目录

一、前言----------2

二、需求分析----------3

三、概要设计----------4

四、详细设计----------5

五、调试分析----------6

六、用户使用说明----------7

七、测试结果----------8

八、总结----------11

九、主要参考文献和附录----------12

前言

员工管理系统是一个工作单位不可缺少的管理工具,它管理的数据对于公司的决策者和管理者来说都至关重要,所以员工管理系统应该能够为用户提供充足的信息和快捷的查询手段。但一直以来各个公司基本上都是靠传统的人工方式来管理员工信息,这种管理方式存在着许多缺点，如：效率低、保密性差，另外时间一长，将产生大量的文件和数据，这对于信息的查找、更新和维护都带来了不少的困难。

当今社会，信息迅速膨胀，随着各个公司的规模增大，有关信息管理工作所涉及的数据量越来越大，员工信息量也大大增加，利用传统的手工查询、登记、修改等方法的处理速度远远跟不上公司的需求，有的公司不得不靠增加人力、物力来进行信息管理。

随着计算机技术的不断提高，计算机作为知识经济时代的产物，其强大的功能已为人们深刻认识，它已进入人类社会的各个行业和领域并发挥着越来越重要的作用，成为人们工作和生活中不可缺少的一部分。

而作为计算机应用的一部分，使用计算机对员工进行管理，具有手工管理所无法比拟的优点。例如：检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高员工管理的效率，也是公司的科学化、正规化管理和与世界接轨的重要条件。

员工管理系统作为一种管理软件正在各公司中得到越来越广泛的应用,且已达到了良好效果。

需求分析

员工信息管理系统是企业管理中的一个重要内容，随着时代的进步，企业也逐渐变得庞大起来。如何管理好企业内部员工的信息，成为企业管理中的一个大问题。在这种情况下，开发一个人力资源管理系统就显得非常必要

现在，市场上可以选购的应用开发产品很多，流行的也有数十种。在目前市场上这些众多的程序开发工具中，有些强调程序语言的弹性与执行效率；有些则偏重于可视化程序开发工具所带来的便利性与效率的得高，各有各的优点和特色，也满足了不同用户的需求。然而，语言的弹性和工具的便利性是密不可分的，只强调程序语言的弹性，却没有便利的工具作配合，会使一些即使非常简单的界面处理动作，也会严重地浪费程序设计师的宝贵时间；相反，如果只有便利的工具，却没有弹性的语言作支持，许多特殊化的处理动作必需要耗费数倍的工夫来处理，使得原来所标榜的效率提高的优点失去了作用。

本系统结合公司实际的人事、制度，经过实际的需求分析，采用功能强大的 Visual C++ 6.0 作为开发工具而开发出来的管理系统。整个系统从符合操作简便、界面友好、灵活、实用、安全的要求出发，本管理系统具有如下功能：

1、问题描述

对单位的员工进行管理，包括插入、删除、查找、排序等功能。

2、要求

员工对象包括姓名、性别、年龄、职位、工龄等信息。

（1）新增一名员工：将新增员工对象按姓名以字典方式员工管理文件中，基本信息中的编号是按照添加顺序自动增加的。

（2）删除一名员工：从员工管理文件中删除一名员工对象，分为根据编号删除、根据姓名删除。

（3）查询：从员工管理文件中查询符合某些条件（编号、姓名）的员工。

（4）修改：根据编号检索出对象，既可以对整个对象修改，也可对某个属性修改。

（5）排序：按照年龄、工龄对所有的员工排序（降序），也可以回复排序以前的员工现实状态。

3、实现提示

员工对象数不必很多，便于一次读入内存，所有操作不经过内外存交换。

（1）当启动程序是，自动从文件（message.txt）中读出员工信息

（2）由键盘输入员工对象存入链表当中。

（3）对员工对象中的"编号、年龄、工龄"按字典顺序进行排序。

（4）对排序后的员工对象进行增、删、查询、修改、排序等操作。

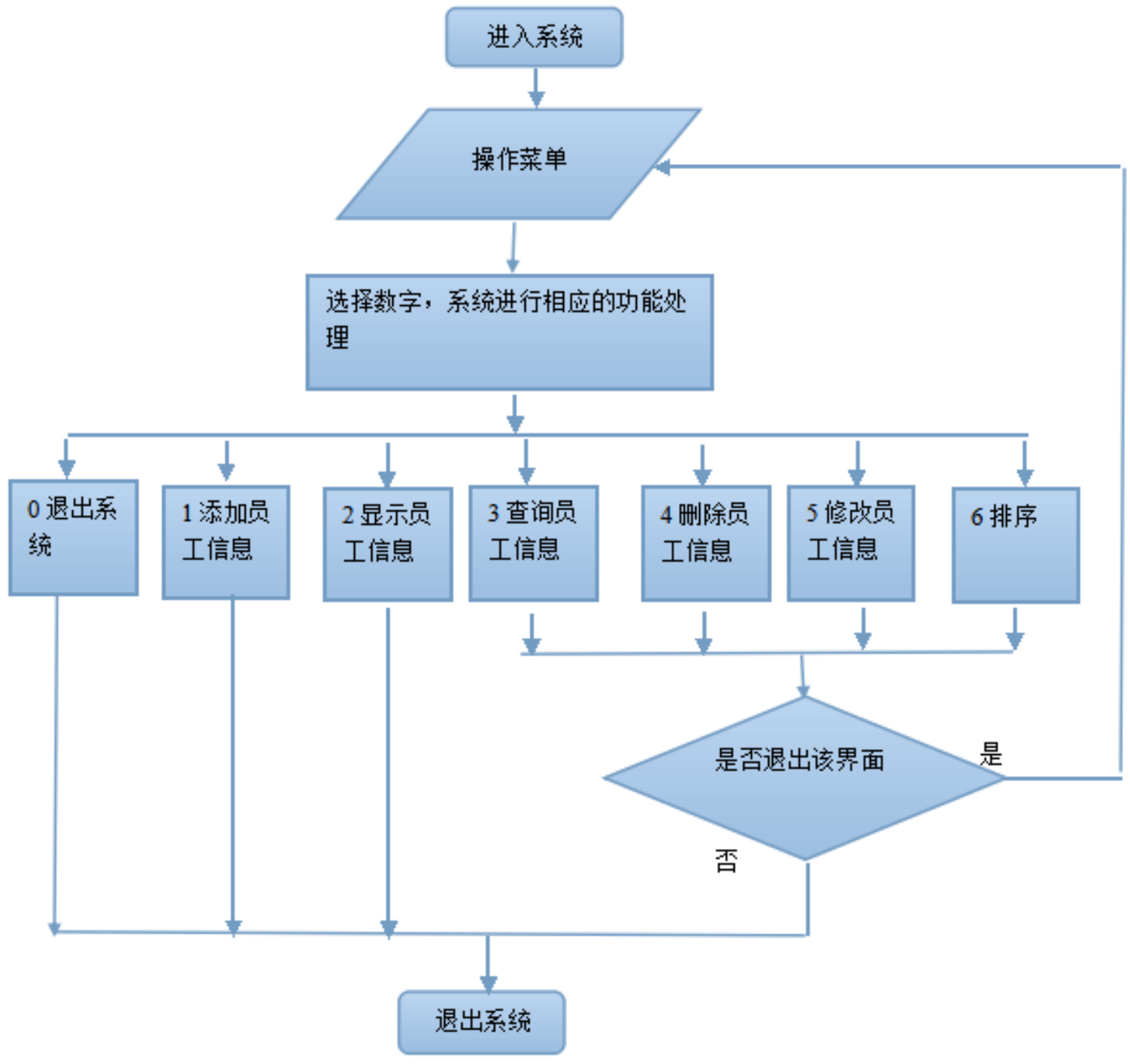
（5）当退出程序时，将此刻单链表中存储的数据写入到文件（message.txt）中去，保存起来。

概要设计

（一）数据类型定义：

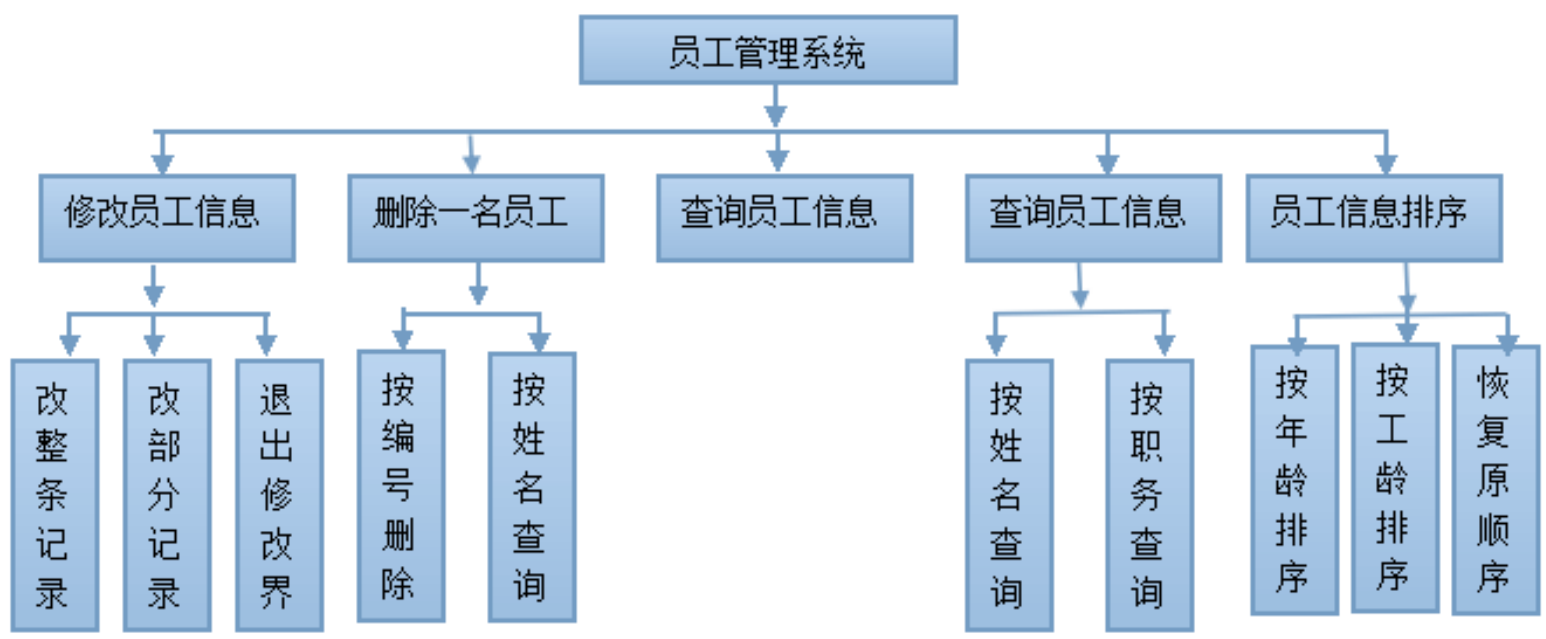
```
typedef struct{
        int num; //编号
        char name[MAX_NUM];//姓名
        int age; //年龄
        char job[MAX_NUM];//职位；
        int workTime;// 工龄
}People;
ypedef struct node{
        People people;
        struct node * next;
        int len;//表示链表长度
    }linklist;
```

（二）流程图：



（图 1）

（三）各程序模块之间的层次图：



（图 2）

详细设计

1、主菜单模块：显示员工管理系统的主菜单，供用户选择所需的功能，通过自己定义的 void main（）函数来实现。

2、添加员工模块：输入员工的编号、姓名、年龄、职位、工龄以，通过自己定义的 void addMessage（）函数来实现。

3、查询员工信息模块：浏览所有员工的相关信息，通过自己定义的 void searchPeople（）函数来实现。

（1）按员工姓名查询：可以按员工工号来查询员工的相关信息，通过自己定义的void searchPeopleByName( )函数来实现。

（2）按员工编号查询：可以按员工职务来查询员工的相关信息，通过自己定义的void searchPeopleByNum( )函数来实现。

（3）退出。

4、删除员工模块：删除需要删除的员工的所有信息，通过自己定义的 void deletePeople( )函数来实现。

（1）按员工姓名删除模块：可以按员工工号来删除员工的相关信息，通过自己定义的 void deletePeopleByName( )函数来实现。

（2）按员工编号删除模块：可以按员工编号删除员工的相关信息，通过自己定义void deletePeopleByNum( )函数来实现。

（3）退出

5、修改模块：可以修改需要修改的员工的相关信息，通过自己定义的 void editMessage（）函数来实现。

（1）修改整条记录，可以修改该员工的全部信息，通过自己定义的void editAll（）函数来实现。

（2）修改部分记录，可以修改该员工的部分信息，通过自己定义的 void editSome（）

函数来实现。

（3）退出。

6、员工信息排序模块：可以按照规定要求对员工信息排序，通过自己定义的 void sort（）函数来实现。

（1）按年龄排序：可以按员工工号对员工信息排序，通过自己定义的 void sortByAge（）函数来。

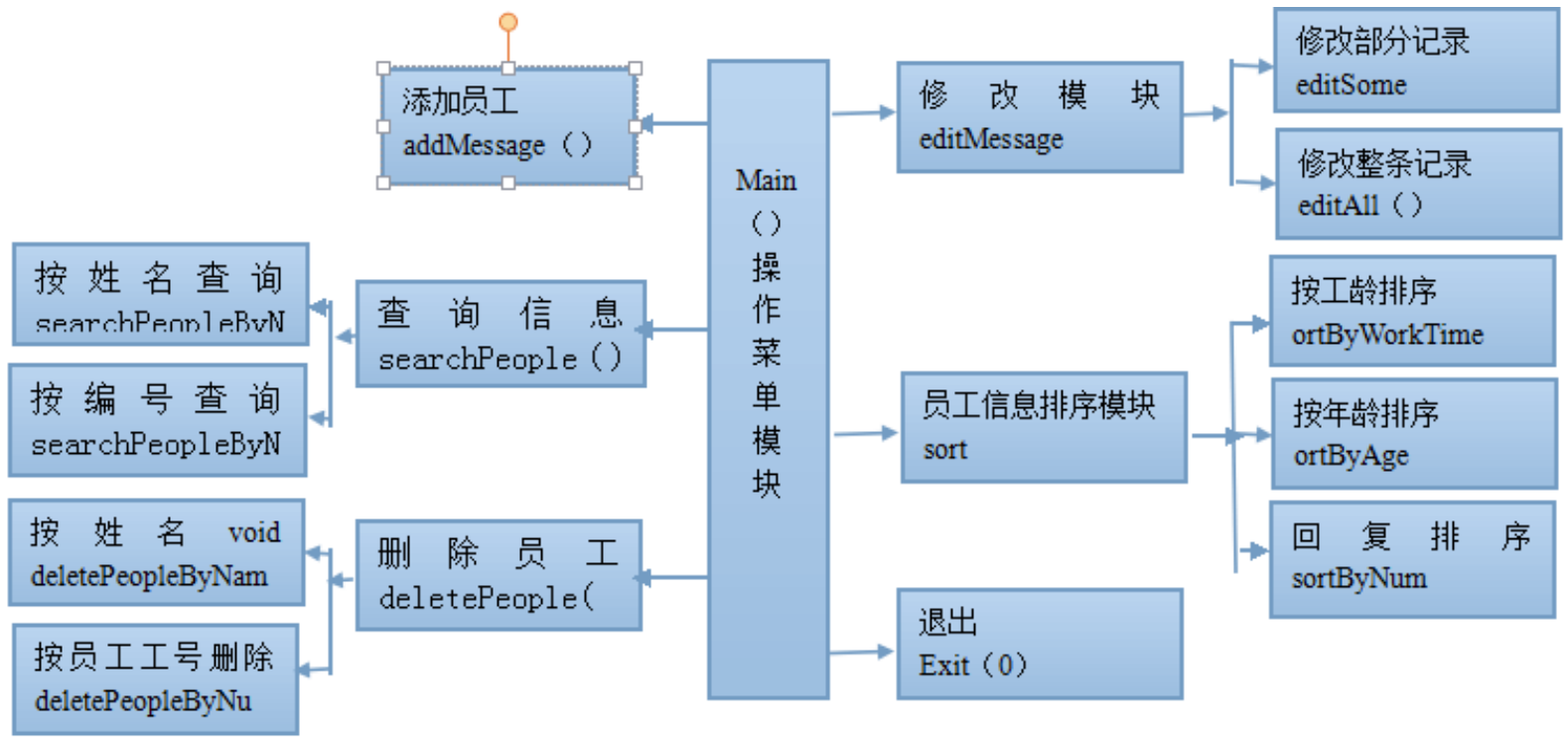
（2）按工龄排序：可以按员工工龄对员工信息排序，通过自己定义的 void sortByWorkTime（）函数来实现。

（3）回复原排序：可以回复排序前的顺序，通过自己定义的 void sortByNum （）函数来实现。

（4）退出。

7、退出系统模块：退出员工信息管理系统，通 exit（0）函数来实现。

函数调用图：



（图 3）

调试分析

测试是使用人工或者自动手段来运行或测试某个系统的过程，其目的在于检验是否满足规定的需求或弄清预期结果与实际结果之间的差别。

在调试查询修改功能过程中，查询的结果显示，没有找到员工信息，最后发现查找的结点不正确，查询应该与输入的值和头结点 next 比较。此外查询结点不知道如何循环，反复修改程序才知道如何继续查找而不出错误。

本次课程设计是围绕数据结构进行。根据问题描述可知，需要解决问题并不复杂，整个问题只需要实现一个员工管理系统功能，那就是在这个系统中实现对员工信息的插入、删除、查询、排序、修改。但是，为了实现该功能，却需要优秀的算法和数据结构以保证实现的时间和空间效率。把员工信息存储在一个单链表中，利用指针实现对员工信息的各项基本操作。

虽然设计的程序完成了题目描述所需要实现的功能，但是仍然存在不如人意的地方。可以排序上面多设计几个算法，实现多角度排序。在这个系统中没有员工序号的信息，所以允

许员工姓名相同，在一定程度上可能存在员工信息重复。

经过这次数据结构课程设计，我们不仅及时巩固的了数据结构、算法、以及软件工程的知识，并明白数据结构和算法对于程序时间和空间性能的影响，及软件工程提供的开发流程和工具对于实现特定功能程序的重要意义。

当我们面对一个实际问题，应该迅速根据问题性质和特点抽象成特定的数据结构，当然每个问题都有可能能够抽象成多种数据结构，每种数据结构适应于不同的算法。因此应该综合考虑这样的数据结构、算法以及它们的空间和时间效率，然后从中选择一个作为实现程序的基础。

用户使用说明

进入员工管理系统，首先看到的就是主菜单界面，然后提示:“请选择主菜单(0---6):”

如果选择 1，进入添加员工模块，按照提示语依次录入员工对象信息。

选择 2，进入展示员工所有的信息模块，DOS 界面显示刚才录入的员工信息。

选择 3，进入员工查询模块，在此模块下:

系统提示：按姓名查询、按编号查询，系统根据用户选择进行相应的处理，退出查询模块时，系统进入主菜单模块。

选择 4，进入删除员工模块，在此模块下:

系统提示：按编号删除、按姓名删除，系统根据用户选择，进行相应的处理，退出删除模块时，系统进入主菜单模块。

选择 5，进入修改员工信息模块，在该模块下:

系统提示：修改整条信息，修改部分信息，系统根据用户选择，进行相应的功能处理。当用户选择退出修改模块时，系统进入主菜单模块。

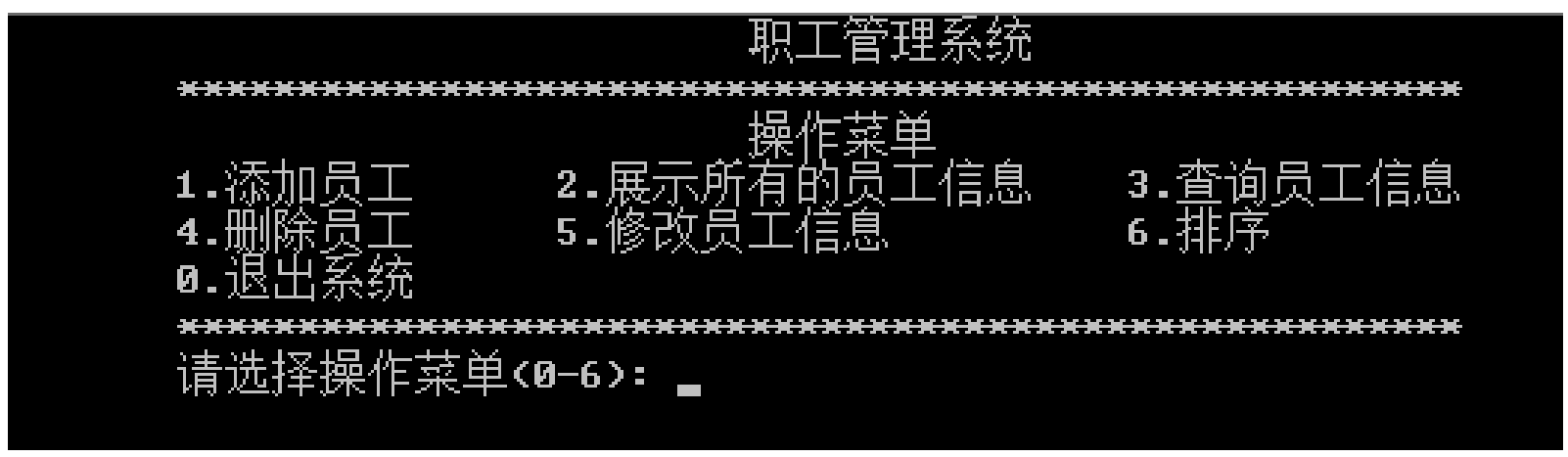
选择 6，进入排序模块，在该模块下:

系统提示：按年龄排序、按工龄排序、回复原排序，系统根据用户的选择，进行相应的处理。

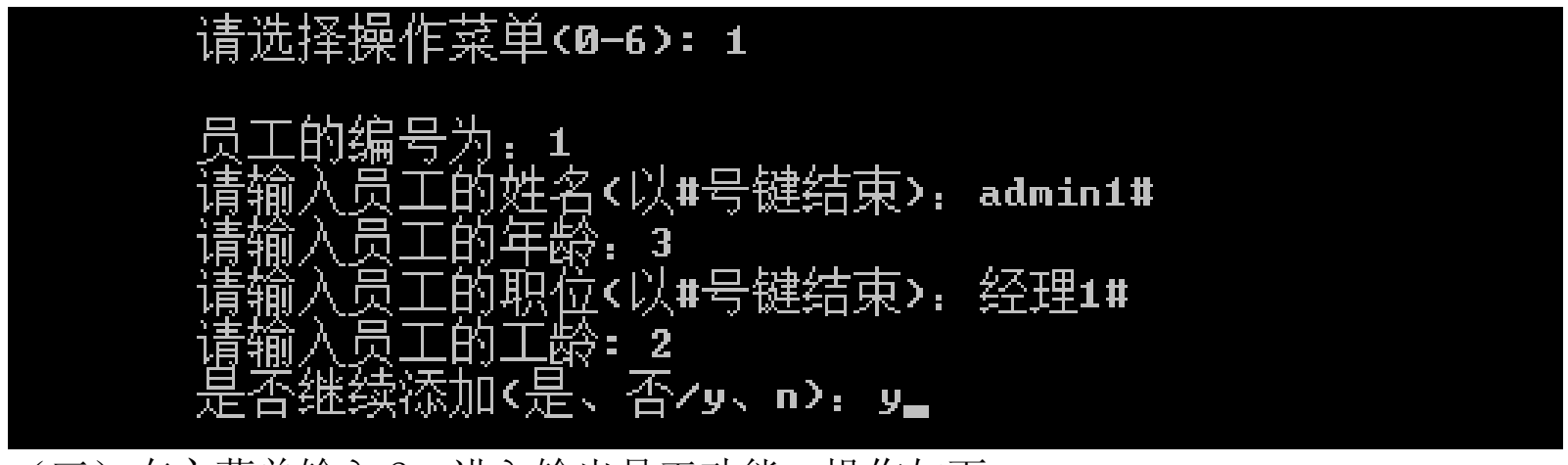
选择 0，退出系统。

测试结果

（一）当操作人员运行程序时，弹出的DOS 界面如下：



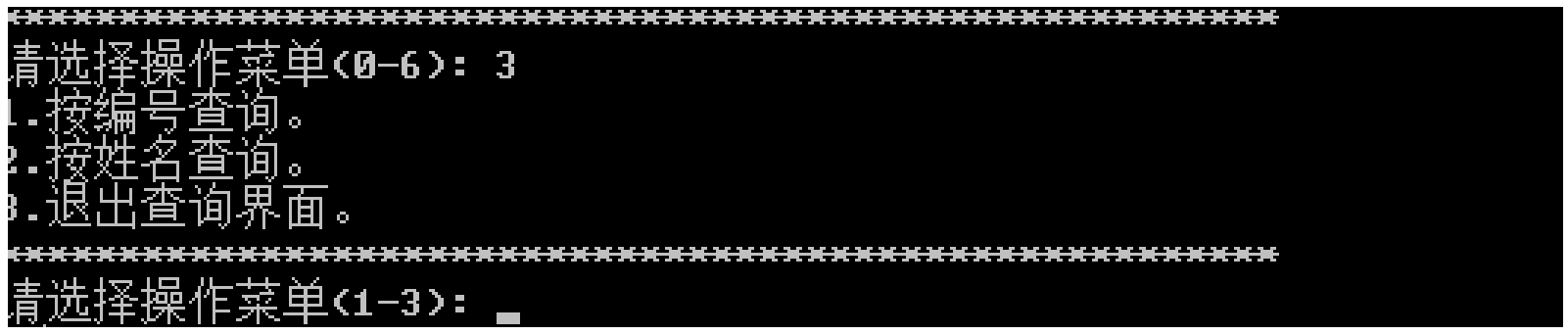
（二）根据提示语，输入1，添加员工信息，操作如下界面：



（三）在主菜单输入2，进入输出员工功能，操作如下：

添加员工、展示所有员工信息功能实现。

（四）在主菜单输入3，查询员工信息



按编号查询,测试如下：

按姓名查询，测试如下：

```
请选择操作菜单(1-3): 2
请输入你要查询的姓名(以#号键结束): admin1#
编号        姓名                    年龄        职位                    工龄
1           admin1                  2           经理1                   1
```

以上查询功能测试功能。

（五）在主菜单，输入 4，删除员工功能：

```
请选择操作菜单(0-6): 4
1.按编号删除。
2.按姓名删除。
3.退出删除界面。
```

按编号删除，测试如下：

```
请选择操作菜单(1-3): 1
请输入你要删除的编号: 4
删除成功!
```

按姓名删除如下：

```
请选择操作菜单(1-3): 2
请输入你要查询的姓名(以#号键结束): admin2#
删除成功!
```

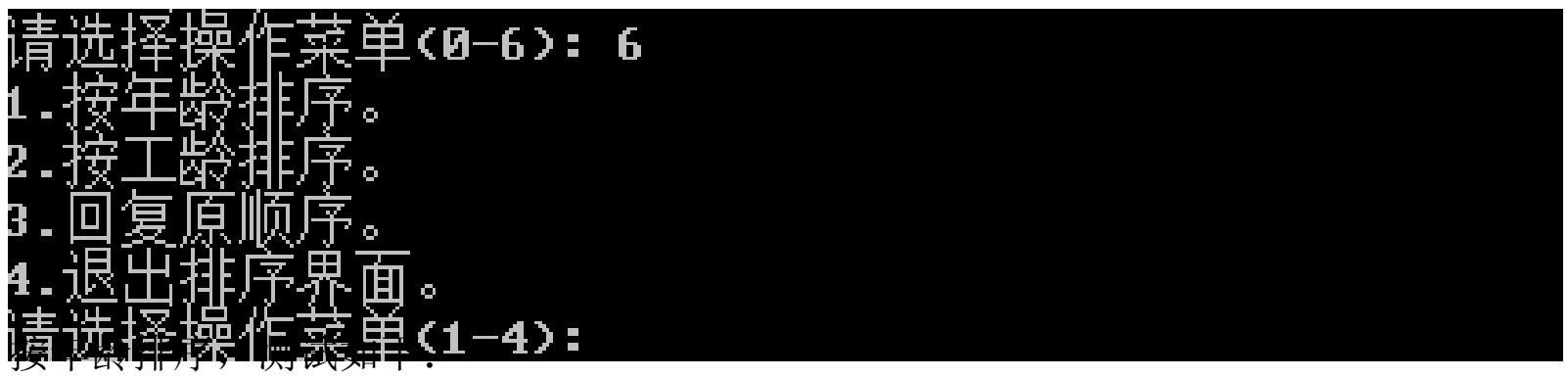
以上测试完成。

（六）在主菜单输入 5，进入修改信息功能：

```
请选择操作菜单(0-6): 5
1.修改整条记录。
2.修改部分。
3.退出修改界面。
请选择操作菜单(1-3): 1
请输入你要修改信息的编号: 1
请输入员工的姓名(以#号键结束): abc#
请输入员工的年龄: 6
请输入员工的职位(以#号键结束): 文秘#
请输入员工的工龄: 5
修改成功!
```

上面是对每一条记录进行的修改。当你选择 2 的菜单时，则是对某条信息的某个字段对其进行内容修改，在这里就不做演示了。

（七）在主菜单输入 6，进入排序功能：



按工龄排序相同操作，测试成功。回复排序，回复到未排序状态，测试成功。

（八）退出，将所有的员工信息写入 message.txt 文件中，实现永久保存。退出系统，测试完成。

总结

在本系统的开发过程中由于时间也比较仓促、准备不充分，系统必然会存在一些缺陷和不足。对员工信息管理的整个流程不够熟悉，在需求分析时未能做到完全满足用户的需求。

课程设计中我们遇到很多问题。我们在开发时，我们查阅了许多资料，了解到即是对员工的编号号，姓名，年龄，工龄等复杂多样的信息，能够较清晰，快捷而操作方便的现代化管理系统。弄清了这个基本概念以后，我们又详细理解了老师所讲的设计要求和注意事项，大致确定了总体的设计思路，初步提出问题的解决方案，以及系统大致设计方案和框架，接下来我们就着手编程。在编程过程中，先是根据系统所要求，找出所需要知识点。编完程序，我们在机房进行了一次又一次的调试，找出了其中的错误，一一纠正，并且修改了其中不太完善的部分，力求做到实用并且精确

尽管本管理系统存在着很多不足，但其功能全面、易于日后程序更新、数据库管理容易、界面友好、操作方便、效率高、安全性好等优点是本管理系统所必需的。通过开发这个系统，我组掌握了的项目基本开发过程，用到的知识巩固了我对 C 语言的学习，但在这次设计中的最大收获并不是掌握这几门开发工具的应用，而是学会了设计系统的思维方法。

通过本次课程设计，我们对数据结构知识掌握了很多，并能将它用以程序编写中，并且提高了自己的解决实际问题的能力。

主要参考文献

[1]李云清，杨庆红，揭安全 .数据结构（ C 语言版）[M].北京:人民邮电大学出版社,2004.6

[2]潘彦.算法设计与分析基础[M]. 北京:清华大学出版社,2007.1

[3]软件工程原理与应用/曾强聪，赵歆编著 北京:清华大学出版社,2011

[4] 吕凤翥.C++语言程序设计（第 2 版）[M].北京:电子工业出版社,2007.2

[5] 严蔚敏，吴伟民.数据结构（C 语言版）[M].北京:清华大学出版社,2002.9

附录：源代码

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
#include <stdlib.h>

#define MAX_NUM 40

typedef struct{
    int num; //编号
    char name[MAX_NUM];//姓名
    int age; //年龄
    char job[MAX_NUM];//职位;
    int workTime;// 工龄
}People;

typedef struct node{
    People people;
    struct node * next;
    int len;//表示链表长度
}linklist;


void doAddMessage(linklist * s){
    int i=0,value1=0,j;
    char ch;
    printf("\t请输入员工的姓名(以#号键结束)：");
    scanf("%c",&ch);
    while(ch!='#'){
        s->people.name[i]=ch;
        i++;
        scanf("%c",&ch);
    }
    for(j=i;j<40;j++){
        s->people.name[j]='\0';
    }
    getchar();
    printf("\t请输入员工的年龄：");
    scanf("%d",&value1);
    s->people.age=value1;

    getchar();
    printf("\t请输入员工的职位(以#号键结束)：");
```

```
i=0;
scanf("%c",&ch);
while(ch!='#'){
    s->people.job[i]=ch;
    i++;
    scanf("%c",&ch);
}
for(j=i;j<40;j++){
    s->people.job[j]='\0';
}

getchar();
printf("\t请输入员工的工龄: ");
scanf("%d",&value1);
    s->people.workTime=value1;
}


//执行修改某个字段的信息
void doEditSome(linklist * p){
    int i=0,max=0,value1=0,j;
    int flag=0;
    char str1[4],str2[40],ch;
    char a[]="姓名";
    char b[]="年龄";
    char c[]="职位";
    char d[]="工龄";
    getchar();
    printf("\t请输入你要修改的字段名称(以#号键结束):");
    scanf ("%c",&ch);
    while(ch!='#'){
        str1[i]=ch;
        i++;
        scanf("%c",&ch);
}

for(i=0;i<strlen(a);i++){
    if(a[i]==str1[i])
        flag=1;
    else{
        flag=0;
        break;
    }
}
```

```
if(flag==0){
    for(i=0;i<strlen(b);i++){
        if(b[i]==str1[i])
            flag=2;
        else{
            flag=0;
            break;
        }
    }
}
if(flag==0){
    for(i=0;i<strlen(c);i++){
        if(c[i]==str1[i])
            flag=3;
        else{
            flag=0;
            break;
        }
    }
}
if(flag==0){
    for(i=0;i<strlen(d);i++){
        if(d[i]==str1[i])
            flag=4;
        else{
            flag=0;
            break;
        }
    }
}
getchar();
if(flag==0){
    printf("\t没有找到您要修改的字段！\n");
}else if(flag==1){
    printf("\t请输入该字段的值(以#号键结束):");
    i=0;
    scanf ("%c",&ch);
    while(ch!='#'){
        str2[i]=ch;
        i++;
        scanf("%c",&ch);
    }
    for(j=i;j<40;j++){
        str2[j]='\0';
```

```
        }
        for(i=0;i<40;i++){
            p->people.name[i]=str2[i];
        }
    }else if(flag==2){
        printf("\t请输入该字段的值:");
        scanf("%d",&value1);
        p->people.age=value1;
    }else if(flag==3){
        printf("\t请输入该字段的值(以#号键结束):");
        i=0;
        scanf ("%c",&ch);
        while(ch!='#'){
            str2[i]=ch;
            i++;
            scanf("%c",&ch);
        }
        for(j=i;j<40;j++){
            str2[j]='\0';
        }
        for(i=0;i<40;i++){
            p->people.job[i]=str2[i];
        }
    }else{
        printf("\t请输入该字段的值:");
        scanf("%d",&value1);
        p->people.workTime=value1;
    }
}


//按编号查询员工信息
void searchPeopleByNum(linklist * head){
    int number,flag=0;
    linklist * p;
    printf("\t请输入你要查询的编号：");
    scanf("%d",&number);
    p=head->next;
    while(p!=NULL){
        if(p->people.num==number){
            printf("\t编号      姓名                年龄      职位                工龄
\n");

    printf("\t%-10d%-20s%-10d%-20s%-10d",p->people.num,p->people.name,p->people.age,p->
```

```
people.job,p->people.workTime);
                printf("\n");
                flag=1;
                break;
            }else{
                p=p->next;
            }
        }
        if(flag==0){
            printf("\t没有查到与你输入编号相匹配的员工信息！\n");
        }
}

//按照姓名查找员工信息
void searchPeopleByName(linklist * head){
    int j=0, i=0,flag=0;
    char a,ch[40];
    linklist * p;
    p=head->next;
    getchar();
    printf("\t请输入你要查询的姓名(以#号键结束)：");
    scanf("%c",&a);
    while(a!='#'){
        ch[i]=a;
        i++;
        scanf("%c",&a);
    }
    while(p!=NULL){
        for(j=0;j<strlen(ch);j++){
            if(p->people.name[j]!=ch[j])
                break;
        }
        if(i==j){
            printf("\t编号      姓名                年龄      职位                工龄
\n");
    printf("\t%-10d%-20s%-10d%-20s%-10d",p->people.num,p->people.name,p->people.age,p->
people.job,p->people.workTime);
                printf("\n");
                flag=1;
                break;
            }
            else
                p=p->next;
        }
```

```
    if(flag==0){
        printf("\t没有查到与你输入姓名相匹配的员工信息！\n");
    }
}

//根据编号删除员工信息
void deletePeopleByNum(linklist * head){
    int number,flag=0;
    linklist * p,* q;
    printf("\t请输入你要删除的编号：");
    scanf("%d",&number);
    p=head->next;
    q=head;
    while(p!=NULL){
        if(p->people.num==number){
            q->next=p->next;
            free(p);
            flag=1;
            break;
        }else{
            q=q->next;
            p=p->next;
        }
    }
    if(flag==0){
        printf("\t没有查到与你输入编号相匹配的员工信息！\n");
    }else{
        printf("\t删除成功！\n");
    }
}

//根据姓名删除 员工信息
void deletePeopleByName(linklist * head){
    int j=0, i=0,flag=0;
    char a,ch[40];
    linklist * p,*q;
    p=head->next;
    q=head;
    getchar();
    printf("\t请输入你要查询的姓名(以#号键结束)：");
    scanf("%c",&a);
    while(a!='#'){
```

```
        ch[i]=a;
        i++;
        scanf("%c",&a);
    }
    while(p!=NULL){
        for(j=0;j<strlen(ch);j++){
            if(p->people.name[j]!=ch[j])
                break;
        }
        if(i==j){
            q->next=p->next;
            free(p);
            flag=1;
            break;
        }
        else{
            q=q->next;
            p=p->next;
        }
    }
    if(flag==0){
        printf("\t没有查到与你输入姓名相匹配的员工信息！\n");
    }else{
        printf("\t删除成功！\n");
    }
}

//修改
void edit(linklist * head,char ch){
    int number,flag=0;
    linklist * p,* q;
    printf("\t请输入你要修改信息的编号：");
    scanf("%d",&number);
    p=head->next;
    q=head;
    while(p!=NULL){
        if(p->people.num==number){
            if(ch=='1'){
                doAddMessage(p);
            }
            if(ch=='2'){
                doEditSome(p);
            }
```

```
                flag=1;
                break;
            }else{
                q=q->next;
                p=p->next;
            }
        }
        if(flag==0){
            printf("\t没有查到与你输入编号相匹配的员工信息！\n");
        }else{
            printf("\t修改成功！\n");
        }
}
//实现 三种条件的排序
void sortAll(linklist * head,char a){
    linklist * p,* q,*s;
    int flag=0;
    int i=0;
    for(i=0;i<head->len;i++){
        flag=0;
        p=head;
        q=p->next;
        s=q->next;
        while(s!=NULL){
            if((q->people.age < s->people.age)&&a=='1'){
                q->next=s->next;
                s->next=q;
                p->next=s;
                flag=1;
            }
            if((q->people.workTime < s->people.workTime)&&a=='2'){
                q->next=s->next;
                s->next=q;
                p->next=s;
                flag=1;
            }
            if((q->people.num > s->people.num)&&a=='3'){
                q->next=s->next;
                s->next=q;
                p->next=s;
                flag=1;
            }
            p=p->next;
```

```
                q=p->next;
                s=q->next;
            }
            if(flag==0)
                break;
        }
        printf("\t排序完成，请继续操作！\n");
}

//添加员工信息
void addMessage(linklist * head){
    char ch;
    int i=0,value1;
    linklist * s,* p;
    s=(linklist *)malloc(sizeof(linklist));
    s->people.num=head->len+1;
    printf("\n\t员工的编号为：%d\n",s->people.num);
    doAddMessage(s);
    p=head;
    while(p->next!=NULL){
        p=p->next;
    }
    s->next=p->next;
    p->next=s;
    head->len=head->len +1;
    getchar();

    printf("\t是否继续添加(是、否/y、n)：");
    scanf("%c",&ch);
    getchar();
    if(ch=='y'||ch=='Y'){
        addMessage(head);
        getchar();
    }
}

//展示所有员工的信息
void showAllMessage(linklist * head){
    linklist * p;
    p=head->next;
    printf("\t编号      姓名                年龄      职位              工龄\n");
    while(p!=NULL){

    printf("\t%-10d%-20s%-10d%-20s%-10d",p->people.num,p->people.name,p->people.age,p->
```

```
people.job,p->people.workTime);
            p=p->next;
            printf("\n");
        }
        printf("\n");
}

//查询员工信息
void searchPeople(linklist * head){
        char ch='y';
        if(head->next==NULL){
            printf("\t目前暂无存储任何人员信息，无法进行任何查询操作！\n");
        }else{
            printf("\t1.按编号查询。\n");
            printf("\t2.按姓名查询。\n");
            printf("\t3.退出查询界面。\n");
            printf("\t*************************************************\n");
            printf("\t请选择操作菜单(1-3)：");
            scanf("%c",&ch);
            switch(ch){
                case '1':
                    searchPeopleByNum(head);
                    break;
                case '2':
                    searchPeopleByName(head);
                    break;
                case '3':break;
            }
        }
        getchar();
}

//删除员工信息
void deletePeople(linklist * head){
        char ch='y';
        if(head->next==NULL){
            printf("\t目前暂无存储任何人员信息，无法进行任何删除操作！\n");
        }else{

            printf("\t1.按编号删除。\n");
            printf("\t2.按姓名删除。\n");
            printf("\t3.退出删除界面。\n");
            printf("\t*************************************************\n");
```

```
            printf("\t请选择操作菜单(1-3): ");
            scanf("%c",&ch);
            switch(ch){
                case '1':
                    deletePeopleByNum(head);
                    break;
                case '2':
                    deletePeopleByName(head);
                    break;
                case '3':break;
            }
        }
        getchar();
}

//修改员工信息
void editMessage(linklist * head){
    char ch='y';
    if(head->next==NULL){
        printf("\t目前暂无存储任何人员信息，无法进行任何删除操作！\n");
    }else{

        printf("\t1.修改整条记录。\n");
        printf("\t2.修改部分。\n");
        printf("\t3.退出修改界面。\n");
        printf("\t*****************************************************\n");
        printf("\t请选择操作菜单(1-3): ");
        scanf("%c",&ch);
        switch(ch){
            case '1':
                edit(head,ch);
                break;
            case '2':
                edit(head,ch);
                break;
            case '3':break;
        }
    }
    getchar();
}

//单链表排序
void sort(linklist * head){
    char ch='y';
```

```
if(head->next==NULL){
    printf("\t目前暂无存储任何人员信息，无法进行任何排序操作！\n");
}else if(head->next->next==NULL){
    printf("\t只有一条数据无需进行排序！\n");
}else{

    printf("\t1.按年龄排序。\n");
    printf("\t2.按工龄排序。\n");
    printf("\t3.恢复原顺序。\n");
    printf("\t4.退出排序界面。\n");
    printf("\t*****************************************************\n");
    printf("\t请选择操作菜单(1-4): ");
    scanf("%c",&ch);
    switch(ch){
            case '1':
                sortAll(head,ch);
                break;
            case '2':
                sortAll(head,ch);
                break;
            case '3':
                sortAll(head,ch);
                break;
            case '4':break;
        }
    }
    getchar();
}

//退出时保存将文件保存到文件中
void saveMessage(linklist * head){
    FILE * fp;
    linklist *p,*p0;
    fp=fopen("message.txt","wb");
    p=head;
    if(fp==NULL){
       printf("\n\t文件保存失败!\n请重新启动本系统...\n");
       exit(0);
    }
    while(p!=NULL){
        //将链表中的信息写入文件中
       if(fwrite(p,sizeof(linklist),1,fp)!=1){
             printf("\n\t写入文件失败!\n请重新启动本系统!\n");
       }
```

```
            p0=p;
            p=p->next;
            free(p0);
        }
        head=NULL;
        fclose(fp);
}
//登录时加载信息
void loadAddMessage(linklist * head){
    FILE *fp;
    linklist *p1,*p2,*p3;
    int flag=0;
    fp=fopen("message.txt","rb");
    p1=(linklist *)malloc(sizeof(linklist));
    fread(p1,sizeof(linklist),1,fp);
    p3 = p2=head;
    //读出信息,重新链入链表
    while(! feof(fp)){
        p1=(linklist *)malloc(sizeof(linklist));
        fread(p1,sizeof(linklist),1,fp);
        p2->next=p1;
        p3=p2;
        p2=p1;
        flag=1;
    }
    p3->next=NULL;
    if(p3->people.num < 1){
          head->len=0;
          head->next=NULL;
    }else if(flag==1){
        head->len=p3->people.num;
    }else{
          head->len=0;
    }
    free(p1);
    fclose(fp);
}
//主函数
int main(){
     char ch='y';
     linklist * head;
     head =(linklist *)malloc(sizeof(linklist));
     head->len =0;
     head->next=NULL;
```

```
loadAddMessage(head);
printf("\t\t\t\t员工管理系统\n");
printf("\t*************************************************\n");
do{
    printf("\t\t\t\t操作菜单\n");
    printf("\t1.添加员工\t2.展示所有的员工信息\t3.查询员工信息\n");
    printf("\t4.删除员工\t5.修改员工信息       \t6.排序\n");
    printf("\t0.退出系统\n");
    printf("\t*************************************************\n");
    printf("\t请选择操作菜单(0-6): ");
    scanf("%c",&ch);
    getchar();
    switch(ch){
        case '1':
            addMessage(head);
            break;
        case '2':
            showAllMessage(head);
            break;
        case '3':
            searchPeople(head);
            break;
        case '4':
            deletePeople(head);
            break;
        case '5':
            editMessage(head);
            break;
        case '6':
            sort(head);
            break;
        case '0':
                saveMessage(head);
                printf("\t保存成功! ");
                exit(0);
            }
    }while(ch!='0');
    getchar();
    return 0;
}
```