

**课 程 设 计**

|  |  |
| --- | --- |
| **课程设计名称：** | **数据结构课程设计** |
| **专 业 班 级 ：** |  |
| **学 生 姓 名 ：** |  |
| **学 号 ：** |  |
| **指 导 教 师 ：** |  |
| **课程设计时间：** | **2022.12.5-2022.12.16** |

**计算机科学与技术 专业课程设计任务书**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **学生姓名** |  | **专业班级** | |  | **学号** | |  |
| **题 目** | 宿舍卫生管理系统 | | | | | | |
| **课题性质** | 工程设计 | | **课题来源** | | | 自拟 | |
| **指导教师** |  | | **同组姓名** | | | 无 | |
| **主要内容** | 开发一个简单的宿舍卫生管理系统。要求实现以下基本功能：有简单的用户管理功能，不同用户在登录系统后会有不同的菜单可进行不同的操作。  管理员可以实现以下功能：  （1）宿舍信息录入功能，主要包括录入宿舍号，宿舍各项卫生成绩，宿舍成员(宿舍信息用文件保存)。  （2）宿舍信息修改功能（修改后的信息也要用文件保存）。  （3）宿舍信息浏览功能。  （4）宿舍信息删除功能  （5）宿舍成绩排序输出  （6）更改管理员密码  学生：  （1）按成绩输出所有宿舍  （2）按宿舍号输出所有宿舍  （3）按宿舍号查询自己宿舍  （4）查询本周文明宿舍  （5）查询本周差评宿舍 | | | | | | |
| **任务要求** | 一、提交材料应包括：（1）系统源代码 （2）课程报告  二、整个设计过程具体要求  （1）**需求分析** 要求学生对案例系统进行分析，设计出需要完成的功能，完善各个模块的调用关系；  （2）**设计过程** 要求学生进一步明确各模块调用关系，进一步完善模块函数细节（函数名、参数、返回值等）；  （3）**实现过程** 要求学生养成良好的编码习惯、完成各个模块并进行测试，最终完成系统整体测试；  （4）**总结阶段** 按照要求完成系统设计和实现报告，并进行总结、答辩。 | | | | | | |
| **参考文献** | 1. Paul Deitel, Harvey Deitel著, 张引等译. C++大学教程（第九版）[M].北京: 电子工业出版社, 2016. 2. 郭炜. 新标准C++程序设计 [M]. 北京: 高等教育出版社, 2016. 3. 谭浩强. C++面向对象程序设计（第2版）[M]. 北京: 清华大学出版社, 2014. 4. 严蔚敏. 数据结构（C语言版）[M]. 北京: 清华大学出版社, 2011.   [5] Stephen Prata著, 张海龙等译. C++ Primer Plus中文版(第6版)[M]. 北京: 人民邮电出版社, 2012. | | | | | | |
| **审查意见** | **指导教师签字：**  **教研室主任签字： 2022年 6月 13日** | | | | | | |

**目录**

[**1 需求分析** 1](#_Toc122272303)

[**1.1 系统概述** 1](#_Toc122272304)

[**1.2 系统中的角色** 1](#_Toc122272305)

[**1.3 功能性需求** 1](#_Toc122272306)

[**1.4 用例图** 1](#_Toc122272307)

[**2 概要设计** 2](#_Toc122272308)

[**2.1 抽象数据类型** 2](#_Toc122272309)

[**2.2 存储结构设计** 3](#_Toc122272310)

[**2.3 功能结构图** 3](#_Toc122272311)

[**2.4 系统子程序及功能设计** 4](#_Toc122272313)

[**3 运行环境** 5](#_Toc122272314)

[**4 开发工具和编程语言** 5](#_Toc122272315)

[**5 详细设计** 5](#_Toc122272316)

[**5.1 insert\_AVL\_number函数** 5](#_Toc122272317)

[**5.2** **insert\_AVL\_grades函数** 6](#_Toc122272318)

[**5.3** **right\_rotate函数** 7](#_Toc122272319)

[**5.4** **left\_rotate函数** 7](#_Toc122272320)

[**5.5** **erase函数** 8](#_Toc122272321)

[**5.6** **InorderTraversal函数** 9](#_Toc122272322)

[**5.7** **findMin函数** 9](#_Toc122272323)

[**5.8** **findMax函数** 10](#_Toc122272324)

[**5.9** **find函数** 11](#_Toc122272325)

[**6 运行实现** 11](#_Toc122272326)

[**7 心得体会** 18](#_Toc122272327)

[**8 参考文献** 19](#_Toc122272328)

**1 需求分析**

* 1. **系统概述**

本系统实现对宿舍卫生的管理。

* 1. **系统中的角色**

|  |  |  |
| --- | --- | --- |
| 角色 | 类别 | 描述 |
| 学生 | 服务对象 | 查询个人宿舍信息，宿舍信息排序，本周最佳（差）宿舍 |
| 管理员 | 管理人员 | 增加、删除、修改、查询宿舍卫生信息 |

* 1. **功能性需求**

1. 管理员：可在宿舍卫生管理系统上进行增加、删除、查询和修改的操作
2. 乘客：可在宿舍卫生管理系统上进行查询个人宿舍信息，宿舍信息排序，本周最佳（差）宿舍
   1. **用例图**

如图1：

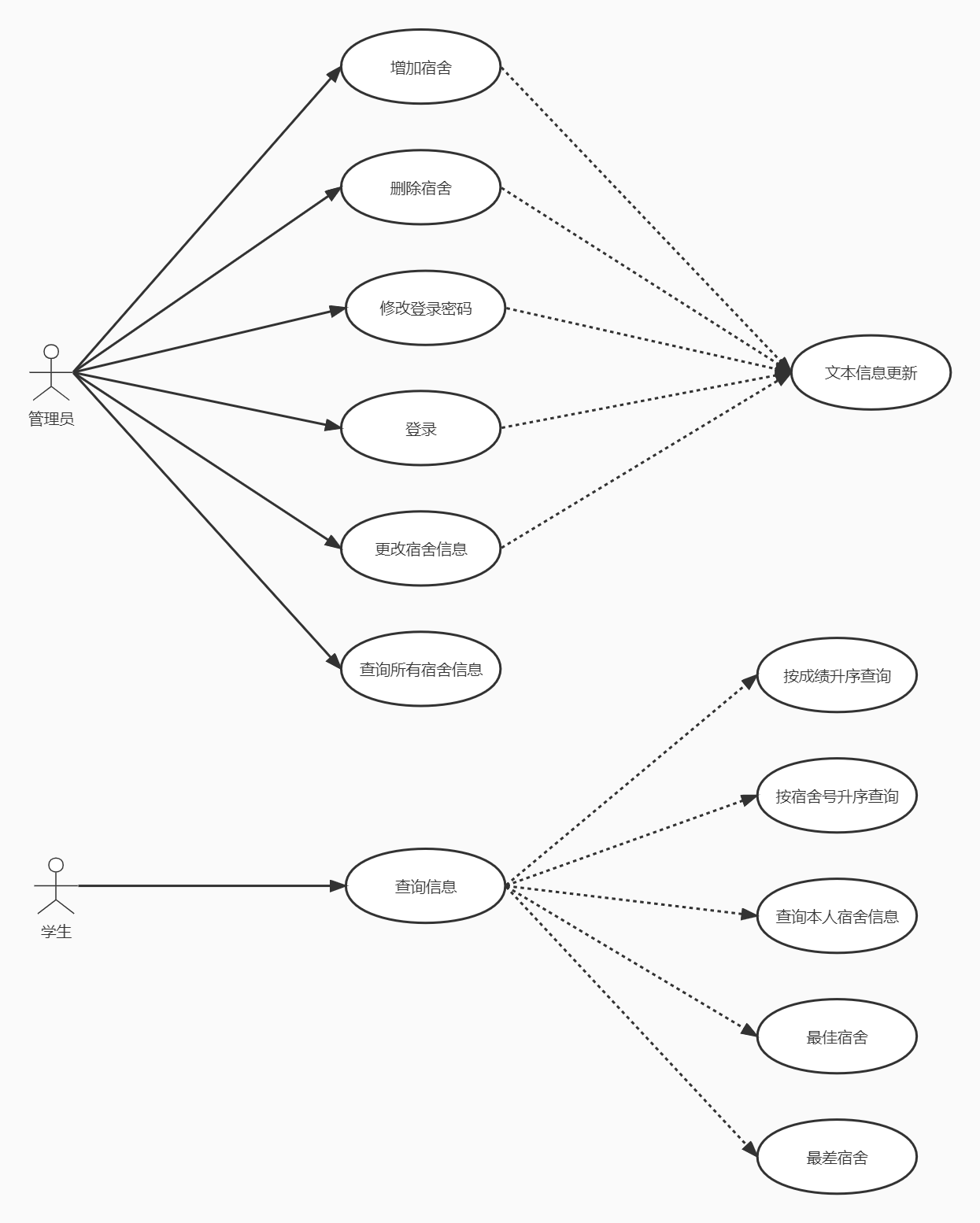


图1-1 用例图

**2 概要设计**

**2.1 抽象数据类型**

AVL：存储AVL树的基本操作

AVLnode：AVL树节点

People：基类

Dormitory：宿舍类，存储宿舍信息

Admin：管理员类，存储管理员的基本功能

Stu：学生类，存储学生的基本功能

基本操作：

AVLnode\* AVL::left\_rotate(AVLnode\* root)：AVL树左旋调整

AVLnode\* AVL::right\_rotate(AVLnode\* root)：AVL树右旋调整

AVLnode\* AVL::erase(AVLnode\* root, string num)：AVL树删除节点

AVLnode\* AVL::insert\_AVL\_number(AVLnode\* root, Dormitory& dorm)：

以宿舍号为key将宿舍插入AVL树

AVLnode\* AVL::insert\_AVL\_grades(AVLnode\* root, Dormitory& dorm)：以成绩为key将宿舍插入AVL树

void AVL::InorderTraversal(AVLnode\* root)：中序遍历AVL树

AVLnode\* AVL::findMin(AVLnode\* root)：返回AVL树中最小值

AVLnode\* AVL::findMax(AVLnode\* root)：返回AVL树中最大值

**2.2 存储结构设计**

本系统中采用AVL树存储宿舍信息。

typedef struct AVLnode {

Dormitory key;

int h;

AVLnode\* lchild, \* rchild;

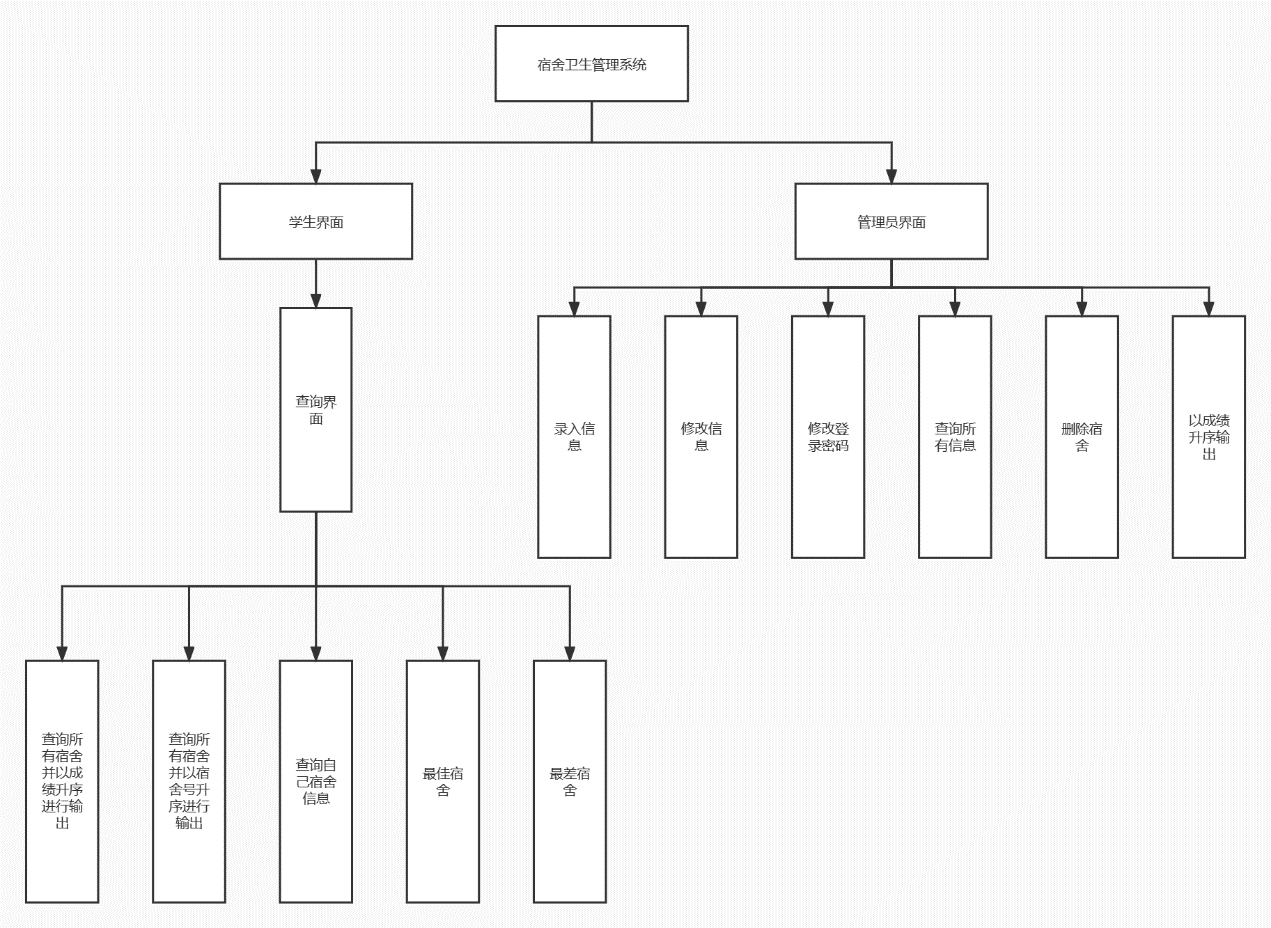
static AVLnode \_\_NIL;

AVLnode(Dormitory dorm = Dormitory(), int h = 0, AVLnode\* lchild = NIL, AVLnode\* rchild = NIL) : key(dorm), h(h), lchild(lchild), rchild(rchild) {}

};

**2.3 功能结构图**

功能结构图如图2-1所示：

图2-1 系统结构图

**2.4 系统子程序及功能设计**

本系统的管理员与学生模块共有16个函数

**2.4.1 学生相关函数**

|  |  |
| --- | --- |
| 函数名称 | 功能简述 |
| display\_grades | 按成绩排序输出 |
| display\_number | 按宿舍号排序输出 |
| query | 查询宿舍信息 |
| best | 文明宿舍 |
| worst | 最差宿舍 |

表2-1 学生相关函数

**2.4.2 管理员相关函数**

表2-2 管理员相关函数

|  |  |
| --- | --- |
| 函数名称 | 功能简述 |
| addDorm | 添加信息 |
| changeDorm | 更改信息 |
| changepassword | 修改登录密码 |
| deleteDorm | 删除信息 |
| display | 浏览全部信息 |
| getpassword | 获取密码 |
| load | 登录 |
| passwordfilesave | 保存密码 |
| petfilesave | 保存信息 |
| sortDorm | 按成绩升序输出 |
| inputpassword | 输入密码 |

**2.4.3 函数的主要关系（可选）**

函数的调用关系图，如图2-2所示：

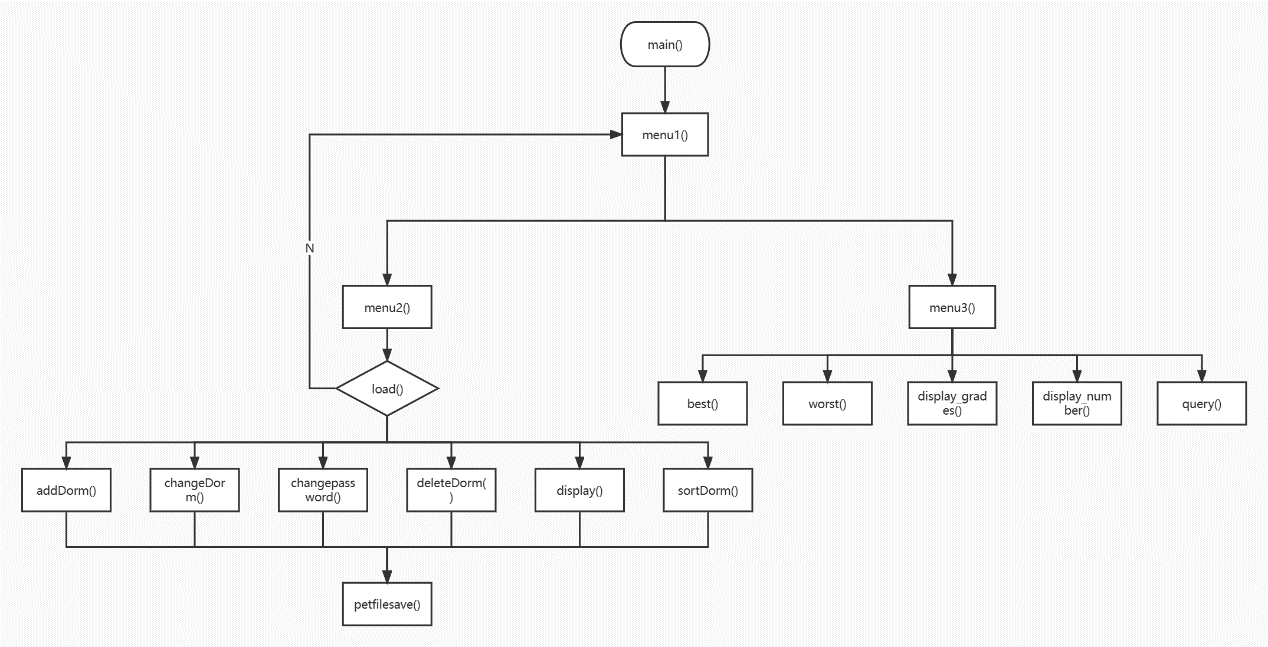


图2-2 函数调用关系图

**3 运行环境**

硬件环境： PC机内存：16G

软件环境： 操作系统：windows10

**4 开发工具和编程语言**

开发环境：visual studio 2022

编程语言：C++语言

**5 详细设计**

**5.1 insert\_AVL\_number函数**

1. 设计思想

根据文件中保存的数据，以宿舍号number作为AVL树的key进行AVl树的插入操作。

1. 涉及的数据结构

本函数主要涉及插入数据即宿舍信息。

1. 关键代码

AVLnode\* AVL::insert\_AVL\_number(AVLnode\* root, Dormitory& dorm) {

if (root == NIL)return getnewnode(dorm);

if (root->key.getNumber() == dorm.getNumber())return root;

if (root->key.getNumber() > dorm.getNumber()) {

root->lchild = insert\_AVL\_number(root->lchild, dorm);

}

else {

root->rchild = insert\_AVL\_number(root->rchild, dorm);

}

update\_height(root);//更新树高

return maintain(root);//返回调整过之后的节点

}

* 1. **insert\_AVL\_grades函数**

1. 设计思想

根据文件中保存的数据，以宿舍成绩grades作为AVL树的key进行AVl树的插入操作。

1. 涉及的数据结构

本函数主要涉及插入数据即宿舍信息。

1. 关键代码

AVLnode\* AVL::insert\_AVL\_grades(AVLnode\* root, Dormitory& dorm) {

if (root == NIL)return getnewnode(dorm);

if (root->key.getSum\_grades() == dorm.getSum\_grades())return root;

if (root->key.getSum\_grades() > dorm.getSum\_grades()) {

root->lchild = insert\_AVL\_grades(root->lchild, dorm);

}

else {

root->rchild = insert\_AVL\_grades(root->rchild, dorm);

}

update\_height(root); //更新树高

return maintain(root); //返回调整过之后的节点

}

* 1. **right\_rotate函数**

1. 设计思想

AVL树的右旋操作

1. 涉及的数据结构

AVL树的节点

1. 关键代码

AVLnode\* AVL::right\_rotate(AVLnode\* root) {

AVLnode\* new\_root = root->lchild;

root->lchild = new\_root->rchild;

new\_root->rchild = root;

update\_height(root);

update\_height(new\_root);//先更新老节点再更新新节点

return new\_root;

}

* 1. **left\_rotate函数**

1. 设计思想

AVL树的左旋操作

1. 涉及的数据结构

AVL树的节点

1. 关键代码

AVLnode\* AVL::left\_rotate(AVLnode\* root) {

AVLnode\* new\_root = root->rchild;

root->rchild = new\_root->lchild;

new\_root->lchild = root;

update\_height(root);

update\_height(new\_root);

return new\_root;

}

* 1. **erase函数**

1. 设计思想

AVL树的节点删除

1. 涉及的数据结构

AVL树的节点，宿舍号

1. 关键代码

AVLnode\* AVL::erase(AVLnode\* root, string num) {

if (root == NIL)return root;

if (num < root->key.getNumber()) {

root->lchild = erase(root->lchild, num);

}

else if (num>root->key.getNumber()) {

root->rchild = erase(root->rchild, num);

}

else {

if (root->lchild == NIL || root->rchild == NIL) {//度为1或0的节点删除

AVLnode\* temp = (root->lchild == NIL ? root->rchild : root->lchild);

delete root;

return temp;

}

else {//度为2的节点删除

AVLnode\* temp = predeccessor(root);

root->key = temp->key;

root->lchild = erase(root->lchild, temp->key.getNumber());

}

}

update\_height(root);

return root;

}

* 1. **InorderTraversal函数**

1. 设计思想

递归实现AVL树的中序遍历

1. 涉及的数据结构

AVL树的节点，宿舍号

1. 关键代码

void AVL::InorderTraversal(AVLnode\* root) {

if (root != NIL) {

InorderTraversal(root->lchild);

cout << left << setw(8) << root->key.getFloor\_grades() << " ";

cout << left << setw(8) << root->key.getBed\_grades() << " ";

cout << left << setw(8) << root->key.getItems\_grades() << " ";

cout << left << setw(4) << root->key.getOther\_grades() << " ";

cout << left << setw(4) << root->key.getSum\_grades() << " ";

cout << left << setw(6) << root->key.getNumber() << " ";

cout << left << setw(8) << root->key.getMember()[0] << " ";

cout << left << setw(8) << root->key.getMember()[1] << " ";

cout << left << setw(8) << root->key.getMember()[2] << " ";

cout << left << setw(8) << root->key.getMember()[3] << endl;

InorderTraversal(root->rchild);

}

}

* 1. **findMin函数**

1. 设计思想

递归查找AVL树中存储最小值

1. 涉及的数据结构

AVL树的节点，宿舍成绩

1. 关键代码

AVLnode\* AVL::findMin(AVLnode\* root) {

if (root == NIL) {

return NULL;

}

if (root->lchild == NIL) {

return root;

}

return findMin(root->lchild);

}

* 1. **findMax函数**

1. 设计思想

递归查找AVL树中存储最大值

1. 涉及的数据结构

AVL树的节点，宿舍成绩

1. 关键代码

AVLnode\* AVL::findMax(AVLnode\* root) {

if (root == NIL) {

return NULL;

}

if (root->rchild==NIL) {

return root;

}

return findMax(root->rchild);

}

* 1. **find函数**

1. 设计思想

查找AVL树中存储的目标值

1. 涉及的数据结构

AVL树的节点，宿舍号

1. 关键代码

AVLnode\* AVL::find(AVLnode\* root, string Number) {

if (root == NIL) {

return NIL;

}

if (Number < root->key.getNumber())

return find(root->lchild, Number);

else if (Number > root->key.getNumber())

return find(root->rchild, Number);

else

return root;

}

**6 运行实现**

系统运行主界面，如图6-1所示：

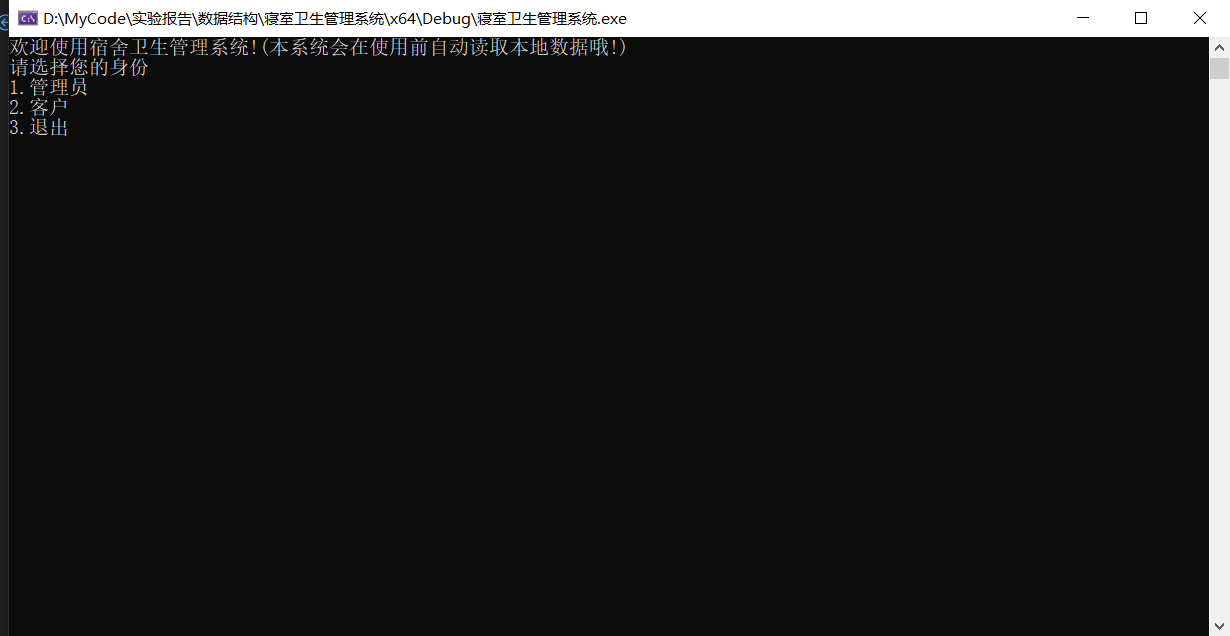


图6-1 宿舍卫生管理系统主菜单

管理员登录界面如图6-2

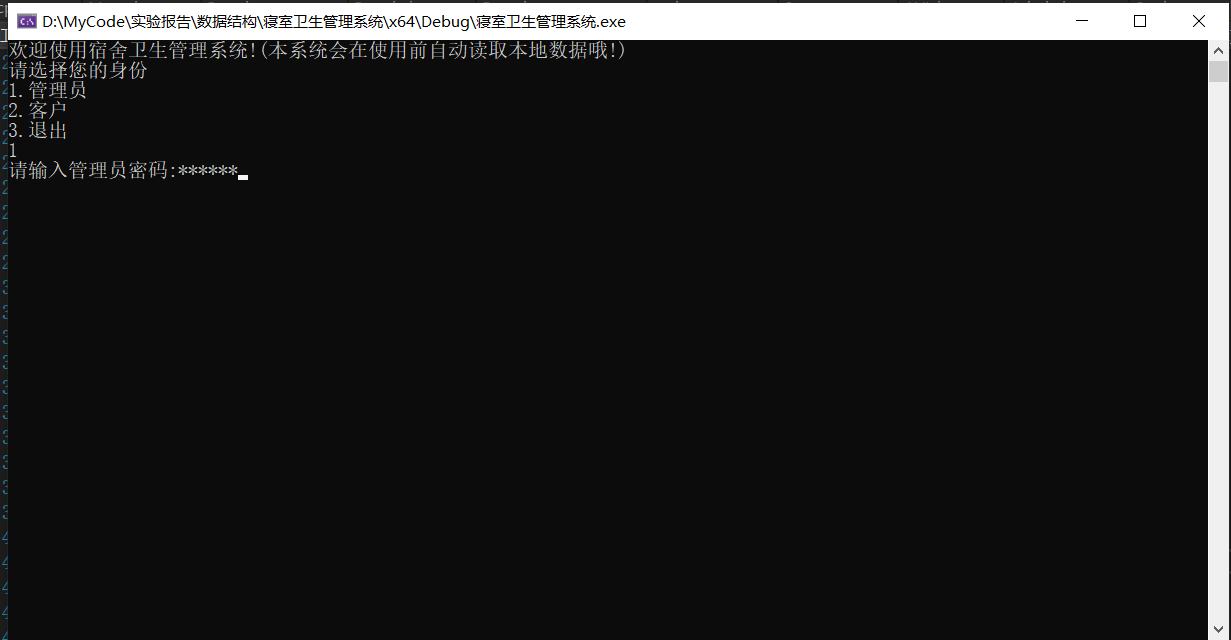


图6-2 管理员登录界面

录入信息界面如图6-3

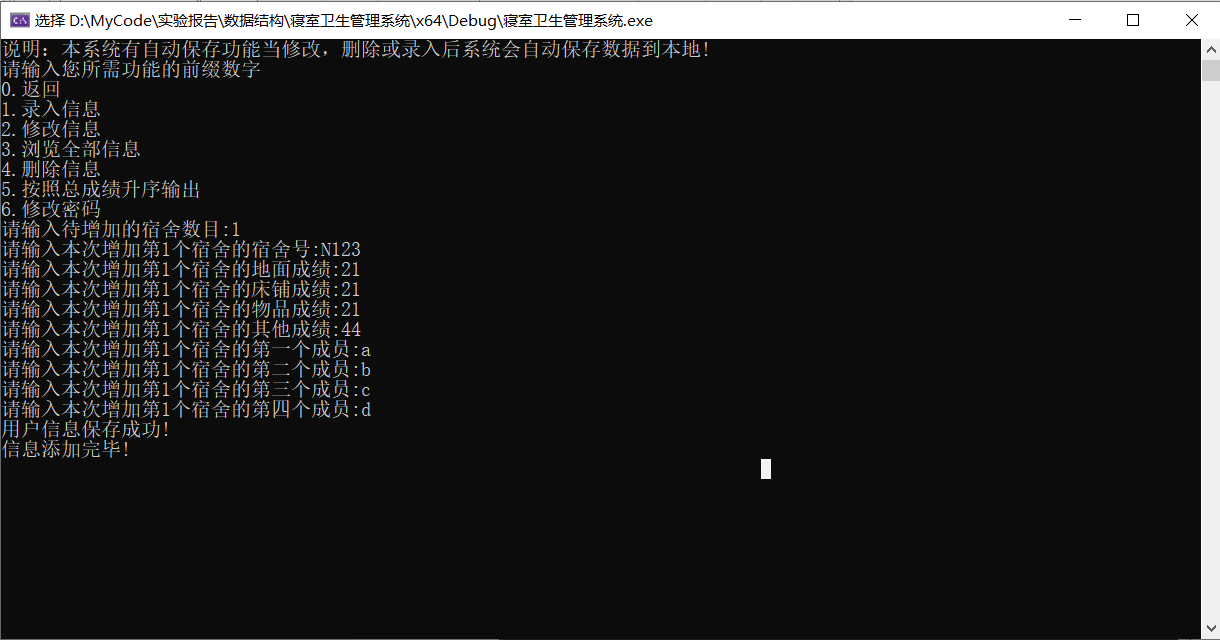


图6-3 管理员录入信息界面

修改信息界面如图6-4



图6-4 管理员修改信息界面

浏览全部信息界面如图6-5

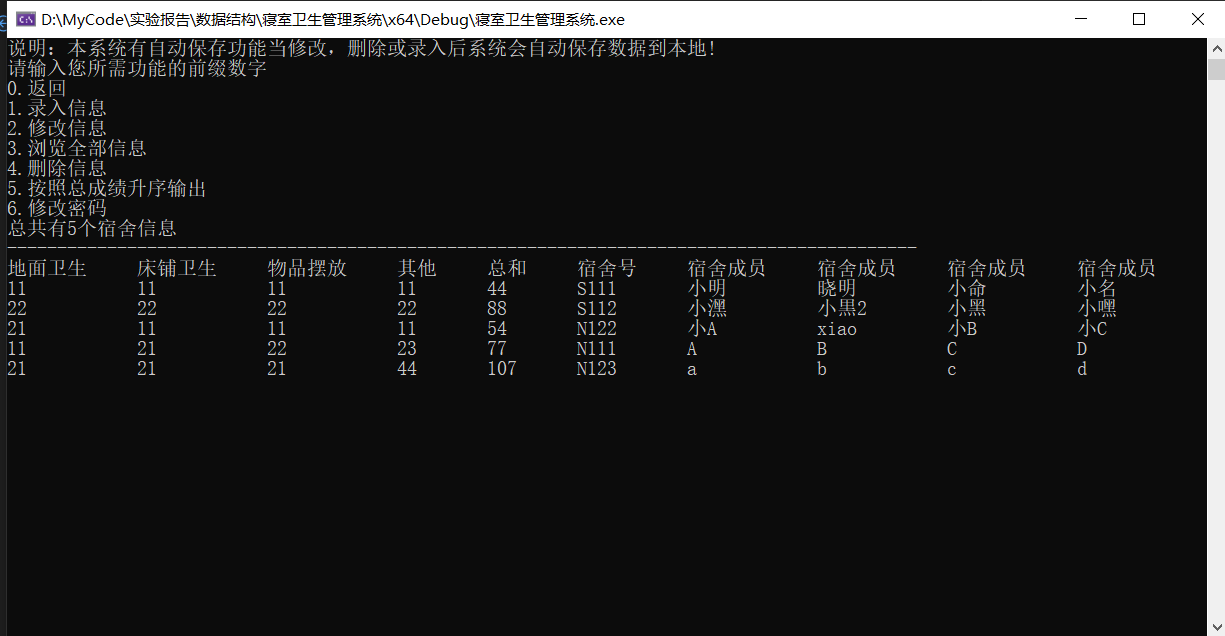


图6-5 管理员浏览全部信息界面

删除信息界面如图6-6

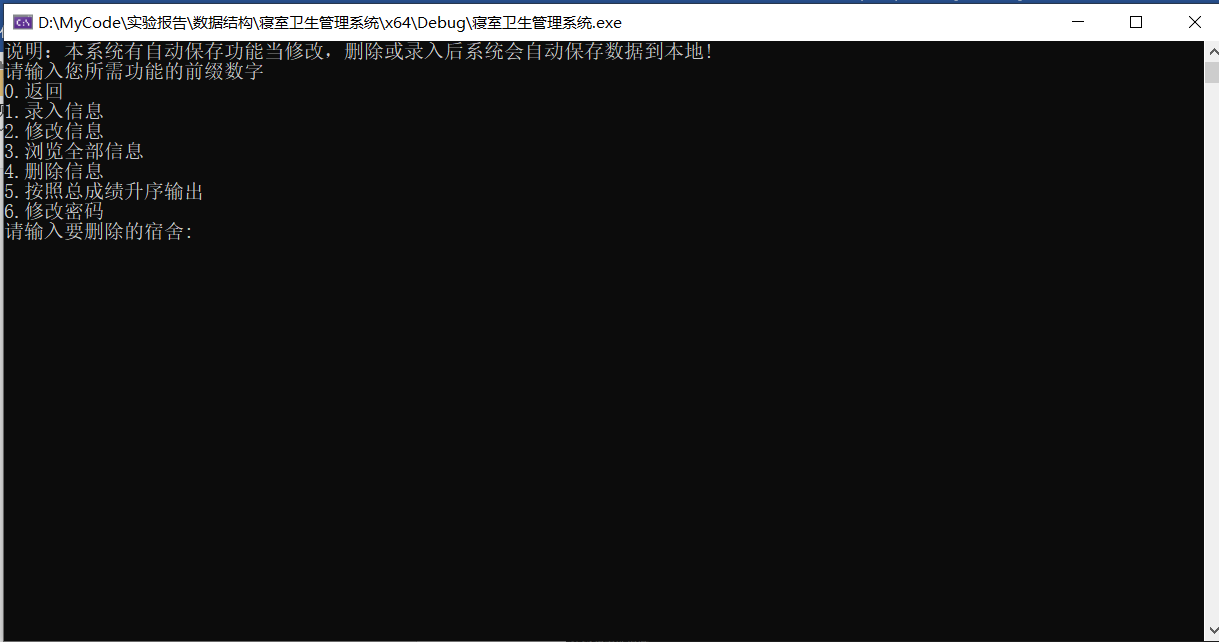


图6-6 管理员删除信息界面

按成绩升序输出如图6-7



图6-7 管理员按成绩升序输出信息界面

修改密码界面如图6-8

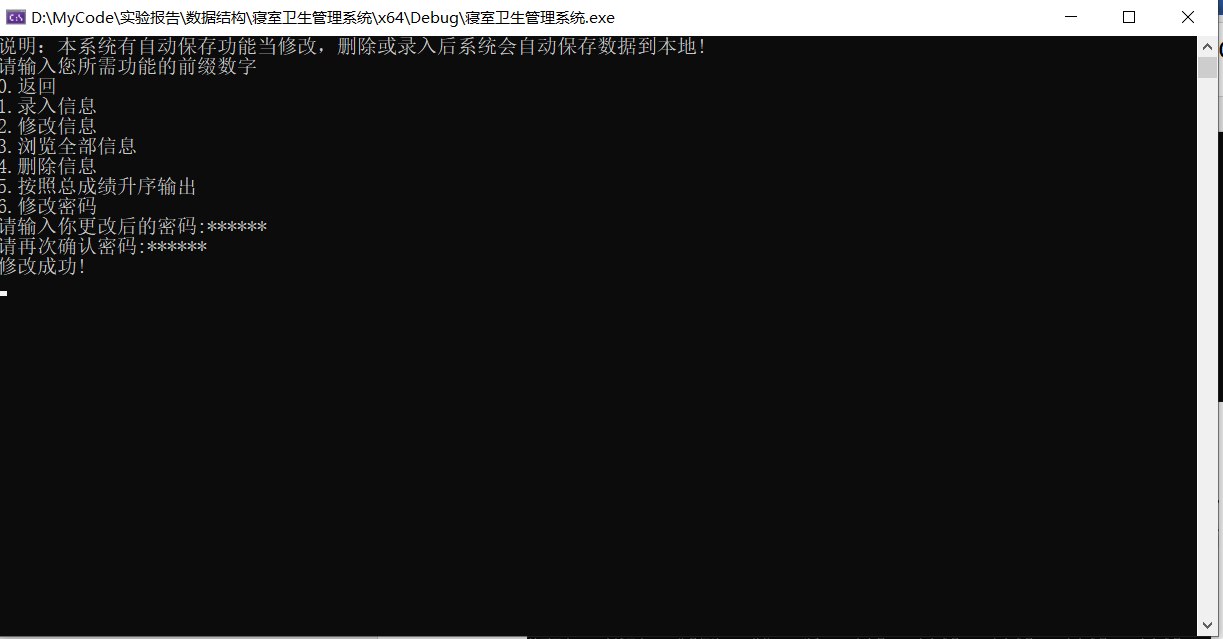


图6-8 管理员修改密码界面

用户界面如图6-9

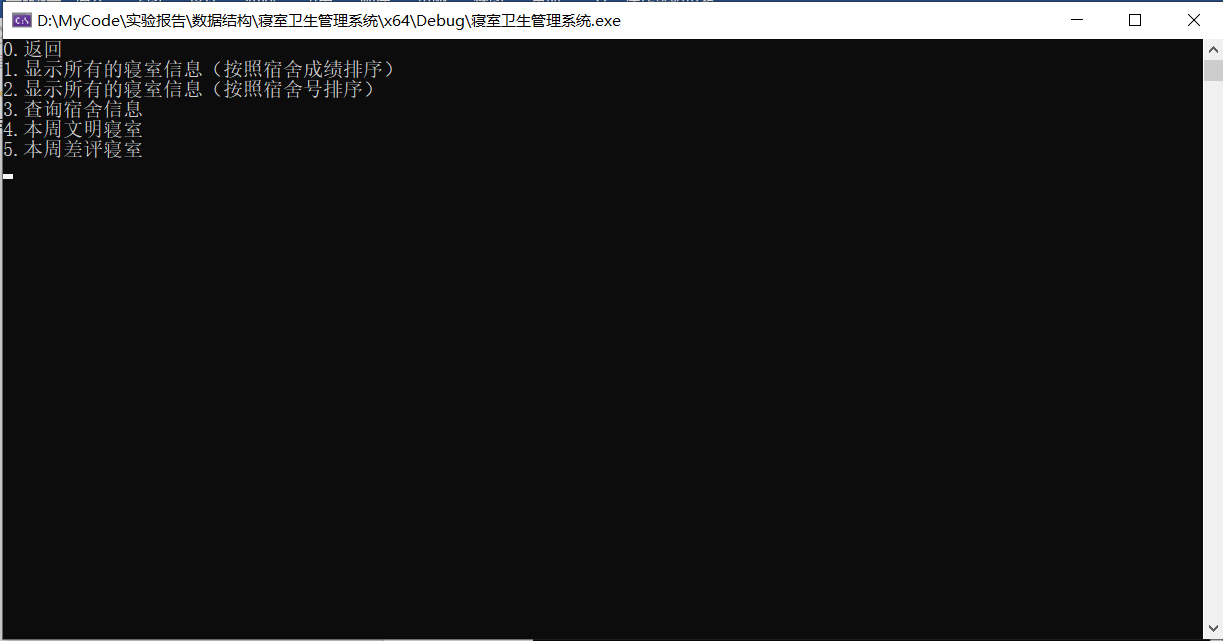


图6-9 用户界面

按宿舍成绩升序输出所有宿舍信息如图6-10

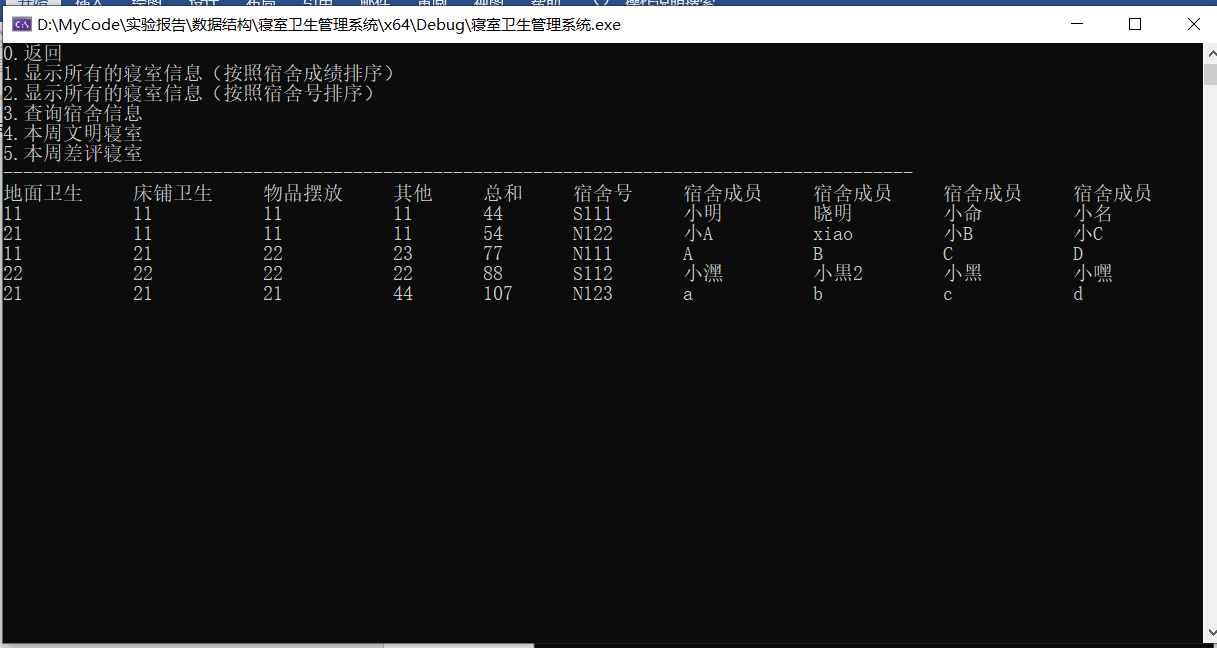


图6-10 按宿舍成绩升序输出所有宿舍信息

按宿舍号升序输出所有宿舍信息如图6-11

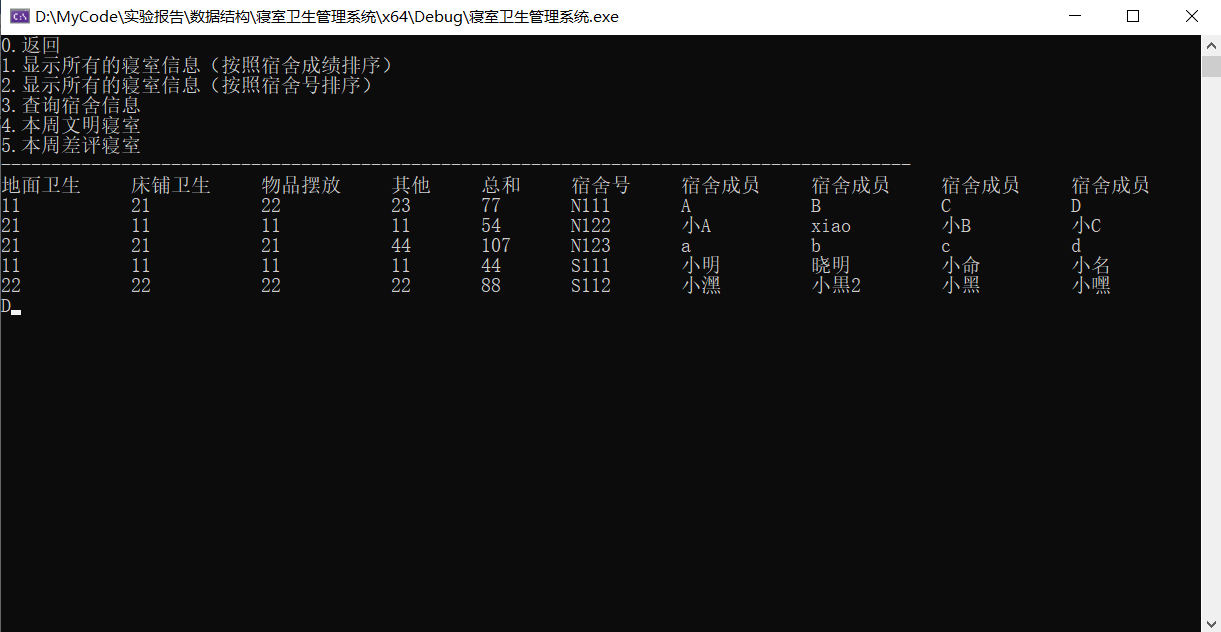


图6-11 按宿舍号升序输出所有宿舍信息

查询宿舍信息如图6-12



图6-12 按宿舍号查询宿舍信息

查询本周文明宿舍如图6-13

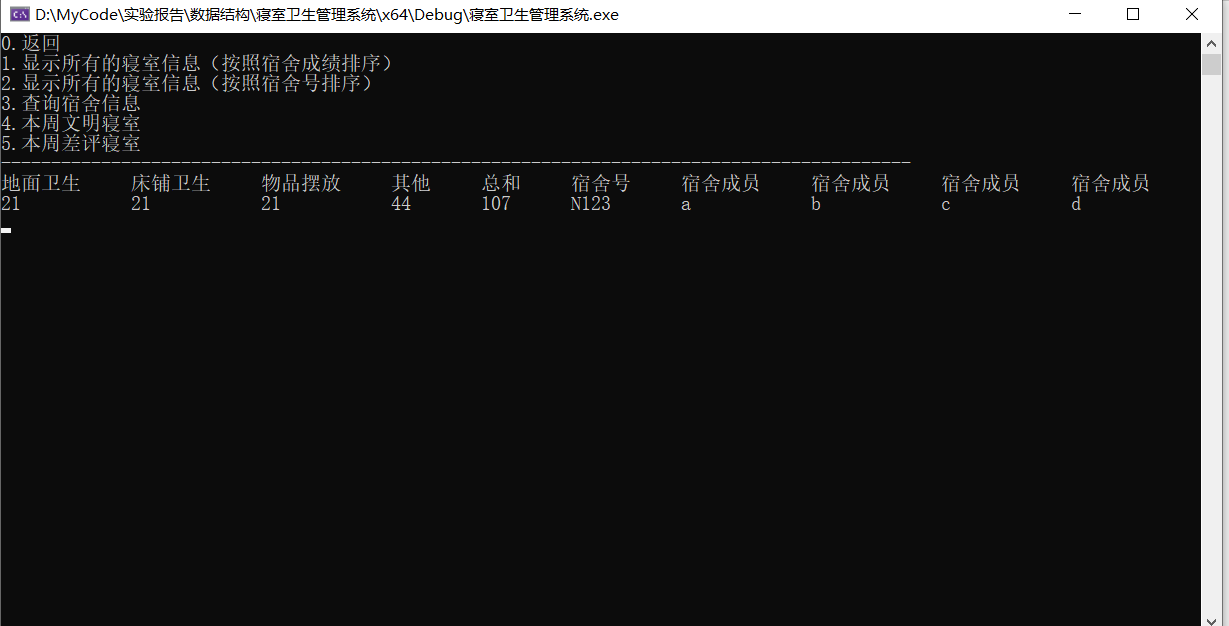


图6-13查询本周文明宿舍

查询本周差评宿舍如图6-14



图6-14查询本周差评宿舍

**7 心得体会**

通过此次以公交线路查询系统为题的课程设计，我更加扎实的掌握了有关数据结构方面的知识。虽然在设计的过程中，遇到了很多的问题，但经过不断地尝试探索以及搜索资料，找到了问题所在，并解决了困难。在这个过程中，学习到了很多新的知识，增强了实践能力，丰富了经验。

**8 参考文献**

[1]Paul Deitel, Harvey Deitel著, 张引等译. C++大学教程（第九版）[M].北京: 电子工业出版社, 2016.

[2]郭炜. 新标准C++程序设计 [M]. 北京: 高等教育出版社, 2016.

[3]谭浩强. C++面向对象程序设计（第2版）[M]. 北京: 清华大学出版社, 2014.

[4]严蔚敏. 数据结构（C语言版）[M]. 北京: 清华大学出版社, 2011.

[5] Stephen Prata著, 张海龙等译. C++ Primer Plus中文版(第6版)[M]. 北京: 人民邮电出版社, 2012.