**第二章作业** （15180120002 包嘉斌）

**题2-17**

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 1024

typedef char datetype;

typedef struct{

datetype date[MAXSIZE];

int last;

}sequenlist;

sequenlist \*L;

//建立顺序表

sequenlist \*Create(){

int i=0;

char ch;

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

L -> last = -1;

printf("请输入顺序表L中的元素，以‘#’结束。\n");

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

L -> date[i++]=ch;

L ->last++;

}

return L;

}

//打印

void Output(sequenlist \*L){

int i;

printf("\n顺序表L中的元素为：");

for(i=0;i<=L->last;i++)

printf("%c ",L->date[i]);

printf("\n");

}

/\*右移\*/

int Rmv(sequenlist \*L){

int i,j,k,len;

len = L->last;

printf("请输入右移位数:");

scanf("%d",&k);

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

{

L->date[len+1] = L->date[len];

for(i=len;i>0;i--)

L->date[i] = L->date[i-1];

L->date[0] = L->date[len+1];

}

return 0;

}

void main(){

int k;

L=Create();

Rmv(L);

Output(L);

}

**题2-21（顺序表）**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 1024

typedef char datetype;

typedef struct

{

datetype date[MAXSIZE];

int last;

}sequenlist;

sequenlist \*L;

//建立

sequenlist \*Create(){

int i=0;

char ch;

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

L -> last = -1;

printf("请输入顺序表L中的元素，以‘#’结束。\n");

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

L -> date[i++]=ch;

L ->last++;

}

return L;

}

//打印

void Output(sequenlist \*L){

int i;

printf("\n逆序后顺序表L中的元素为：");

for(i=0;i<=L->last;i++)

printf("%c ",L->date[i]);

printf("\n");

}

//顺序表逆置

void nz(sequenlist \*L)

{

int i,j,len;

char c;

len = L->last+1;

j=L->last;

for(i=0;i<len/2;i++,j--)

{

c=L->date[i];

L->date[i]=L->date[j];

L->date[j]=c;

}

}

void main(){

L=Create();

nz(L);

Output(L);

}

**题2-21（单链表）**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 1024

typedef char datetype;

typedef struct node

{

datetype date;

struct node \*next;

}linklist;

linklist \*L;

//建立单链表

linklist \*Creat()

{

char ch;

linklist \*head,\*s,\*r;

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

r=head;

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

{

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

s->date=ch;

r->next=s;

r=s;

}

r->next=NULL;

return head;

}

//打印

void Output(linklist \*head)

{

linklist \*t;

t=head->next;

printf("\n");

while(t->next!=NULL)

{

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

t=t->next;

}

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

}

//单链表逆置

void nz(linklist \*head)

{

linklist \*p,\*s,\*t;

p=head;

s=p->next;

while(s->next!=NULL)

{

t=s->next;

s->next=p;

p=s;

s=t;

}

s->next=p;

head->next->next=NULL;

head->next=s;

}

void main(){

L=Creat();

Output(L);

nz(L);

Output(L);

}

**题2-27**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 1024

typedef char datetype;

typedef struct node

{

datetype data;

struct node \*next;

}linklist;

linklist \*L;

//建立单链表

linklist \*Creat()

{

char ch;

linklist \*head,\*s,\*r;

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

r=head;

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

{

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

s->data=ch;

r->next=s;

r=s;

}

r->next=NULL;

return head;

}

//打印

void Output(linklist \*head)

{

linklist \*t;

t=head->next;

printf("\n");

while(t->next!=NULL)

{

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

t=t->next;

}

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

}

//递减合并

linklist \*Union(linklist \*la,linklist\*lb)

{

linklist \*p,\*q,\*r,\*u,\*s,\*t;

p=la->next;

q=lb->next;

r=la;

while((p!=NULL)&&(q!=NULL))

{

if(p->data>q->data)

{

u=q->next;

r->next=q;

r=q;

q->next=p;

q=u;

}

else

{

r=p;

p=p->next;

}

}

if(q!=NULL) r->next=q;

p=la;

s=p->next;

while(s->next!=NULL)

{

t=s->next;

s->next=p;

p=s;

s=t;

}

s->next=p;

la->next->next=NULL;

la->next=s;

return la;

}

void main(){

linklist \*L1,\*L2;

L1=Creat();

L2=Creat();

//Output(L);

L=Union(L1,L2);

Output(L);

}

**题2-31**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define MAXSIZE 1024

typedef char datetype;

typedef struct node

{

datetype data;

struct node \*next;

}linklist;

linklist \*L;

//建立单链表

linklist \*Creat()

{

char ch;

linklist \*head,\*s,\*r;

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

r=head;

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

{

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

s->data=ch;

r->next=s;

r=s;

}

r->next=NULL;

return head;

}

//打印

void Output(linklist \*head)

{

linklist \*t;

t=head->next;

printf("\n");

while(t->next!=head)

{

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

t=t->next;

}

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

}

//分类

void fl(linklist\* L)

{

linklist \*p,\*q,\*la,\*lb,\*lc,\*r,\*s,\*t;

la=(linklist\*)malloc(sizeof(linklist));

lb=(linklist\*)malloc(sizeof(linklist));

lc=(linklist\*)malloc(sizeof(linklist));

p=L->next;

r=la;

s=lb;

t=lc;

while(p->next!=NULL)

{

if('0'<=p->data && p->data<='9')

{

r->next=p;

r=r->next;

p=p->next;

}

else if(('a'<=p->data && p->data<='z') || ('A'<=p->data && p->data<='Z'))

{

s->next=p;

s=s->next;

p=p->next;

}

else

{

t->next=p;

t=t->next;

p=p->next;

}

}

if('0'<=p->data && p->data<='9')

{

r->next=p;

r=r->next;

}

else if(('a'<=p->data && p->data<='z') || ('A'<=p->data && p->data<='Z'))

{

s->next=p;

s=s->next;

}

else

{

t->next=p;

t=t->next;

}

r->next=la;

s->next=lb;

t->next=lc;

Output(la);

Output(lb);

Output(lc);

}

void main(){

L=Creat();

fl(L);

}