#include<stdio.h>  
#include<malloc.h>  
#define OK 1  
#define ERROR 0  
#define LISTSIZE 100  
#define LIST\_SIZE 10  
typedef int ElemType;  
typedef int Status;  
typedef struct  
{  
ElemType \*elem;  
int listsize;  
int length;  
}SqList;  
Status InitList(SqList &L)  
{  
L.elem=(ElemType\*)malloc(LISTSIZE\*sizeof(ElemType));  
if(!L.elem)return ERROR;  
L.length=0;  
L.listsize=LISTSIZE;  
return OK;  
}  
Status CreateList(SqList &L,int element[],int len)  
{  
int k;  
L.elem=(ElemType\*)malloc(LISTSIZE\*sizeof(ElemType));  
if(!L.elem)return ERROR;  
L.length=len;  
L.listsize=LISTSIZE;  
for(k=0;k<L.length;k++)  
{  
L.elem[k]=element[k];  
}  
return OK;  
}  
void PrintList(SqList L)  
{  
printf("输出为:\n");  
int k;  
for(k=0;k<L.length;k++)  
{  
printf("%3d",L.elem[k]);  
}  
printf("\n");  
}  
//在第i个元素之前插入元素e  
Status ListInsert\_Sq(SqList &L,int i,ElemType e)  
{  
int j;  
ElemType \*newbase;  
if(L.length>=L.listsize)  
{  
newbase=(ElemType\*)realloc(L.elem,(L.listsize+LIST\_SIZE)\*sizeof(ElemType));  
if(!newbase)  
return ERROR;  
L.elem=newbase;  
L.listsize+=LIST\_SIZE;  
}  
if(i<1||i>L.length+1)  
return ERROR;  
for(j=L.length;j>=i;--j)  
L.elem[j]=L.elem[j-1];  
L.elem[i-1]=e;  
++L.length;  
return OK;  
}  
//删除第i个元素  
Status ListDelete\_Sq(SqList &L,int i,ElemType &e)  
{  
int j;  
if(i<1||i>L.length)  
return ERROR;  
e=L.elem[i-1];  
for(j=i;j<L.length;j++)  
{  
L.elem[j-1]=L.elem[j];  
}  
L.length--;  
return OK;  
}  
//获取线性表上的第i个元素  
Status GetElem\_Sq(SqList L,int i,ElemType &e)  
{  
if(i<1||i>L.length)  
return ERROR;  
e=L.elem[i-1];  
return OK;  
}  
//顺序表的逆置  
Status Inv(SqList &L)  
{  
int i,j;  
for(i=0,j=L.length-1;i<j;i++,j--)  
{  
ElemType e;  
e=L.elem[i];  
L.elem[i]=L.elem[j];  
L.elem[j]=e;  
}  
return OK;  
  
}  
void main()  
{  
int element[]={15,3,59,27,8,11,32};  
SqList La;  
ElemType e;  
if(!InitList(La))  
return;  
CreateList(La,element,7);  
PrintList(La);  
ListInsert\_Sq(La,1,20);//插入  
ListDelete\_Sq(La,1,e);//删除  
PrintList(La);  
Inv(La);//逆置  
PrintList(La);  
return;  
}