第 3 章

学生成绩管理系统的设计与实现

随着科学的发展和社会的进步，许多过去由人工处理的繁杂事务开始交付计算机来完成。学生成绩管理系统可以说是每个教育单位的得力助手，它利用计算机对学生成绩进行统一管理，实现学生成绩信息管理工作流程的系统化、规范化和自动化，提高了广大教师的工作效率。因此，学生成绩管理系统对教育部门或单位起着越来越重要的作用。

3.1 设计目的

本程序旨在训练读者的基本编程能力，了解管理信息系统的开发流程，熟悉 C 语言的文件和单链表的各种基本操作。本程序中涉及结构体、单链表、文件等方面的知识。通过本程序的训练，使读者能对 C 语言的文件操作有更深刻的了解，掌握利用单链表存储结构实现对学生成绩管理的原理，为进一步开发出高质量的信息管理系统打下坚实的基础。

3.2 功能描述

如图 3-1 所示，此成绩管理系统主要利用单链表实现，它由如下五大功能模块组成。

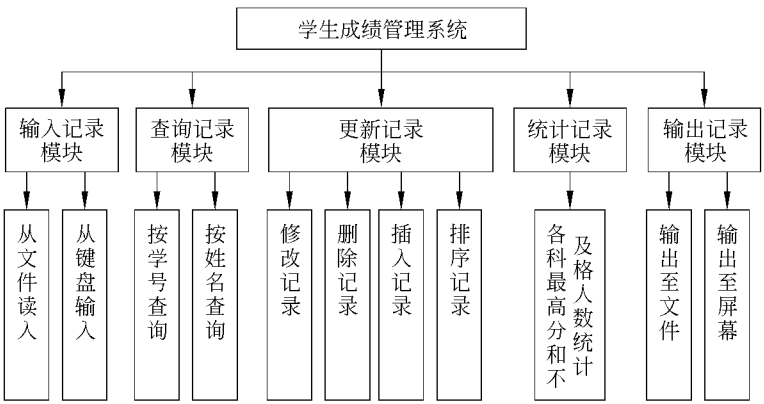

图 3-1 学生成绩管理系统功能模块图

(1) 输入记录模块。输入记录模块主要完成将数据存入单链表中的工作。在此成绩管理系统中，记录可以从以二进制形式存储的数据文件中读入，也可从键盘逐个输入学生

记录。学生记录由学生的基本信息和成绩信息字段构成。当从数据文件中读入记录时，它就是在以记录为单位存储的数据文件中，将记录逐条复制到单链表中。

(2) 查询记录模块。查询记录模块主要完成在单链表中查找满足相关条件的学生记录。在此成绩管理系统中，用户可以按照学生的学号或姓名在单链表中进行查找。若找到该学生的记录，则返回指向该学生记录的指针；否则，返回一个值为 NULL 的空指针，并打印出未找到该学生记录的提示信息。

(3) 更新记录模块。更新记录模块主要完成对学生记录的维护。在此成绩管理系统中，它实现了对学生记录的修改、删除、插入和排序操作。一般而言，系统进行了这些操作之后，需要将修改的数据存入源数据文件。

(4) 统计记录模块。统计记录模块主要完成对各门功课最高分和不及格人数的统计。

(5) 输出记录模块。输出记录模块主要完成两个任务。第一，它实现对学生记录的存盘操作，即将单链表中的各节点中存储的学生记录信息写入数据文件中。第二，它实现将单链表中存储的学生记录信息以表格的形式在屏幕上打印出来。

3.3　总体设计

3.3.1　功能模块设计

1. 主控 main()函数执行流程

本成绩管理系统执行主流程如图 3-2 所示。它先以可读写的方式打开数据文件，此文件默认为 c:\stMmt，若该文件不存在，则新建此文件。当打开文件操作成功后，从文件中一次读出一条记录，添加到新建的单链表中，然后执行显示主菜单和进入主循环操作，进行按键判断。

在判断键值时，有效的输入为 0～9 之间的任意数值，其他输入都被视为错误按键。若输入为 0(即变量 select=0)，它会继续判断是否在对记录进行了更新操作之后进行了存盘操作，若未存盘，则全局变量 savefag=1，系统会提示用户是否需要进行数据存盘操作，用户输入 Y 或 y，系统会进行存盘操作。最后，系统执行退出成绩管理系统的操作。

若选择 1，则调用 Add()函数，执行增加学生记录操作；若选择 2，则调用 Del()函数，执行删除学生记录操作；若选择 3，则调用 Qur()函数，执行查询学生记录操作；若选择 4，则调用 Modify() 函数，执行修改学生记录操作；若选择 5，则调用 Insert()函数，执行插入学生记录操作；若选择 6，则调用 Tongji()函数，执行统计学生记录操作；若选择 7，则调用 Sort()函数，执行按降序排序学生记录的操作；若选择 8，则调用 Save()函数，执行将学生记录存入磁盘中的数据文件的操作；若选择 9，则调用 Disp()函数，执行将学生记录以表格形式打印输出至屏幕的操作；若输入为 0～9 之外的值，则调用 Wrong()函数，给出按键错误的提示。

2. 输入记录模块

输入记录模块主要实现将数据存入单链表中。这部分的操作较为简单。当从数据文

件中读出记录时,它调用了 fread(p,sizeof(Node),1,fp)文件读取函数,执行一次从文件中读取一条学生成绩记录信息存入指针变量 p 所指的节点中的操作,并且这个操作在 main()中执行,即当成绩管理系统进入显示菜单界面时,该操作已经执行了。若该文件中没有数据,系统会提示单链表为空,没有任何学生记录可操作,此时,用户应选择 1,调用 Add()函数,进行学生记录的输入,即完成在单链表 1 中添加节点的操作。值得一提的是,这里的字符串和数值的输入分别采用函数来实现,在函数中完成输入数据任务,并对数据进行条件判断,直到满足条件为止,这样大大减少了代码的重复和冗余,符合模块化程序设计的特点。

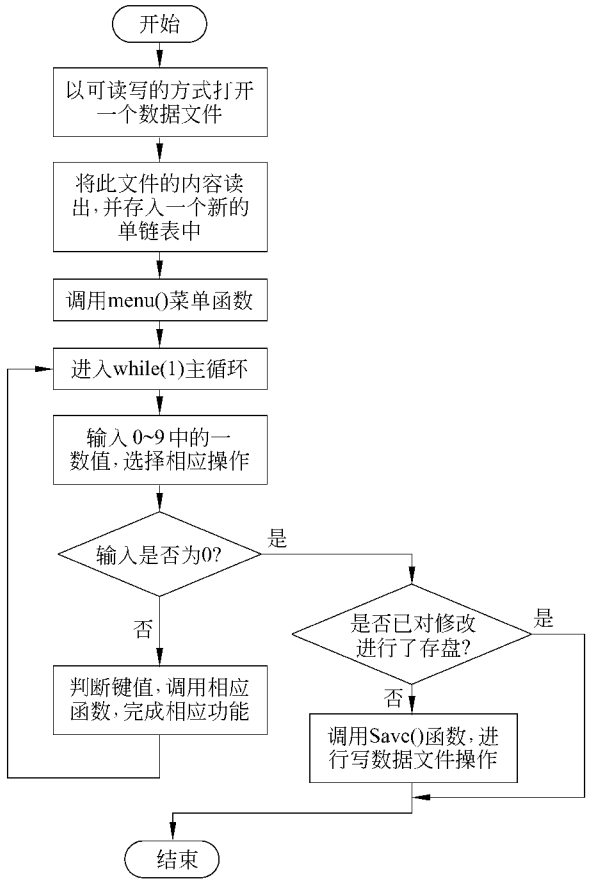

图 3-2 主控函数执行流程图

3. 查询记录模块

查询记录模块主要实现了在单链表中按学号或姓名查找满足相关条件的学生记录。在查询函数 Qur()中,l 为指向保存了学生成绩信息的单链表的首地址的指针变量。为了遵循模块化编程的原则,我们将在单链表中进行的指针定位操作设计成了一个单独的函数 Node * Locate(Link,l,char findmess[],cha nameornum[]),参数 finemess[]保存要查找的具体内容,nameornum[]保存要查找的字段(值为字符串类型的 num 或者 name),若找到该记录,则返回指向该节点的指针;否则,返回一个空指针。

4. 更新记录模块

此模块主要实现了对学生记录的修改、删除、插入和排序操作。因为学生记录是以单链表的结构形式存储的，所以这些操作都在单链表中完成。下面分别介绍这 4 个功能模块。

(1) 修改记录。

修改记录操作需要对单链表中目标节点的数据域中的值进行修改，它分两步完成。第一步，输入要修改的学号，输入后调用定位函数 Locate()在单链表中逐个对节点数据域中学号字段的值进行比较，直到找到该学号的学生记录；第二步，若找到该学生记录，修改除学号之外的各字段的值，并将存盘标记变量 saveflag 置 1，表示已经对记录进行了修改，但还未执行存盘操作。

(2) 删除记录。

删除记录操作完成删除指定学号或姓名的学生记录，它也分两步完成。第一步，输入要删除的学号或姓名，输入后调用定位函数 Locate()在单链表中逐个对节点数据域中学号或姓名字段的值进行比较，直到找到该学号或姓名的学生记录，返回指向该学生记录的节点指针；第二步，若找到该学生记录，将该学生记录所在节点的前驱节点的指针域指向目标节点的后继节点。

(3) 插入记录。

插入记录操作完成在指定学号的随后位置插入新的学生记录。首先，它要求用户输入某个学生的学号，新的记录将插入在该学生记录之后；然后，提示用户输入一条新的学生记录的信息，这些信息保存在新节点的数据域中；最后，将该节点插入在指定位置学号之后。它的具体插入执行过程如图 3-3 所示，图中 q 为位置学号所在节点的指针变量，其中，p 为 q 所指节点的后继节点的指针变量，q->next=p，指针变量 i 指向新记录所在的节点，为插入节点 i，依次执行的操作为：i->next=q->next；q->next=i。

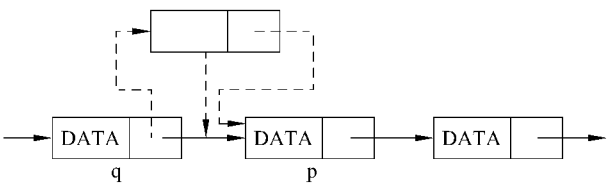

图 3-3　单链表中插入学生记录节点示意图

(4) 排序记录。

有关排序的算法有很多，如冒泡排序、插入排序等。针对单链表结构的特点，可采用插入排序算法实现按总分的从高到低对学生记录进行排序，排序完成之后，即可按顺序给名次字段赋值。

在单链表中，实现插入排序的基本步骤如下。

① 新建一个单链表 1，用来保存排序结果，其初始值为待排序单链表中的头节点。

② 从待排序链表中取出下一个节点，将其总分字段值与单链表 1 中的各节点中总分字段的值进行比较，直到在链表 1 中找到总分小于它的节点。若找到如此节点，系统将待

排序链表中取出的节点插入此节点前,作为其前驱。否则,将取出的节点放在单链表 1 的尾部。

③ 重复第②步,直到从待排序链表取出的节点的指针域为 NULL,即此节点为链表的尾部节点,排序完成。

5. 统计记录模块

该模块的实现比较简单,它主要通过循环读取指针变量 p 所指的当前节点的数据域中各字段的值,并对各个成绩字段进行逐个判断的形式,完成单科最高分学生的查找和各科不及格人数的统计。

6. 输出记录模块

当把记录输出至文件时,调用 fwrite(p,sizeof(Node),1,fp)函数,将 p 指针所指节点中的各字段值,写入文件指针中所指的文件。当把记录输出至屏幕时,调用 void Disp (Link 1)函数,将单链表 1 中存储的学生记录信息以表格的形式在屏幕上打印出来。

3.3.2 数据结构设计

1. 学生成绩信息结构体

```
typedef struct student                    /*标记为 student*/
{
char num[10];                             /*学号*/
char name[15];                            /*姓名*/
int cgrade;                               /*C 语言成绩*/
int mgrade;                               /*数学成绩*/
int egrade;                               /*英语成绩*/
int total;                                /*总分*/
float ave;                                /*平均分*/
int mingci;                               /*名次*/
};
```

结构体 student 将用于存储学生的基本信息,它将作为单链表的数据域。为了简化程序,这里只取了 3 门成绩。其各字段的值的含义如下。

num[10]:保存学号。

Name[15]:保存姓名。

cgrade:保存 C 语言成绩。

mgrade:保存数学成绩。

egrade:保存英语成绩。

total:保存总分。

ave:保存平均分。

mingic:保存名次。

2. 单链表 node 结构体

```
typedef struct node
```

```
{
struct student data;      /* 数据域 */
struct node * next;       /* 指针域 */
}Node, * Link;            /* Node 为 node 类型的结构变量, * Link 为 node 类型的指针变量 */
```

这样定义了一个单链表的结构，结构标记为 node，data 为 student 结构类型的数据，作为单链表结构中的数据域，next 为单链表中的指针域，用来存储其直接后继节点的地址。Node 为 node 类型的结构变量，* Link 为 node 类型的指针变量。

3.3.3　函数功能描述

1. Printheader()

函数原型为 void printheader()，Printheader()函数用于在以表格形式显示学生记录时，打印输出表头信息。

2. printdata()

函数原型为 void printdata(Node * pp)，Printdata()函数用于在以表格形式显示学生记录时，打印输出单链表 pp 中的学生信息。

3. stringinput()

函数原型为 void stringinput(char * t,int lens,char * notice)，Stringinput()函数用于输入字符串，并进行字符串长度验证(长度＜lens)。t 用于保存输入的字符串，因为是以指针形式传递的，所以 t 相当于该函数的返回值。notice 用于保存 printf()中输出的提示信息。

4. NumberinPut()

函数原型为 int NumberinPut(char * notice)，NumberinPut()函数用于输入数值型数据，notice 用于保存 printf()中输出的提示信息，该函数返回用户输入的整型数据。

5. Disp()

函数原型为 void Disp(Link 1)，Disp()函数用于显示单链表 1 中存储的学生记录，内容为 student 结构中定义的内容。

6. Locate()

函数原型为 Node * Locate(Link 1,char findmess[],char nameornum[])，Locae()函数用于定位链表中符合要求的节点，并返回指向该节点的指针。参数 findmess[]保存要查找的具体内容，nameornum[]保存按什么字段在单链表 1 中查找。

7. Add()

函数原型为 void Add(Link 1)，Add()函数用于在单链表 1 中增加学生记录的节点。

8. Qur()

函数原型为 void Qur(Link 1)，Qur()函数用于在单链表 1 中按学号或姓名查找满足条件的学生记录，并显示出来。

9. Del()

函数原型为 void Del(Link 1)，Del()函数用于先在单链表 1 中找到满足条件的学生记录的节点，然后删除该节点。

10. Modify()

函数原型为 Void Modify(Link 1),Modify()函数用于在单链表 1 中修改学生记录。

11. Insert()

函数原型为 void Insert(Link 1),Insert()函数用于在单链表 1 中插入学生记录。

12. Tongji()

函数原型为 void Tongji(Link 1),Tongji()函数用于在单链表 1 中完成学生记录的统计工作,统计该班的总分第一名、单科第一名和各科不及格人数。

13. Sort()

函数原型为 Void Sort(Link 1),Sort()函数用于在单链表 1 中完成利用插入排序算法实现单链表的按总分字段的降序排序。

14. Save()

函数原型为 void Save(Link 1),Save()函数用于将单链表 1 中的数据写入磁盘中的数据文件。

15. 主函数 main()

整个成绩管理系统控制部分,其详细说明可参考图 3-2。

3.4 程序实现

3.4.1 源码分析

1. 程序预处理

包括加载头文件,定义结构体、常量和变量,并对它们进行初始化工作。

```
#include "stdio.h"                /*标准输入输出函数库*/
#include "stdlib.h"               /*标准函数库*/
#include "string.h"               /*字符串函数库*/
#include "conio.h"                /*屏幕操作函数库*/
#define                                  HEADER1 "
--------------------STUDENT--------------------  \n"
#define HEADER2 "    |    number    |      name     |Comp|Math|Eng |   sum   |
ave  |mici | \n"
#define                                   HEADER3 "
|-----------|----------|----|----|--------|-------|-----| "
#define FORMAT  "        |    %-10s |%-15s|%4d|%4d|%4d| %4d   | %.2f |%4d |\n"
#define                              DATA
p->data.num,p->data.name,p->data.egrade,p->data.mgrade,p->data.cgrade,p->
data.total,p->data.ave,p->data.mingci
#define END     "      ---------------------------  \n"

int saveflag=0;                          /*是否需要存盘的标志变量*/
/*定义与学生有关的数据结构*/
typedef struct student                   /*标记为 student*/
```

```
{
char num[10];                          /*学号*/
char name[15];                         /*姓名*/
int cgrade;                            /*C语言成绩*/
int mgrade;                            /*数学成绩*/
int egrade;                            /*英语成绩*/
int total;                             /*总分*/
float ave;                             /*平均分*/
int mingci;                            /*名次*/
};

/*定义每条记录或节点的数据结构,标记为: node*/
typedef struct node
{
struct student data;                   /*数据域*/
struct node *next;                     /*指针域*/
}Node, *Link;          /*Node为node类型的结构变量, *Link为node类型的指针变量*/
```

2. 主函数 main()

main()函数主要实现对整个程序的运行控制,以及相关功能模块的调用,详细分析可参考图 3-2。

```
void main()
{

  Link l;                              /*定义链表*/
  FILE *fp;                            /*文件指针*/
  int select;                          /*保存选择结果变量*/
  char ch;                             /*保存(y,Y,n,N)*/
  int count=0;                         /*保存文件中的记录条数(或节点个数)*/
  Node *p, *r;                         /*定义记录指针变量*/


  l=(Node *)malloc(sizeof(Node));
  if(!l)
   {
      printf("\n allocate memory failure ");      /*如没有申请到,打印提示信息*/
      return ;                                    /*返回主界面*/
   }
  l->next=NULL;
  r=l;
  fp=fopen("C:\\student","ab+");
           /*以追加方式打开一个二进制文件,可读可写,若此文件不存在,会创建此文件*/
  if(fp==NULL)
  {
```

```
    printf("\n=====>can not open file!\n");
    exit(0);
  }

while(!feof(fp))
{
  p=(Node*)malloc(sizeof(Node));
  if(!p)
   {
      printf(" memory malloc failure!\n");  /*没有申请成功*/
      exit(0);                              /*退出*/
   }

  if(fread(p,sizeof(Node),1,fp)==1)         /*一次从文件中读取一条学生成绩记录*/
  {
   p->next=NULL;
   r->next=p;
   r=p;                                     /*r指针向后移一个位置*/
   count++;
   }
}

fclose(fp);                                 /*关闭文件*/
printf("\n=====>open file sucess,the total records number is : %d.\n",count);
menu();
while(1)
{
   system("cls");
   menu();
   p=r;
   printf("\n          Please Enter your choice(0~9):");    /*显示提示信息*/
   scanf("%d",&select);

  if(select==0)
  {
   if(saveflag==1)          /*若对链表的数据有修改且未进行存盘操作,则此标志为1*/
   { getchar();
     printf("\n=====>Whether save the modified record to file?(y/n):");
     scanf("%c",&ch);
     if(ch=='y'||ch=='Y')
       Save(l);
   }
   printf("=====>thank you for useness!");
   getchar();
```

```
    break;
    }

    switch(select)
    {
    case 1:Add(l);break;                          /*增加学生记录*/
    case 2:Del(l);break;                          /*删除学生记录*/
    case 3:Qur(l);break;                          /*查询学生记录*/
    case 4:Modify(l);break;                       /*修改学生记录*/
    case 5:Insert(l);break;                       /*插入学生记录*/
    case 6:Tongji(l);break;                       /*统计学生记录*/
    case 7:Sort(l);break;                         /*排序学生记录*/
    case 8:Save(l);break;                         /*保存学生记录*/
    case 9:system("cls");Disp(l);break;           /*显示学生记录*/
    default: Wrong();getchar();break;             /*按键有误,必须为数值 0~9*/
    }
}
}
```

3. 主菜单界面

用户进入成绩管理系统时,需要显示主菜单,提示用户进行选择,完成相应任务。此代码被 main()函数调用。

```
void menu()                              /*主菜单*/
{
system("cls");                           /*调用 DOS 命令,清屏。与 clrscr()功能相同*/
textcolor(10);                           /*在文本模式中选择新的字符颜色*/
gotoxy(10,5);                            /*在文本窗口中设置光标*/
cprintf("                The Students' Grade Management System \n");
gotoxy(10,8);
cprintf("    ************************Menu*********************************\n");
gotoxy(10,9);
cprintf("     *   1 input   record            2 delete record             * \n");
gotoxy(10,10);
cprintf("     *   3 search  record            4 modify record             * \n");
gotoxy(10,11);
cprintf("     *   5 insert  record            6 count  record             * \n");
gotoxy(10,12);
cprintf("     *   7 sort    reord             8 save   record             * \n");
gotoxy(10,13);
cprintf("     *   9 display record            0 quit   system             * \n");
gotoxy(10,14);
cprintf("    ************************************************************\n");
/*cprintf()送格式化输出至文本窗口屏幕中*/
}
```