

C++程序设计课程设计实验报告

2018/2019(2)



实验题目 物业维修管理系统

学生姓名 胡皓睿

学生学号 201806061108

学生班级 计实1801

任课教师 田贤忠

提交日期 2019/6/12

计算机科学与技术学院

物业维修管理系统 实验报告

1. 实验内容

物业维修管理系统用物业对维修进行管理，要求完成的主要的功能包括增加和删除维修记录、修改维修记录，查询预约到期记录，根据楼号统计各幢楼的维修记录数量及总维修收费金额、总维修材料成本金额，所有维修记录浏览（以实际维修日期排序显示）。要求使用学习过的C/C++程序设计的知识完成物业维修管理系统的设计与实现。

基本要求：能从文件读出维修记录信息，并能将信息保存到文件

1. 运行环境

物业维修管理在Visual Studio 2019平台下开发，操作系统：Windows 10。

硬件环境：

处理器：Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80GHz

内存：8.00GB

系统类型：64位操作系统

1. 实验课题分析（主要的模块功能、流程图）
   1. 物业维修管理系统的主要功能
      1. 对记录有关的操作（增删改查）

插入新记录，删除记录，修改记录，查询记录

* + 1. 和数据有关的操作

对数据进行分析统计

将数据保存至磁盘

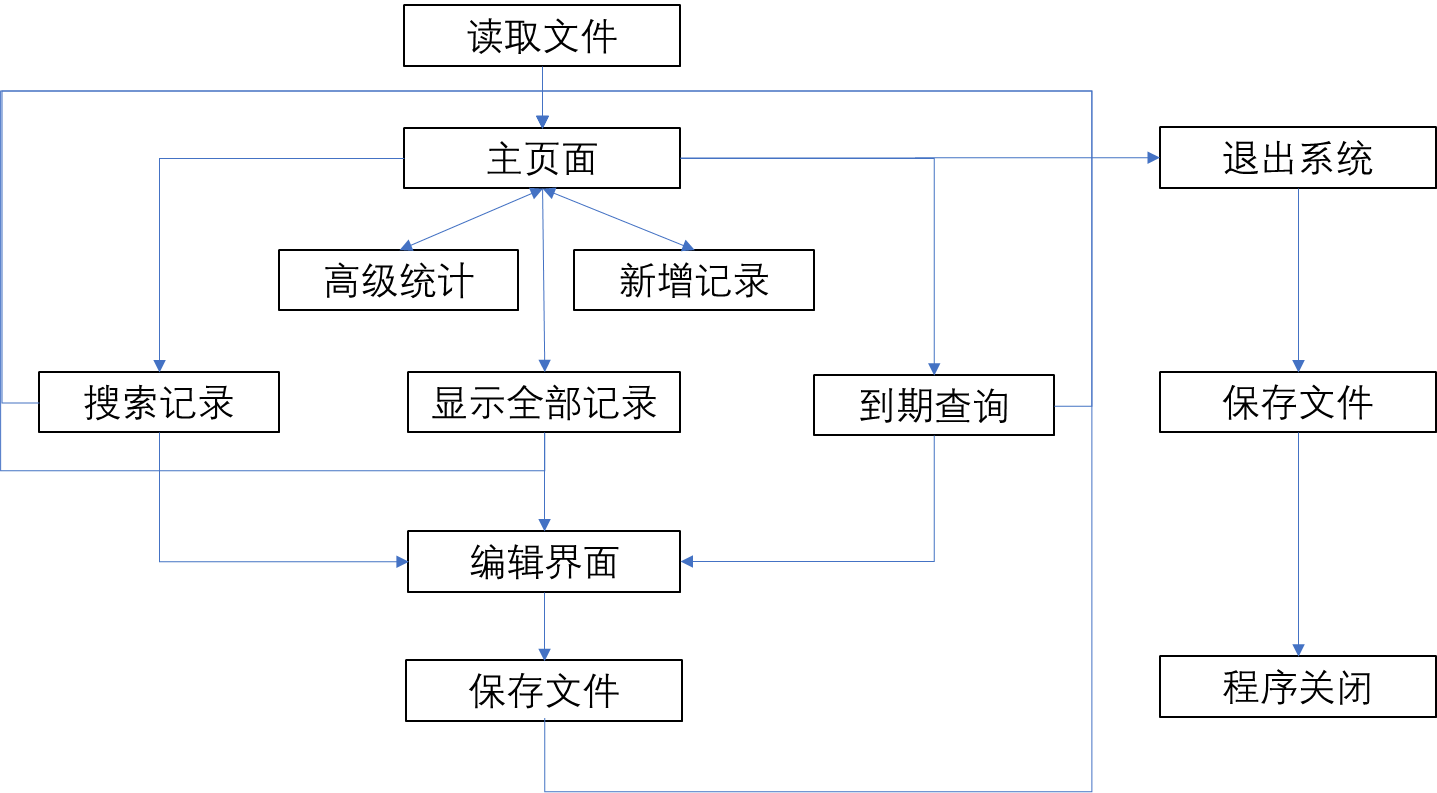
* + 1. 提高数据可用性的操作

对数据进行分类和排序

使用合适的界面显示出数据

* 1. 系统分析及设计

交互逻辑如该流程图所示



* 1. 系统的实现
     1. 模块化的思想

把每一个小功能都放在一个单独的函数里，需要使用某个特定的功能时，只需调用特定的小函数，避免了超过一百行的函数的出现，保证的程序的可读性。

把自己写的函数放到了function.h和function.cpp里，保证了在各个模块中都可以有效调用。

#pragma once

#include "date\_time.h"

void pause();

void cls();

date today();

bool is\_proper\_num(char c);

bool is\_proper\_num(string s);

int to\_int(string s);

double to\_double(string s);

date\_time now();

string toString(int n);

string toString(double n);

* + 1. 存储文件的设计

存储文件以行为单位，每一行保存一条记录，依次存放每组数据的每一个信息，并以空格分隔

16 107 2019 2 4 2020 12 13 空调 992.160156 158.565956 胡皓睿 无 0

44 503 2019 2 6 2020 8 4 空调 955.808119 133.982216 胡皓睿 无 1

123 809 2019 3 26 2019 9 21 空调 176.447249 248.292044 胡皓睿 无 1

12 102 2019 4 2 2020 2 4 空调 271.719538 344.647204 胡皓睿 无 0

40 400 2019 4 19 2020 11 19 空调 95.819722 159.902427 胡皓睿 无 0

28 807 2019 4 23 2019 9 17 空调 873.208928 323.437684 胡皓睿 无 1

21 903 2019 5 18 2020 1 28 空调 556.061035 203.160528 胡皓睿 无 1

1 234 2019 5 23 2019 5 22 晚安 0.000000 0.000000 胡皓睿 无 0

2 704 2019 7 17 2019 12 2 空调 279.226689 374.188971 胡皓睿 无 1

4 204 2019 8 8 2020 3 25 空调 248.653876 348.857203 胡皓睿 无 0

35 200 2019 8 12 2019 9 19 空调 622.889584 454.007933 胡皓睿 无 0

3 302 2019 8 28 2020 10 22 空调 537.636801 110.827359 胡皓睿 无 0

13 700 2019 9 27 2020 4 10 空调 525.725958 413.176538 胡皓睿 无 0

14 601 2019 10 16 2019 2 5 空调 749.860286 498.156597 胡皓睿 无 1

41 601 2019 11 4 2020 3 15 空调 30.832447 440.633998 胡皓睿 无 1

27 902 2019 11 16 2020 4 20 空调 700.389866 372.357866 胡皓睿 无 0

39 1003 2019 11 26 2019 3 14 空调 518.105820 227.329702 胡皓睿 无 0

37 201 2019 12 26 2019 8 11 空调 369.295358 341.926398 胡皓睿 无 1

42 203 2020 1 21 2019 5 10 空调 648.776988 371.896643 胡皓睿 无 1

38 703 2020 2 10 2019 9 5 空调 556.214153 318.930560 胡皓睿 无 0

33 905 2020 4 24 2020 9 22 空调 213.185504 3.608270 胡皓睿 无 1

41 408 2020 5 8 2019 4 15 空调 516.345216 142.161082 胡皓睿 无 1

37 304 2020 6 27 2019 8 5 空调 227.762987 170.013024 胡皓睿 无 0

7 601 2020 8 18 2019 3 22 空调 530.714175 472.107996 胡皓睿 无 1

4 509 2020 10 6 2019 12 23 空调 565.289868 49.338826 胡皓睿 无 0

7 401 2020 11 17 2020 6 14 空调 904.692682 493.364641 胡皓睿 无 1

* + 1. 类的编写
       1. date类

date类，即日期类，包含了时间（年，月，日）的成员，并重载了流提取，大于等运算符进行计算，并包含成员函数tomorrow，用于返回下一天的日期（以date形式返回）

#include <iostream>

using namespace std;

#pragma once

class date

{

protected:

int day, month, year;

public:

date(int y = 2019, int m = 5, int d = 22);

void show()

{

cout << year << '/' << month << '/' << day;

}

string out();

date tomorrow();

friend bool operator >(date& d1, date& d2);

friend bool operator <(date& d1, date& d2);

friend bool operator ==(date& d1, date& d2);

friend ostream& operator <<(ostream& output, date& d);

friend istream& operator >>(istream& input, date& d);

};

* + - 1. date\_time类

date\_time类派生自date类，添加了时分秒显示的功能，用于显示当前的时间

#pragma once

#include "date.h"

class date\_time :public date

{

private:

int hour;

int minute;

int second;

public:

date\_time(date d = date(2000, 1, 1), int h = 0, int m = 0, int s = 0);

void show();

~date\_time();

};

* + - 1. record类

record类是该程序的关键，包含了要保存的所有数据，以及修改数据，判断完成情况，显示结果等大量功能。

#pragma once

#include "date.h"

#include <string>

class record

{

protected:

int num\_building, num\_house; //楼号，房号

date date\_reserve, date\_fact; //预约时间，实际时间

string content; //维修内容

double money\_income, money\_outcome; //收费，成本

string people; //检修人

string remark; //备注

bool done; //是否已经完成

public:

string show\_in\_line();

friend bool operator <(record& r1, record& r2)

{

return r1.date\_reserve < r2.date\_reserve;

}

friend bool operator >(record& r1, record& r2)

{

return r1.date\_reserve > r2.date\_reserve;

}

void save\_to\_file();

void show();

void show(int);

void edit\_num(int, int);

void edit\_date\_reserve(date);

void edit\_date\_fact(date);

void edit\_content(string);

void edit\_money(double, double);

void edit\_people(string);

void edit\_remark(string);

void edit\_done();

bool unfinished();

bool overdate(date);

bool day\_fact\_is(date);

bool num\_building\_is(int n)

{

if (n == num\_building) return true; else return false;

}

bool name\_is(string s)

{

if (s == people) return true; else return false;

}

bool exist(string s);//判断是否存在子串

double get\_income()

{

return money\_income;

}

double get\_outcome()

{

return money\_outcome;

}

void set(int s\_num\_building, int s\_num\_house, date s\_date\_reserve, date s\_date\_fact, string s\_content, double s\_money\_income, double s\_money\_outcome, string s\_people, string s\_remark, bool s\_done = 0);

record();

record(int s\_num\_building, int s\_num\_house, date s\_date\_reserve, date s\_date\_fact, string s\_content, double s\_money\_income, double s\_money\_outcome, string s\_people, string s\_remark, bool s\_done = 0);

~record();

};

* + 1. 交互界面的实现

cout << "正在读取文件，请等待\n";

init();

cls();

srand((int)time(0));

now().show();

cout << '\n';

cout << "┌──────────────────┐\n";

cout << "│ 1.新增记录 │\n";

cout << "│ 2.搜索记录 │\n";

cout << "│ 3.查看所有记录 │\n";

cout << "│ 4.预约到期查询 │\n";

cout << "│ 5.高级统计功能 │\n";

cout << "│ 6.显示随机记录 │\n";

cout << "│ 7.退出系统 │\n";

cout << "└──────────────────┘\n";

cout << "请输入你要使用的功能： ";

cin >> c;

switch (c[0] - '0') {

case 1:new\_record(); break;

case 2:search\_record(); break;

case 3:find\_all\_record(); break;

case 4:intime\_service(); break;

case 5:statistic\_income(); break;

case 6:random\_record(); break;

case 7:save\_all(); cout << "感谢使用，再见"; return 0; break;

default:

cout << "输入错误，请重新输入\n";

pause();

}

cls();

* + 1. 对误操作的预防
       1. 误操作出现的原因

在要求输入int型数据时，用户可能会误输入string型数据，导致程序出错

* + - 1. 误操作的预防方法

在可能出现误操作的地方，使用string暂时存储要输入的变量，在完成判断后再将其转回int类型

cout << "请输入新的预约时间:（输入一个小于10的数n，表示n天后）";

cin >> day\_later;

while (!is\_proper\_num(day\_later))

{

cout << "输入有误，请重新输入";

cin.ignore(100, '\n');//清空cin流

cin >> day\_later;

}

if (to\_int(day\_later) < 10)

{

date date\_tmp = today();

for (int x = 0; x < to\_int(day\_later); x++)

{

date\_tmp = date\_tmp.tomorrow();

}

records[n].edit\_date\_reserve(date\_tmp);

}

else

{

int year = to\_int(day\_later);

string month, day;

cin >> month;

while (!is\_proper\_num(month))

{

cout << "月份输入有误，请重新输入:";

cin.ignore(100, '\n');//清空cin流

cin >> month;

}

cin >> day;

while (!is\_proper\_num(day))

{

cout << "日期输入有误，请重新输入:";

cin.ignore(100, '\n');//清空cin流

cin >> day;

}

records[n].edit\_date\_reserve(date(year, to\_int(month), to\_int(day)));

}

* + 1. 功能的优化
       1. 身份验证功能

要求用户输入密码进行身份验证，同时，保存在磁盘上的密码使用MD5进行加密，保证安全性。

验证阶段：

string str, password\_MD5ed, result;

ifstream infile;

infile.open("password.dat", ios::in);

infile >> password\_MD5ed;

infile.close();

do

{

cout << "请输入管理员密码:";

cin >> str;

MD5 md5(str);

result = md5.md5();

cout << "密码错误\n";

} while (result != password\_MD5ed);

cls();

（使用了开源的MD5库）

class MD5

{

public:

typedef unsigned int size\_type; // must be 32bit

MD5();

MD5(const std::string& text);

void update(const unsigned char\* buf, size\_type length);

void update(const char\* buf, size\_type length);

MD5& finalize();

std::string hexdigest() const;

std::string md5() const;

friend std::ostream& operator<<(std::ostream&, MD5 md5);

private:

void init();

typedef unsigned char uint1; // 8bit

typedef unsigned int uint4; // 32bit

enum { blocksize = 64 }; // VC6 won't eat a const static int here

void transform(const uint1 block[blocksize]);

static void decode(uint4 output[], const uint1 input[], size\_type len);

static void encode(uint1 output[], const uint4 input[], size\_type len);

bool finalized;

uint1 buffer[blocksize]; // bytes that didn't fit in last 64 byte chunk

uint4 count[2]; // 64bit counter for number of bits (lo, hi)

uint4 state[4]; // digest so far

uint1 digest[16]; // the result

// low level logic operations

static inline uint4 F(uint4 x, uint4 y, uint4 z);

static inline uint4 G(uint4 x, uint4 y, uint4 z);

static inline uint4 H(uint4 x, uint4 y, uint4 z);

static inline uint4 I(uint4 x, uint4 y, uint4 z);

static inline uint4 rotate\_left(uint4 x, int n);

static inline void FF(uint4& a, uint4 b, uint4 c, uint4 d, uint4 x, uint4 s, uint4 ac);

static inline void GG(uint4& a, uint4 b, uint4 c, uint4 d, uint4 x, uint4 s, uint4 ac);

static inline void HH(uint4& a, uint4 b, uint4 c, uint4 d, uint4 x, uint4 s, uint4 ac);

static inline void II(uint4& a, uint4 b, uint4 c, uint4 d, uint4 x, uint4 s, uint4 ac);

};

* + - 1. 时间显示功能

由原有的date类派生出一个新的date\_time类，用于获取，存放和显示时间。

date\_time now()

{

tm t;

time\_t now;

time(&now);

localtime\_s(&t, &now);

date d(t.tm\_year + 1900, t.tm\_mon + 1, t.tm\_mday);

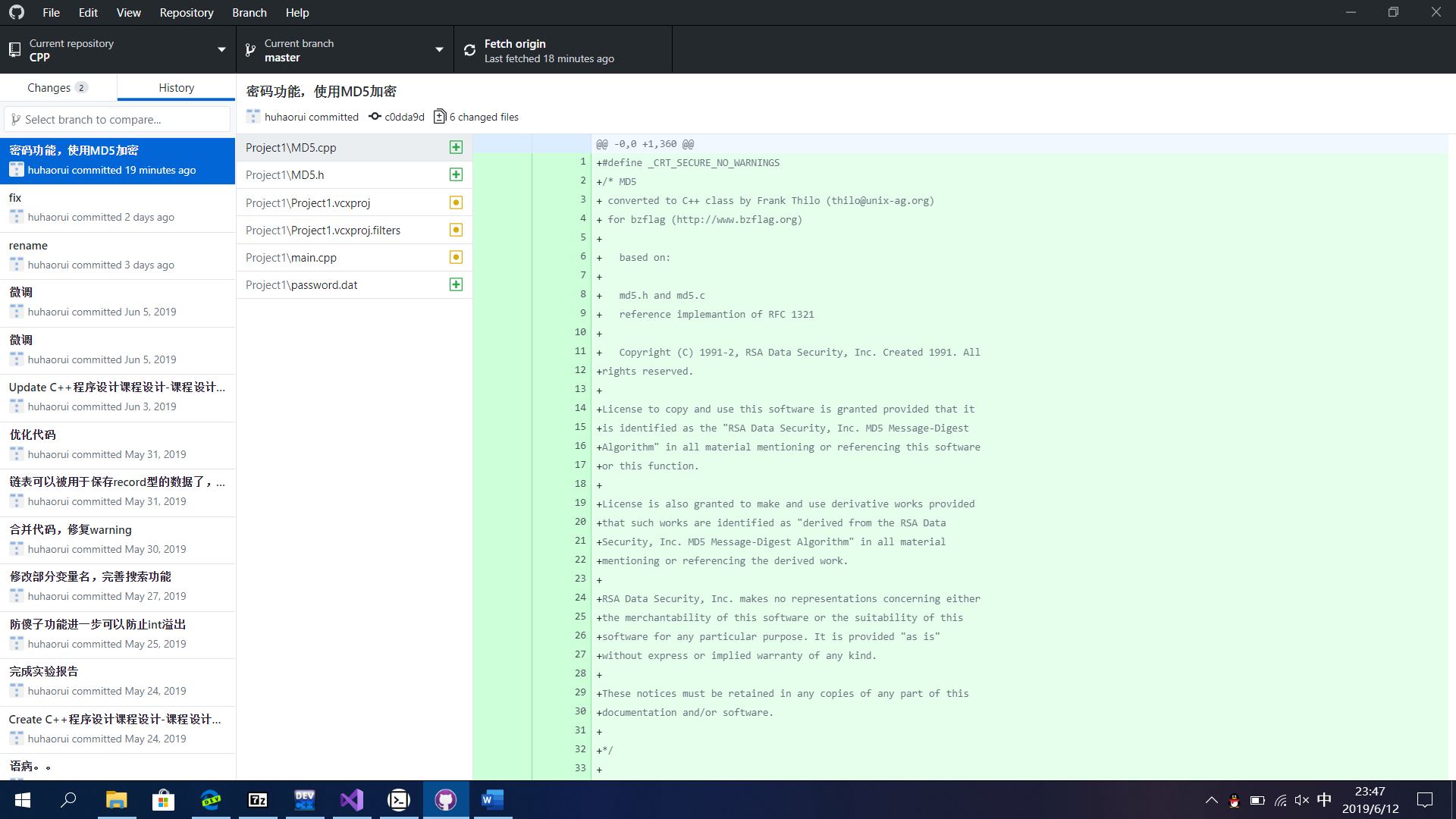
date\_time dt(d, t.tm\_hour, t.tm\_min, t.tm\_sec);

return dt;

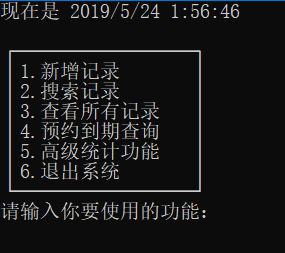
}

* + 1. 代码管理

使用 [GitHub](https://github.com/huhaorui/CPP) 进行代码同步与管理

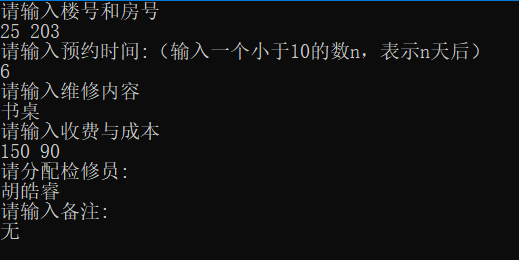


1. 实验调试、测试、运行记录及分析
   1. 使用说明
      1. 主界面

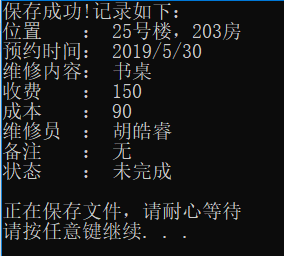


主界面包含了时间的显示，并有各个功能的入口，输入对应的数字进入对应的功能。

* + 1. 新增记录功能



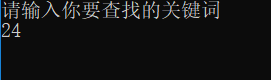
每当系统给出一个提示，输入对应的数据（如果输入数据有误，系统会检测并要求重新输入）

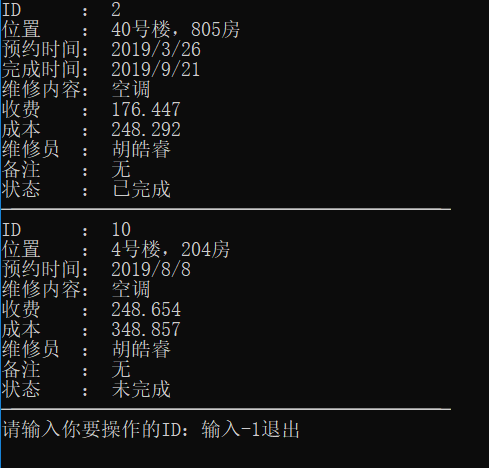


按下回车键，系统即会自动保存文件，然后回到主界面

* + 1. 搜索界面

在搜索界面中，你会被要求输入关键词，系统会自动对关键词进行匹配



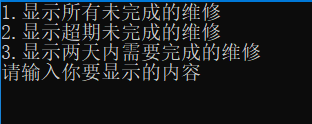


系统会要求你输入对应ID进行修改操作

* + 1. 所有记录页面

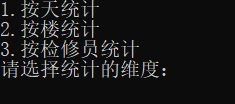
系统会显示出所有的记录，你可以输入ID进行修改操作

* + 1. 预约到期查询

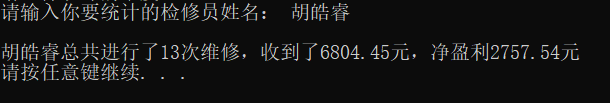


预约到期查询包含以下三个功能，均可以搜索出对应的记录，并根据ID进行操作

* + 1. 高级统计功能



可以以多维度统计数据，以“按检修员统计”为例



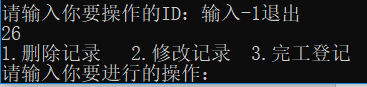
系统会自动计算出该检修员的有关信息

* + 1. 退出系统

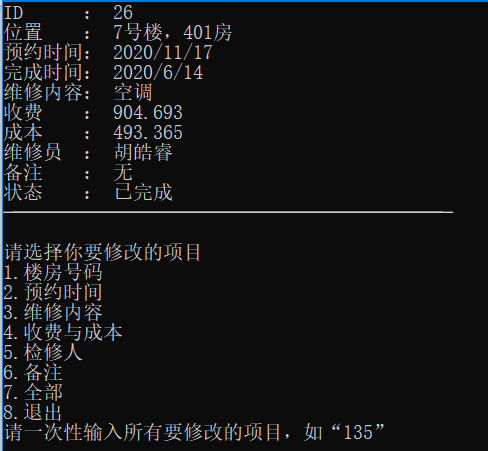
选择这个选项，即可保存数据，退出软件

* + 1. 记录操作界面

通过前方的多个入口，均可进入该页面



存在删除，修改，完工三个功能，介绍一下修改功能



系统会显示出这条记录，并提供可修改的选项，选择选项即可对记录进行修改。

1. 实验总结（优点、不足、收获及体会）

我设计的物业维修管理系统完成了实验的所有要求，并且对其做了许多自己的优化。

为了保证输入的错误数据的有效性，我对所有的输入数据都进行的合理性的校验，如果输入了不合理的数据，自动发出警告并要求重新输入

为了测试系统在巨大数量数据下的稳定性，使用了Excel软件辅助构建了一张超过万行的数据表，并且完美通过了考验

通过这次课程设计，我深刻的明白到：课本知识与实践能力相结合的重要性。会读程序的人并不一定会编程序。要想把一门专业课程学好，必须增强自己的动手实践能力，而不是一天到晚只知道看书，那种行为只不过是“纸上谈兵”。看再多的书都不如自己亲手试一试。俗话说的好：会打仗的士兵才是好士兵。再者，课本上的知识不一定是完全准确的，只有自己动手进行试验过才知道。实践是检验真理的唯一标准，这话不假。

1. 附录：源代码