



西安财经大学

《面向对象技术与编程课程设计》

题目 题目 10 宾馆房间管理系统

班级 计本 1801

学号 1831050010

姓名 陈伯硕

任课教师 马君

2019 年 6 月 6 日

目录

1 问题描述	2
2 基本要求	2
3 需求分析	3
4 概要设计	3
5 详细设计	3
5.1 date 类	3
5.2 Room 类	3
5.3 Guest 类	4
5.4 csv 类	4
6 调试分析	5
6.1 guest 类的日期传递问题	5
6.2 csv 问题	6
6.3 文件写入问题	6
6.4 文件多次写入	7
7 用户使用说明	8
8 测试结果	8
8.1 单元测试	8
9 程序设计总结	11
参考文献	11

1 问题描述

设计一个程序实现对宾馆房间的基本管理，可以实现：客房信息的录入功能；客人入住登记、客人退房结算；客房信息浏览功能，浏览全部客户的信息，客房信息和客户信息分别保存于不同文件；客房信息查询，查询空房间情况，实现按房间号查询等。

2 基本要求

(1) 至少包含四个类：Date 类（日期），客房 Room 类，主要包含客房信息（房号,类型，是否有客人等）及相关操作；客人 Guest 类，主要完成客户信息（身份证，入住时间，姓名，性别等）的相关操作；Manage 类实现对客房的管理。

(2) 用文本编辑器编写一个 room.txt 的文件，文件中应包含 20 条以上记录（房间的初始状态），再编辑一个 guest.txt 的文本文件，包含 10 条以上客人记录。在运行程序时自动载入。

在(2)中为了数据处理方便，替换成了csv文件

3 需求分析

设计程序模拟旅馆管理系统, 记录房间的使用情况和客人的住宿情况

4 概要设计

说明本程序中主程序的流程以及各程序模块之间的层次(调用) 关系。

5 详细设计

5.1 date 类

类似于课本[1]的date类的实现如图1

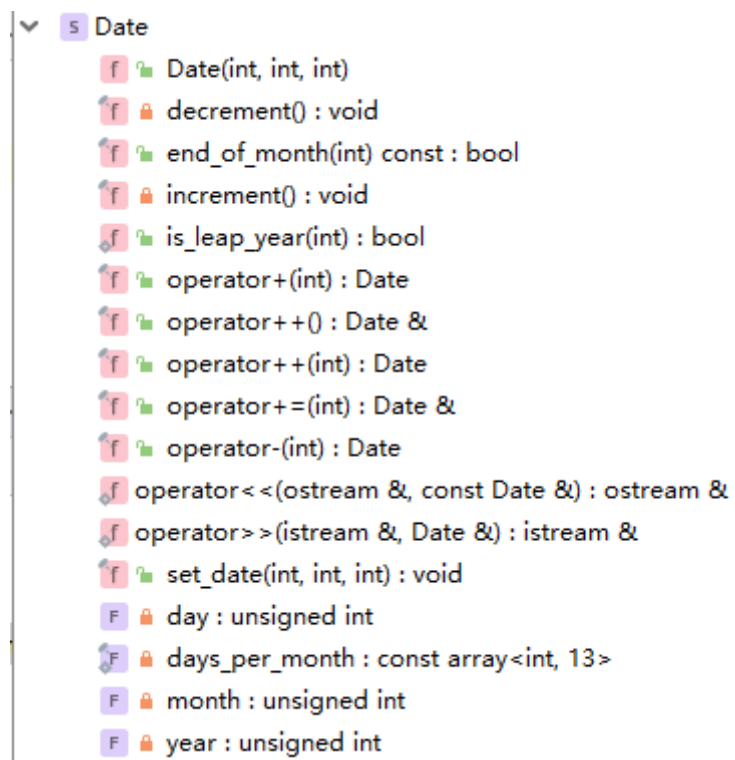


图 1: date 结构图

5.2 Room 类

简单的room实现如图2

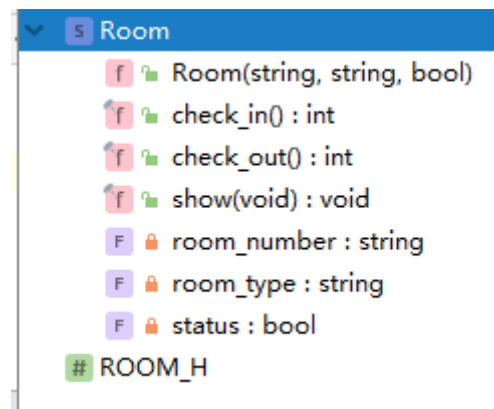


图 2: room 结构图

5.3 Guest 类

简单的Guest 类如图3

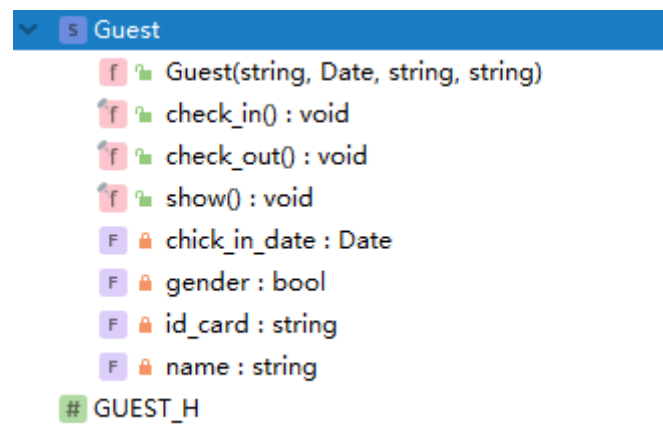


图 3: guest 结构图

5.4 csv 类

将对csv的操作封装成类,结果如图4

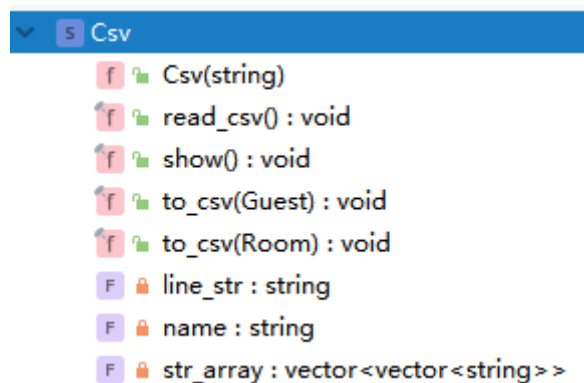


图 4: csv 结构图

6 调试分析

6.1 guest 类的日期传递问题

在guest类中日期总是如图5,无法传递正确日期

```
Ca\ guest.exe
His name is: Mr.a
ID_card: 100000
check in date: 1900-1-1

His name is: a
ID_card: 108
check in date: 1900-1-1
请按任意键继续. . .
```

图 5: 所有日期都是1900-1-1

经过测试(如图6), date的复制构造函数没有问题

```
d2 = d1;
cout << "after d2 = d1" << endl;
cout << "d1 is " << d1 << endl;
cout << "d2 is " << d2 << endl;

return 0;
```

```
Ca\ date.exe
d1 is 2010-12-27
d2 is 1900-1-1
after d2 = d1
d1 is 2010-12-27
d2 is 2010-12-27
请按任意键继续. . .
```

图 6: date的复制构造函数

如图7最后在构造函数加了一个this指针解决,具体原因未知.

```
-    chick_in_date = chick_in_date ;  
+    this->chick_in_date = chick_in_date;
```

图 7: 加了一个this指针解决

6.2 csv 问题

使用csv文件格式时注意每个单元格不要带",",否则会视为两个数据单元

6.3 文件写入问题

测试中发现, 运行写入文件的代码之后展示表格的时候没有新增的记录信息(如图8)

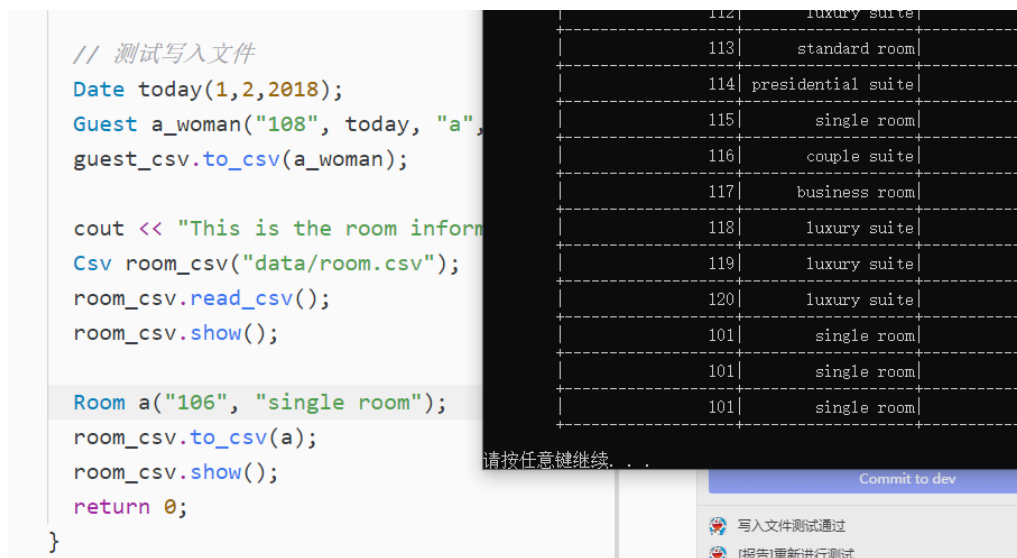


图 8: 错误的录入

原因:

程序读取信息时调用函数"read_csv()",但是在调用函数"to_csv()"写入文件时没有改变存储数据的数组"str_array"

解决方案:

如图9,在"to_csv()"函数体中调用一次"read_csv()",即可重新载入数据,方便数据的再次查看

```

117| business room| full|
118| luxury suite| empty|
119| luxury suite| empty|
120| luxury suite| empty|
101| single room| empty|
101| single room| empty|
101| single room| empty|
106| single room| empty|
1008| single room| empty|

任意键继续. . .
Room_csv.read_csv();
room_csv.show();

Room a("1008", "single room");
room_csv.to_csv(a);
room_csv.show();
return 0;

70 > /**
71 >  * 写入(针对room.csv)
72 >  */
73 > void to_csv(Room a) {
74 >     ofstream outfile(name, ios::app);
75 >     outfile << a;
76 >     outfile.close();
77 >     read_csv(); // 由于文件改变, 需要重新读取
78 > }
79 >
80 > void to_csv(Guest a) {
81 >     ofstream outfile;
82 >     outfile.open(name, ios::app);
83 >     outfile << a;
84 >     outfile.close();
85 > }
86 > private:

```

图 9: 修复问题

优化:

在实际应用中,可能写入CSV之后不在使用,造成系统开销增大,可以设置标志(flag)在"show()"函数使用时检测是否调用过"to_csv()",修改后测试结果如图10

```

16 csv.exe
17 | 101| single room| empty|
18 | 101| single room| empty|
19 | 101| single room| empty|
20 | 106| single room| empty|
21 | 1008| single room| empty|
22 | 1009| single room| empty|
23 | 1009| single room| empty|
24 | 1010| single room| empty|
25 | 1024| single room| empty|
26 |
27 | 请按任意键继续. . .
28 Room a("1024", "single room");
29 room_csv.to_csv(a);
30 room_csv.show();
31 return 0;

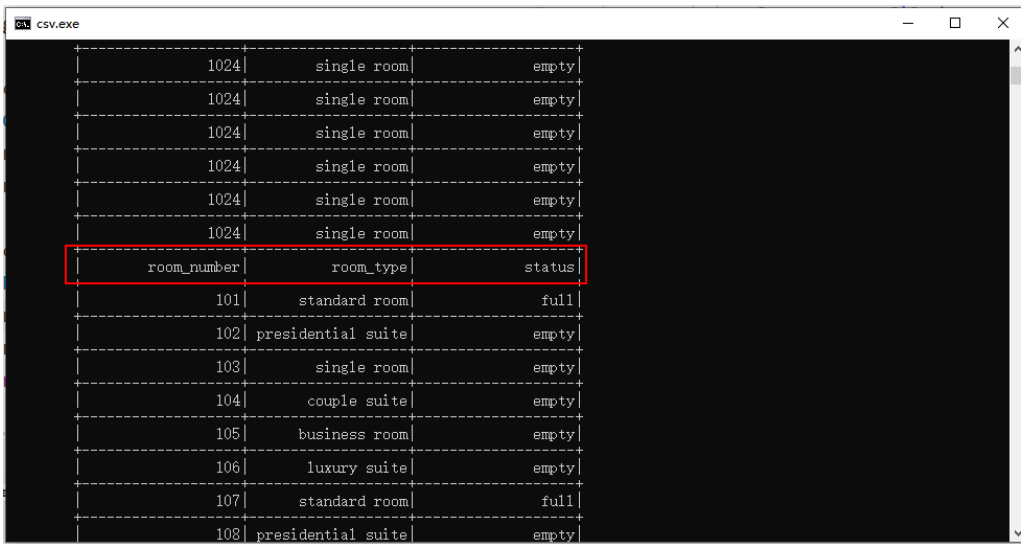
44 if (flag) { // 检测标志, 看是否改变csv
45     read_csv(); // 由于文件改变, 需要重新读取
46 }
47 // 绘制表格(第一行框架)
48 cout << "\t+";
49 for(size_t h = 0; h < str_array[0].size())
50     for(size_t k = 0; k < 19; ++k){
51         cout << "-";
52     }
53     cout << "+";
54 }
55 cout << "\n";
56 // 表格主主体数据
57 for(size_t i = 0; i < str_array.size(); ++i)
58     cout << "\t|";
59     for(size_t j = 0; j < str_array[i].size()

```

图 10: 引入标志来减少开销

6.4 文件多次写入

经过以上调整优化,发现每个表格打印了两次(如图11)



1024	single room	empty
1024	single room	empty
1024	single room	empty
1024	single room	empty
1024	single room	empty
1024	single room	empty
room_number	room_type	status
101	standard room	full
102	presidential suite	empty
103	single room	empty
104	couple suite	empty
105	business room	empty
106	luxury suite	empty
107	standard room	full
108	presidential suite	empty

图 11: 表格由于错误打印两次

错误原因:

由于”read_csv()”是使用函数”push_back()”入栈,重新调用相当于在原有二维数组的基础上增加内容,从而导致内容多次出现,产生错误,增加系统开销.

解决: 重构”to_csv()”函数, 将相应内容直接写到文件和数组中, 不再调用其他函数,具体修改如图12

```
-   outfile << a;
-   outfile.close();
+   vector<string> line_array = {a.get_room_number(), a.get_room_type(),
+                               a.get_status()};
+   outfile << a.get_room_number() << "," << a.get_room_type() << ","
+   << a.get_status();
```

图 12: 重新修改写入程序,直接写入数组和文件

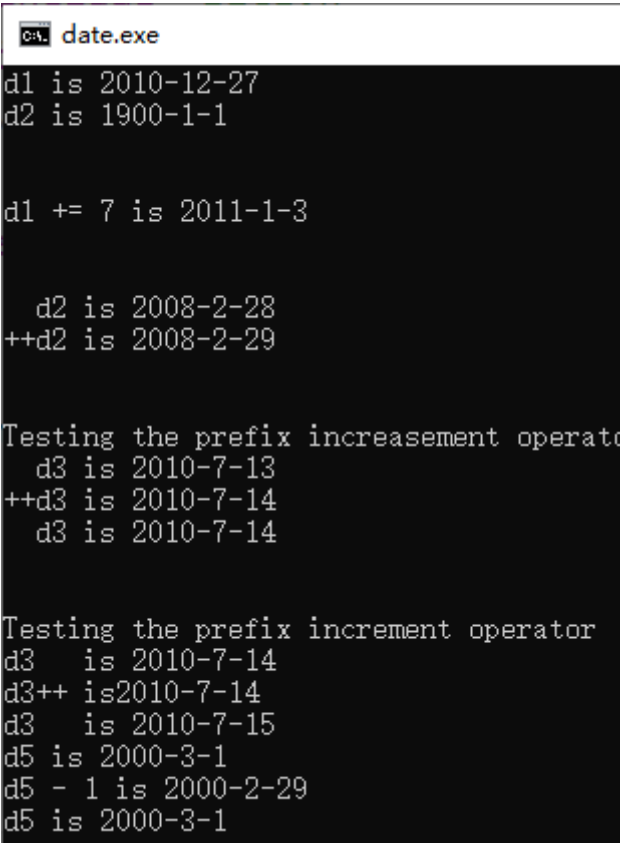
7 用户使用说明

说明如何使用你编写的程序，详细列出每一步的操作步骤。

8 测试结果

8.1 单元测试

对于每个头文件, 对应一个测试.cpp文件,分别对date(图13),room(图14),guest(图15),csv(图16)



```
CA: date.exe
d1 is 2010-12-27
d2 is 1900-1-1

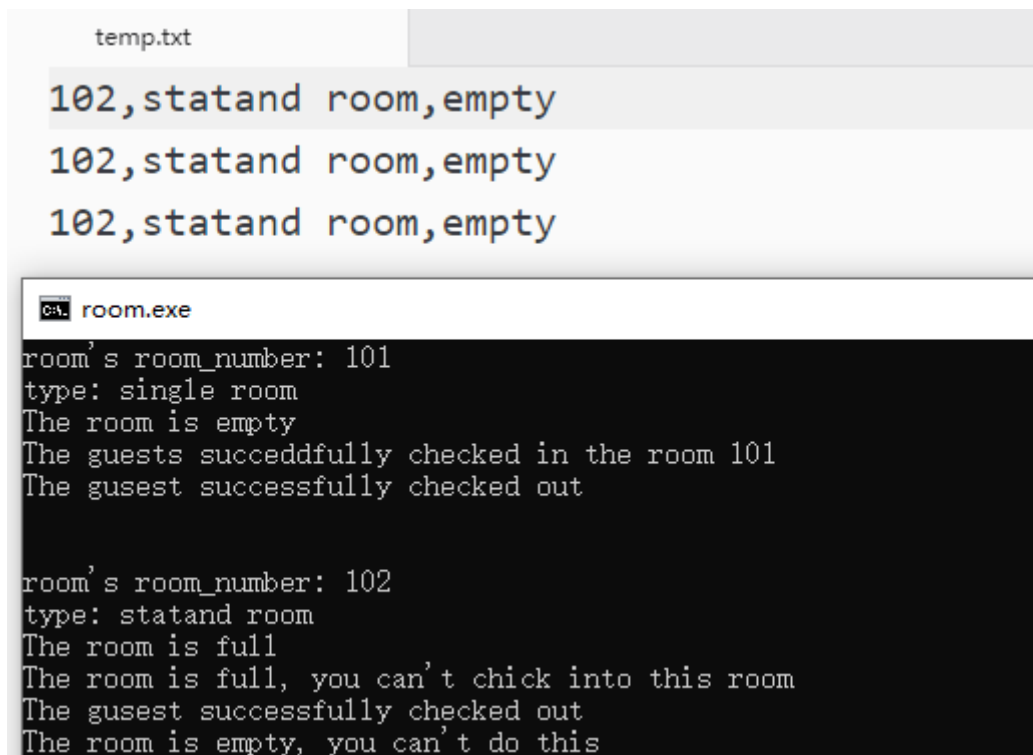
d1 += 7 is 2011-1-3

    d2 is 2008-2-28
++d2 is 2008-2-29

Testing the prefix increasement operator
    d3 is 2010-7-13
++d3 is 2010-7-14
    d3 is 2010-7-14

Testing the prefix increment operator
d3    is 2010-7-14
d3++ is2010-7-14
d3    is 2010-7-15
d5 is 2000-3-1
d5 - 1 is 2000-2-29
d5 is 2000-3-1
```

图 13: date.cpp 对date类的测试



```
temp.txt
102,statand room,empty
102,statand room,empty
102,statand room,empty

room.exe
room's room_number: 101
type: single room
The room is empty
The guests succedddfully checked in the room 101
The gusest successfully checked out

room's room_number: 102
type: statand room
The room is full
The room is full, you can't chick into this room
The gusest successfully checked out
The room is empty, you can't do this
```

图 14: room.cpp 对room的测试



```
guest.exe
His name is: Mr.a
ID_card: 100000
check in date: 1900-1-1

His name is: a
ID_card: 108
check in date: 1900-1-1
请按任意键继续. . .
```

图 15: guest.cpp 对guest类的测试

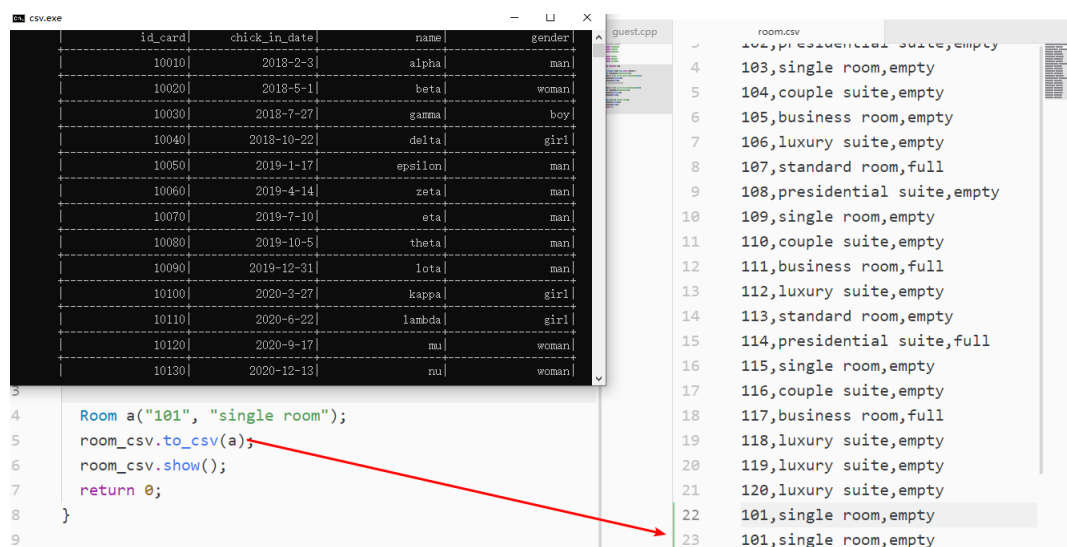


图 16: csv.cpp 对csv类的测试

9 程序设计总结

在这次程序设计中,探索各个模块的应用和项目的结构,开始习惯查阅参考文档^[5],同时开始学习开源目录的代码规范^[2],写代码开始注重简洁。

其中,谷歌代码规范给我很多启发,对于新手来说告诉我什么叫好看简洁的代码。^[3]当然,以为水平技术原因,只看懂了部分代码规范,但是许多特性可以让开发更有条理

在程序设计中,我学会了用程序设计的思维考虑问题,遇到了很多设计的问题,看出对程序设计者学习能力的要求,开发程序会不断遇到问题,要通过互联网和文档资料学会解决,探索更多的新技术,新方法,新理念。这一次尝试了git的分支功能,知道了开发工具对效率的重要性。

由于时间关系,大多数功能不够完善,代码不够简洁,需要更多时间来完善重构,建议老师们早点发布作业,来提高程序设计的质量。

参考文献

- [1] Paul Deitel and Harvey Deitel. *C++ How to Program*. Pearson, 9th edition, 2016.
- [2] fex team. 开源项目目录规范. <https://github.com/fex-team/styleguide/blob/master/project.md>. Accessed May 31, 2019.
- [3] google. Google style guides. <https://github.com/google/styleguide>. Accessed May 31, 2019.
- [4] Stanley B. Lippman, Jose Lajoie, and Barbara E. Moo. *C++ Primer*. Addison-Wesley Professional, 5th edition, 2012.
- [5] The C++ Resources Network. C++ reference. <http://www.cplusplus.com>. Accessed May 29, 2019.