```
/*
输入:
按提示输入
输出:
初始链表
合并后链表
*/
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cstring>
using namespace std;
int len=0;
const int maxn=100;
int numa[maxn];
int numb[maxn];
int unionnum[2*maxn];
typedef struct node
{
    int data;
    struct node *next;
};
int cmp(const void *a,const void *b) // 比较函数
{
    return *(int*)a-*(int*)b;
}
node* clearnode(node* head)
{
    head=(node*)malloc(sizeof(node));
    head->data=-1;
    head->next=NULL;
```

```
}
node* createnode(node* head,int* num)
{
    int i=2;
    head=(node*)malloc(sizeof(node));
    node *r,*s;
    s=(node*)malloc(sizeof(node));
    head->next=s;
    head->data=num[0];
    s->data=num[1];
    r=(node*)malloc(sizeof(node));
    r=s;
    len=2;
    while(num[i])
    {
        node* tmp=(node*)malloc(sizeof(node));
        tmp->data=num[i];
        r->next=tmp;
        r=tmp;
        i++;
        len++;
    }
    r->next=NULL;
    return head;
}
node* Union(node* ha,node* hb) // 合并链表
{

    int i=0;
    node *p=ha;
    while(p)
```

```
    {
        unionnum[i]=p->data;
        i++;
        p=p->next;
    }
    p=hb;
    while(p)
    {
        unionnum[i]=p->data;
        i++;
        p=p->next;
    }
    qsort(unionnum,i,sizeof(unionnum[0]),cmp); // 快速排序
    ha=createnode(ha,unionnum);
    return ha;
}
int outputnode(node* hc)
{
    node *p=hc;
    while(p)
    {
        printf("%d ",p->data);
        p=p->next;
    }
    return 1;
}
int main()
{
    memset(numa,0,sizeof(numa[0])); // 将数组元素初始化为0
    memset(numb,0,sizeof(numb[0]));
    int lena,lenb;
```

```
    printf("输入a和b的长度: \n");
    scanf("%d%d",&lena,&lenb);
    printf("输入a的数据: \n");
    for(int i=0;i<lena;i++)
        scanf("%d",&numa[i]);
    printf("输入b的数据: \n");
    for(int i=0;i<lenb;i++)
        scanf("%d",&numb[i]);
    node *ha=(node*)malloc(sizeof(node));
    node *hb=(node*)malloc(sizeof(node));
    ha=createnode(ha,numa);
    hb=createnode(hb,numb);
    printf("链表a为: \n");
    outputnode(ha);
    printf("\n链表b为: \n");
    outputnode(hb);
    ha=Union(ha,hb);
    printf("\n链表c为: \n");
    outputnode(ha);
    return 0;
}
```