkstring \*p;

p=S;

do{

if(p->data==s){

p->data=c;

}

p=p->next;

}

while(p->next!=NULL);

}

void main(){

char m,n;

S=Create();

Output(S);

printf("请依次输入要被替换的字符和替换结果字符\n");

m=getche();

n=getche();

Replace(S,m,n);

Output(S);

}

## 习题4-26

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define SMAX 16

typedef int datatype;

typedef struct{

int i,j;

datatype v;

}node;

typedef struct{

int m,n,t;

node data[SMAX];

}spmatrix;

//建立矩阵

spmatrix\* Create(){

spmatrix\* mat;

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

int i,r,s,da,t=0;

char ch;

printf("请输入稀疏矩阵行数，列数，非零元个数\n");

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

printf("请输入稀疏矩阵三元组表\n");

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

scanf("%d %d %d",&r,&s,&da);

mat->data[t].i=r;

mat->data[t].j=s;

mat->data[t].v=da;

t++;

}

return mat;

}

//打印

void Output(spmatrix\*a){

int k,p;

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

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

if(a->data[k].i==p){

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

printf("\n");

}

}

}

}

//乘法

spmatrix\* Mult(spmatrix \*a,spmatrix \*b){

int p,q,x,y,flag=1,bno=0;

spmatrix \*c;

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

c->m=a->m;

c->n=b->n;

if(a->t==0||b->t==0) return c;

for(p=0;p<a->m;p++){ //按行遍历矩阵a

for(q=0;q<b->n;q++){ //按列遍历矩阵b

for(x=0;x<a->t;x++){ //遍历a中每个元素

if(a->data[x].i==p){ //若元素属于当前处理行（p）

for(y=0;y<b->t;y++){ //遍历b中每个元素

if(b->data[y].j==q){ //若元素属于当前处理列（q）

if(flag){

c->data[bno].v=0;

flag=0;

}

if(a->data[x].j==b->data[y].i){

c->data[bno].v=c->data[bno].v+a->data[x].v\*b->data[y].v;

}

}

}

}

}

if(!flag && c->data[bno].v!=0){

//若a的第p行与b的第q列均存在非零元素且按矩阵法则相乘得到结果不为0

c->data[bno].i=p;

c->data[bno].j=q;

bno++; //将结果放入c中，计算下一个c元素

flag=1;

}

}

}

c->t=bno+1;

return c;

}

void main(){

spmatrix \*a,\*b,\*c;

a=Create();

printf("矩阵a三元组表为\n");

Output(a);

b=Create();

printf("矩阵b三元组表为\n");

Output(b);

c=Mult(a,b);

printf("a\*b结果三元组表为\n");

Output(c);

}