****

课程设计报告书

**题目：**

**学 院 计算机科学与工程学院**

**专 业 计算机科学与技术1班**

**学生姓名 张炳元**

**学生学号 201730615025**

**指导教师 张见威**

**课程编号 145055**

**课程学分 3.5**

**起始日期 2018.9.3**

|  |  |
| --- | --- |
| 教  师  评  语 | 教师签名：  日期： |
| 成  绩  评  定 |  |
| 备  注 |  |

**课题名称**

**学生成绩管理系统**

一**、选题背景**

说明本课题应解决的主要问题及应达到的技术要求，简述本设计的指导思想。

学生成绩管理对于学校来说是至关重要的，一个良好的学生成绩管理系统能够为学校的老师和学生的管理和学生的查询带来很多方便；以前的都是登记在纸质资料或者运用Excel来登记并且手工完成查找和修改，给大量数据的查找，更新和维护带了不少的困难，工作效率低下。

现在互联网快速发展，计算机科学日渐成熟，计算机科学改善人们的生活条件，提高人们办事效率的观念也越来越被群众接受。

为了快速实现增删查改，有效地操作数据，需要运用数据结构知识来优化功能。由于链表的动态性，所以管理系统主要是用链表来存储数据，并辅助用归并排序和哈希开地址散列法来优化，实现高效快速地增删查改。

二**、方案论证(设计理念)**

**由于链表的动态性，所以管理系统主要是依靠链表用树结构存储数据，并辅助用归并排序和哈希开地址散列法来优化，实现高效快速地增删查改。**

说明设计原理（理念）并进行方案选择，阐明为什么要选择这个设计方案以及所采用方案的特点。

包括：重点说明要实现的功能及其要求、系统的安全性、数据的完整性、应用的运行环境及其性能等要求。

2.1学习成绩管理系统应满足下面功能：

每个学生的信息包括：学号，姓名，修的课程，学分，每门课程的总评成绩等；

支持添加、删除学生功能，将该学生相关的所有信息删除；

支持添加、删除课程的功.能，需要将该课程的所有分数信息都删除；

支持输入某个学生某门课程的分数的功能；

支持分数统计，可统计每门课程的及格率，平均分数，最低分，最高分；

将所有学生的某门功课成绩做成表格输出（可以是简单文本格式)；

图形化界面；

可生成全班学生通讯录文件输出；

可生成某个学生的成绩单文件输出。

2.2数据的完整性

其中学生有三个变量：学号，姓名，电话号码存放在student结构中，并且建立索引链表stuIndex有100个索引项，键值key=sno%100,用开地址哈希散列法来存数据；

课程信息有三个变量：课程号，课程名，学分存放在cnoIndex结构，

并且存在一个指针，用来存放学生的成绩，也就是将同一门的课程的所有学生的数据存放在在一起。每个成绩以学号和课程号为主码。

2.2运行环境：vs2013，win10

三**、过程论述**

重点说明设计是如何实现的，包括：对设计工作的详细表述。要求层次分明、表达确切。

要求：每个图都必须有文字说明，图前说明为什么使用该图、图的主要作用；图后说明图中各成分的作用，和成分之间的交互或图所表达的流程。

3.1主要的结构struct

其中学生有三个变量：学号，姓名，电话号码存放在student结构中，其中主码为学号int sno，并且建立索引链表stuIndex有100个索引项，键值key=sno%100,用开地址哈希散列法来存数据；

struct student

{

long int sno;

string sname;

string pnumber;

student \*next;

};

struct stuIndex

{

int key;

student \*head;

stuIndex \*next;

};

课程信息有三个变量：课程号，课程名，学分存放在cnoIndex结构，

主码为课程号int cno；并且存在一个指针，用来存放学生的成绩，也就是将同一门的课程的所有学生的数据存放在在一起。每个成绩以学号sno和课程号cno为主码。

struct cnoGrade

{

int cno;

int sno;

int grade;

cnoGrade \*next;

};

struct cnoIndex

{

int cno;

string cname;

double credit;

cnoGrade \*head;

cnoIndex \*next;

};

3.2三个头文件

**3.2.1**student.h主要实现和学生个人信息有关的功能

//创建学生的索引

stuIndex\* create\_stuIndex(stuIndex \*head, int n);

//实现从文件中导入数据

stuIndex\* loadStu(stuIndex \*head);

//寻找索引地址

stuIndex\* find(stuIndex \*head, int sno);

//判断学号是否存在，插入和删除时需要判断使用

bool judge\_sno(student \*head, int sno);

//插入学生信息，在头链表中导入结点

void insert\_sno(stuIndex \*head);

//删除学生信息中个人信息部分；

void delete\_sno(stuIndex \*head, int sno);

//输出学生通讯录到文档

void txt\_phone(stuIndex \*head);

//学生信息的更改，可以通过这个文档来查看；

void txt\_stu\_now(stuIndex \*head);

**3.2.2** CS\_grade.h主要实现和课程信息有关的以及学生的成绩

int min(int a, int b);

int max(int a, int b);

//成绩单\_01，将同一个课程的成绩放在cnoGrade中

cnoIndex\* load\_grade(cnoIndex \*head);

//归并排序cnoGrade

cnoGrade \*list\_split(cnoGrade \*head);//实现链表的归并排序之拆分链表,返回后面半段

cnoGrade\* merge(cnoGrade \*h1, cnoGrade \*h2);//合并两个链表

cnoGrade\* sort\_cnoGrade(cnoGrade \*head);//按照学号归并排序，先分，后并

//归并排序cnoIndex

cnoIndex \*list\_split(cnoIndex \*head);//实现链表的归并排序之拆分链表,返回后面半段

cnoIndex\* merge(cnoIndex \*h1, cnoIndex \*h2);//合并两个链表

cnoIndex\* sort\_cnoIndex(cnoIndex \*head);//按照学号归并排序，先分，后并

//创建成绩单\_02，按照课程链接表格

cnoIndex\* load\_cno(cnoIndex \*p);

//删除导入数据的重复

void nocopy\_cs(cnoIndex\* h);

//判断课程号是否存在，在插入和删除时需要用到

bool judge\_cno(cnoIndex \*head, int cno);

void txt\_grade\_now(cnoIndex \*h);

void txt\_cno\_now(cnoIndex \*head);

//6打印单科成绩到文件夹

void txt\_cno(cnoIndex \*h);

//5统计单个课程分数

void groupby(cnoIndex \*head);

//统计课程成绩

void txt\_groupby(cnoIndex \*head);

//2-2-2删除学生成绩cs

void delete\_sno\_grade(cnoIndex \*head, int sno);

//3-2-2删除课程号以及成绩

void delete\_cno(cnoIndex \*head);

//3-1添加课程

void insert\_cno(cnoIndex \*head);

3.2.3UI.h

//实现功能时的界面

void UI();

3.3主程序和代码的运行

初步设定的功能实现和界面

0添加学生

1删除学生

2添加功课

3删除功课

4输入学生某门功课成绩

5统计分数

6将所有功课的学生成绩做成表格输出txt

7生成通讯录

8某个同学的成绩单输出

9统计成绩的文档

10课程信息

11成绩信息

12学生信息

我们要实现的的功能主要有上面所示，在main()函数中的这样实现的

int i;

while (1)

{

UI();

cin >> i;

switch (i)

{

case 0:

insert\_sno(stuhead);

break;

case 1:

delete\_stu(cnohead,stuhead);

break;

case 2:

insert\_cno(cnohead);

break;

case 3:

delete\_cno(cnohead);

break;

case 4:

insert\_grade(cnohead, stuhead);

break;

case 5:

groupby(cnohead);

break;

case 6:

txt\_cno(cnohead);

break;

case 7:

txt\_phone(stuhead);

break;

case 8:

txt\_grade(cnohead,stuhead);

break;

case 9:

txt\_groupby(cnohead);

break;

case 10:

txt\_cno\_now(cnohead);

break;

case 11:

txt\_grade\_now(cnohead);

break;

case 12:

txt\_stu\_now(stuhead);

break;

default:

break;

}

system("pause");

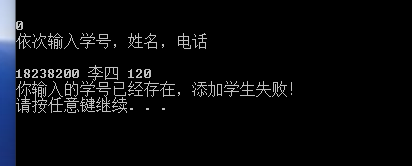
}

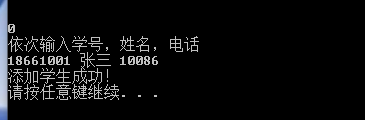
运行之后

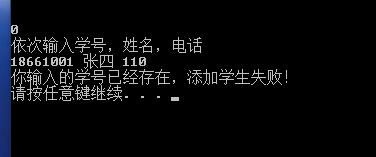


下面是各个功能实现的原理和主要函数以及他们的实现界面

功能0添加学生

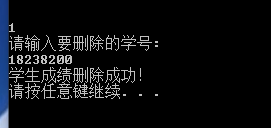


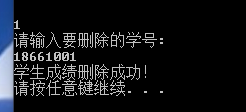


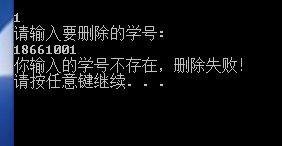


实现对应的函数insert\_sno()，先用judge\_sno()判断学号是否已经存在，存在则添加不了，学号不存在则在对应的索引中插入结点，如上面学号18661001在第一次可以实现插入，在第二次的时候就不行了

功能1删除学生

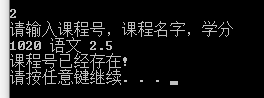


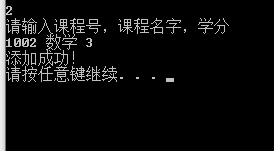




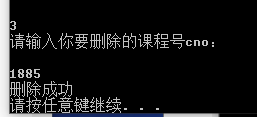
删除学生的函数接口delete\_sno(),在里面同样有一个判断函数judge\_sno(),如果存在的，则删除对应的结点学生信息结点以及对应学生号的成绩；不存在，则会报错；

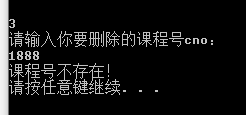
功能2添加功课的原理分别同功能0类似，先判断，再插入；



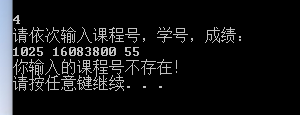


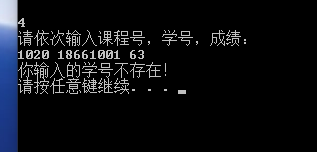
功能3删除功课也有部分类似同功能1类似， 先判断，再删除对应的课程索引结点和里面的课程成绩；

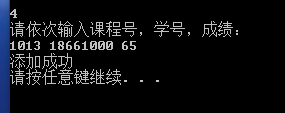


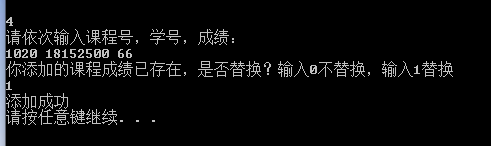


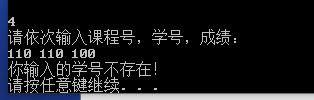
功能4插入学生成绩





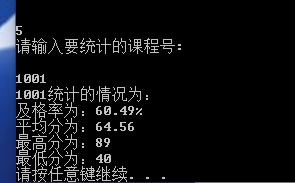


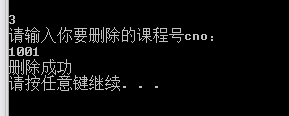


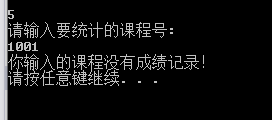


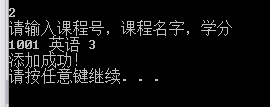
Insert\_grade()函数中有两个判断函数，先执行判断学号是否在student中存在judge\_sno()，如果不存在，直接返回；如果存在，则执行判断课程号是否存在的函数judge\_cno();不存在也返回，课程号存在的号，还要再判断判断在这个课程索引当中是否已经存在这个学生的成绩的，不存在的话，直接添加成功；存在的话，则可以选择是否替换原有的成绩；

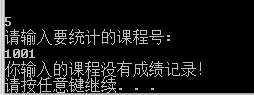
功能5统计分数







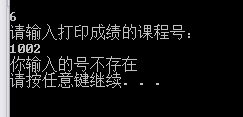


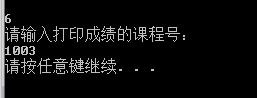


实现的函数是groupby（），在里面先判断输入的课程号是否存在judge\_cno()，不存在则返回，存在则继续判断是否有成绩记录，没有也返回；

其他的6-9是实现的文档形式

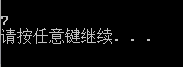
6实现的函数txt\_cno()





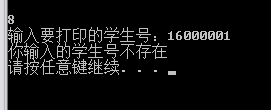
7实现的函数

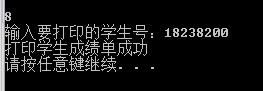
Txt\_phone();



8实现的函数

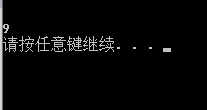
Txt\_sno();





9实现的函数

Txt\_groupby();



10 11 12 功能是为了方便查看数据增删查改之后的情况为将数据形成文档。

四**、结果分析**

对研究过程中所获得的主要的数据、现象进行定性或定量分析，得出结论和推论。

从上面的运行结果来看，上述的功能都实现了，运用了归并排序和哈希链地址散列法，学生成绩20万规模，50万规模，100万规模以及200万成绩的能很快得到满足要求的增删查改要求。

最后一次测试的是有10万学生，1万个课程，200w成绩的测试结果。

在插入学生中，我没有对数据进行排序，所以插入的时候，我直接在表头的下一个结点查入，不用去找插入的排序位置，而我在对课程还有成绩都有运用归并排序去处理数据，这样我在判断的时候，则不用遍历全部的数据，也是一种相对优化的方法，不过就是需要引进归并排序函数处理一下顺序；

与我一开始没有应用哈希散列相比，优化后运行速度非常快。在学生成绩管理系统中，用哈希散处理数据非常快，非常实用。

五**、**课程设计总结

总结可以包括:课程设计过程的收获、遇到的问题，遇到问题解决问题过程的思考、程序调试能力的思考，课程设计实现过程中的收获和体会等。

开始动手的时候，很多知识都忘了，并且不知道如何下手，就重新将课本看了一遍之后再开始写，那种重新掌握知识的感觉真的不错；之前掌握不牢，所以才会有“提笔忘字”，双手放在电脑上不知道从何开始的感觉，复习完之后，写起来也非常流畅和高效；在写cnoIndex结构的时候，出现了很多bug，要么是删除不彻底，要么是添加的位置不得当，所以debug的时间是最长的；后面解决了问题之后，又想起了将判断的过程放在一个函数中，也就可以不用总是写判断函数，优化之后，代码看起来整洁舒服；在逐一实现功能的时候，不能把所有的功能一起测试，而是应该先实现基本的功能，再把各个功能合在一起实现，调控。

由于我一开始的目的是想让自己主动去掌握知识，所以选择一个人做，效果也是如此，逼自己去学习，我掌控了很多知识，对知识应用更好，但是就造成了所有工作都是我自己完成，所以，后面有类似大作业，还是得和同学合作完成，当自己遇到问题，可以一起讨论，并且掌握，这也对我自己的能力提升有所帮助。

这次掌握最好的关于链表的使用，关于空链表，以及头结点不变的问题，都是非常注意的，这就是链表最重要的点。

还有我在处理读取TXT文档时，也是掌握好多以前没有注意到的点。

这次我收获非常多，最大的变化是我积极去解决问题，以及我学习态度的转变。

参考文献

1. 陈宝平.数据结构（c++版）[M].北京；清华大学出版社，2012.1
2. Clifford A.Shaffer.数据结构与算法分析（c++版）（第三版）[M].

北京：电子工业出版社，2013.1

1. Alex\_McAvoy数据结构与算法 —— 线性表 —— 顺序表https://blog.csdn.net/u011815404/article/details/88363099