

# 火车票订票系统作业要求

---

2019 ACM班 数据结构

## 概况

---

### 实现内容

本次作业要求实现一个类似于12306的火车票订票系统，该系统的评分包括**功能完整性**、**后端正确性**以及**相关文档**三部分。

该系统：

- 能将用户数据、购票数据、车次数据进行**本地存储**、并对其进行高效操作。
- 具有**用户友好的图形操作界面**，其中：
  - 能供管理员通过图形化方式管理系统，进行如增减车次，查询购票情况等操作。
  - 能供普通用户通过图形化方式使用系统，进行如注册登录，查询退票等操作。

### 程序结构

本次作业**强制要求前后端分离**，**前端通过网络与后端通信**。出于公平性考虑，本次作业对于前后端通信的接口进行了规定。

#### 后端正确性部分

后端部分在外存进行数据管理，向前端提供接口。评测的部分类似于传统OI题目，要求运行程序后进入响应输入的状态，在接受完整命令（单行命令或多行命令）后**立即**输出结果并**再次进入**响应输入的状态，直到命令要求退出系统为止。具体评测时采用管道检验输入输出是否正确。

**注意：后端部分强制使用C++。**

#### 功能完整性部分

前端部分调用后端，通过图形化方式向用户提供服务。调用方式，可以是作为c++编译时的一个组件；通过重定向与后端交互；或将后端构建为动态链接库并调用等等。实现的界面可以是Qt、VS等生成的，也可以是网页的形式。

**注意：前端部分语言不限。**

同时需支持**多用户同时操作**，以及完善且用户友好的前端。该部分没有给定的测试数据，将在最终展示时予以评分。

### 分组规则

共9组，每组四到五人，四人组至少有一名B班同学，五人组至少有两名B班同学。

### 内容分工

没有硬性的分工要求，由各组组长自行决定，原则上在考虑到组员水平的前提下工作量应当尽量平均。在组队完成后，组长尽快提交一份组员分工规划，完成后在文档中提交每个组员实际工作量的说明。该项会在最后为组内分数调整作为参考。

# 评测方式

---

## 性能评测（7分）

性能评测要求大家提供符合接口说明要求的后端程序，由给出的脚本进行评测。

**注意：与图书管理系统相似，脚本可能多次开关用户程序，请大家做好准备。**

程序运行时使用的数据应尽量存储在外存之中，**后端评测时会对占用内存大小进行严格的限制**。数据库强制要求使用文件存储的数据结构存储于外存之中，推荐使用B+树。

按照程序**运行速度与外存占用大小两方面**进行天梯排名，并从满分等距递减给分。

运行速度与外存占用大小在评分时为平行评分，即各占3.5分并分别给分。

**注意：通过所有测试即可得到75%的基础分。**

## 完善度评测（5分）

完善度是指用户能够完全以**图形方式**自然地使用所有前端提供的功能（包含但不限于后端提供的所有指令），以及完整的支持多用户系统（包括多用户同时操作的行为），并且能够正常的通过网络实现前后端通信。

## 易用度及文档（3分）

易用度包含系统是否美观、布局设计是否合理、文档是否完全等方面。

文档方面要求提供**完整的《开发文档》**（包括模块划分图、每个模块的功能、类设计、文件设计等）以及**完整的《使用手册》**（包括系统安装手册、用户手册等）以及各个组员工作情况的说明。

## 作业内Bonus（3分）\*

若你在作业中自行实现了其他**有用、有趣**的功能特性（不与主要要求冲突），助教组可酌情给出三分以内的Bonus，**可用于弥补性能评测、完善度评测以及易用度及文档中的失分**（即加分后该项大作业的分数不会超过15分）。

# 接口说明

---

## 要求实现的命令

参数具体要求见下方**数据格式**部分

### 用户相关

- 用户注册

- 格式

```
register *name* *password* *email* *phone*
```

- 返回值

```
*id*( or -1)
```

- 说明

**返回的id由2019开始（含2019），且2019号用户为默认管理员，每次新用户id依次递增，若注册失败则返回-1**

- 样例

```
register 张三 zhangsan zhangsan@sjtu.edu.cn 12345678
```

```
-> 2018
```

- 用户登录

- 格式

```
login *id* *password*
```

- 返回值

```
0 or 1
```

- 说明

1为登录成功, 0为失败

- 样例

```
login 2018 zhangsan
```

```
-> 1
```

- 查询用户信息

- 格式

```
query_profile *id*
```

- 返回值

```
*name* *email* *phone* *privilege* (or 0)
```

- 说明

未找到id则返回0

- 样例

```
query_profile 2018
```

```
-> 张三 zhangsan@sjtu.edu.cn 12345678 2
```

- 修改用户信息

- 格式

```
modify_profile *id* *name* *password* *email* *phone*
```

- 返回值

```
0 or 1
```

- 说明

1为登录成功, 0为失败

- 样例

```
modify_profile 2018 张三 zhangsan zhangsan@sjtu.edu.cn 87654321
```

```
-> 1
```

- 修改用户权限

- 格式

```
modify_privilege *id1* *id2* *privilege*
```

- 返回值

0 or 1

- 说明

1为成功, 0为失败

id1 要将 id2 改为 privilege(>0) 等级

普通用户不能使用该条指令

管理员可将其他用户升级为管理员

管理员不可将管理员降级成普通用户

管理员把管理员设置为管理员的行为定义为成功 (返回1)

- 样例

```
modify_profile 2018 2018 1
```

```
-> 0
```

## 车票相关

- 查询车票

- 格式

```
query_ticket *loc1* *loc2* *date* *catalog*
```

- 返回值

```
listnum(\n) [train_id loc(from) date(from) time(from) loc(to) date(to) time(to)
```

```
[ticket_kind num(ticket_left) price] * num(price) (\n)]*listnum
```

```
(or -1)
```

- 说明

查询loc1到loc2发车日期为date的catalog类列车

第一行返回查到的票共有多少行, 若为-1则说明查询非法; 接下来每行代表一张票的相应信息。按trainid字典序升序输出, 所有实数输出绝对或相对误差在 $10^{-6}$ 视为正确(包括下文所有操作)

(special judge)

- 样例

```
query_ticket 北京 上海 2018-03-28 CD
```

```
-> 1
```

```
c100 北京 2018-03-28 08:00 上海 2018-03-28 08:01 一等座 2000 765.50 二等座 2000 765.49 三等座 2000 765.48
```

- 带中转查询车票

- 格式

```
query_transfer *loc1* *loc2* *date* *catalog*
```

- 返回值

```
[train_id loc(from) date(from) time(from) loc(to) date(to) time(to) [ticket_kind num(ticket_left) price] * num(price) (\n)] * 2
```

(or -1)

- 说明

查询loc1到loc2发车日期为date的均为catalog类两列车，使得这两列车总时长（含中转等待时间）最短。

若有多组合法的均为最短时间中转方案，**则随意输出一种即可，spj会验证这点。**

第一行若为-1则说明查询非法；接下来两行代表两张可以用于中转的票务信息。

- 样例

```
query_transfer 北京 上海 2018-03-28 CD
```

```
-> c101 北京 2018-03-28 08:00 夏威夷 2018-03-28 08:01 一等座 2000 765.50 二等座 2000 765.49  
三等座 2000 765.48
```

```
-> c102 夏威夷 2018-03-28 08:02 上海 2018-03-28 08:03 一等座 2000 765.50 二等座 2000 765.49  
三等座 2000 765.48
```

- 订购车票

- 格式

```
buy_ticket *id* *num* *train_id* *loc1* *loc2* *date* *ticket_kind*
```

- 返回值

1 or 0

- 说明

1为成功，0为失败

用户id 购买 num 张 date 这天的 train\_id 这辆车从loc1到loc2的ticket\_kind这种票

- 样例

```
buy_ticket 6666 1 c101 北京 夏威夷 2018-03-28 一等座
```

```
-> 1
```

- 查询购票信息

- 格式

```
query_order *id* *date* *catalog*
```

- 返回值

```
listnum(\n) [train_id loc(from) date(from) time(from) loc(to) date(to) time(to) dist  
[ticket_kind num(ticket_left) price] * num(price) (\n)]*listnum
```

(or -1)

- 说明

查询用户id购买的date这天类别为catalog的所有车票

第一行返回查到的票共有多少行，若为-1则说明查询非法；接下来每行代表一张票的相应信息。**输出顺序随意，保证总和正确即可**

- 样例

```
query_order 6666 2018-03-28 C
```

```
-> 1
```

```
-> c101 北京 2018-03-28 08:00 夏威夷 2018-03-28 08:01 一等座 1 765.50 二等座 0 765.49 三等座 0 765.48
```

- 退订车票

- 格式

```
refund_ticket *id* *num* *train_id* *loc1* *loc2* *date* *ticket_kind*
```

- 返回值

```
1 or 0
```

- 说明

退订id订购的date那天train\_id这列火车从loc1到loc2的ticket\_kind这种票

- 样例

```
refund_ticket 6666 1 c101 北京 夏威夷 2018-03-28 一等座
```

```
-> 1
```

## 车次相关

- 新建车次

- 格式

```
add_train *train_id* *name* *catalog* *num(station)* *num(price)* *(name(price) ) x num(price)*
```

```
*[name time(arriv) time(start) time(stopover) (price) x num(price) ] x num(station)*
```

- 返回值

```
1 or 0
```

- 说明

添加车号为train\_id, 名为name的车, 该车经过num(station)站(<=60), 共有num(price)种票价(<=5), 并在之后——列出是哪种票。

接下来的num(station)行给出各个车站相关信息。

注意刚刚新建的车次中所售车票**不能被query\_ticket查询到**, 需要使用sale\_train指令开始发售后才能被查询或购买。

- 样例

```
add_train abc123456 G123456 G 2 1 商务座
```

```
北京 xx:xx 08:00 00:00 ￥0.0
```

```
夏威夷 08:01 xx:xx 00:00 ￥1.5
```

```
-> 1
```

- 公开车次

- 格式

```
sale_train *train_id*
```

- 返回值

```
1 or 0
```

- 说明

将train\_id的车次的车票开放发售

已发售的车次不能再发售（返回0） **注意：已发售的车次不能delete 或 modify**

- 样例

```
sale_train abc123456
```

```
-> 1
```

- 查询车次

- 格式

```
query_train *train_id*
```

- 返回值

```
*train_id* *name* *catalog* *num(station)* *num(price)* *(name(price) ) x  
num(price)*
```

```
*[name time(arriv) time(start) time(stopover) (price) x num(price) (\n)] x  
num(station)*
```

- 返回值

- 说明

查询列车号为train\_id的列车

若查询的车次不存在，返回0

- 样例

```
query_train abc123456
```

```
-> abc123456 G123456 G 2 1 商务座
```

```
-> 北京 xx:xx 08:00 00:00 ￥0.0
```

```
-> 夏威夷 08:01 xx:xx 00:00 ￥1.5
```

- 删除车次

- 格式

```
delete_train *train_id*
```

- 返回值

```
1 or 0
```

- 说明

删除train\_id这次列车，删除已售票的列车为非法操作

- 样例

```
delete_train abc123456
```

```
-> 1
```

- 修改车次

- 格式

```
modify_train *train_id* *name* *catalog* *num(station)* *num(price)* *(name(price) )
x num(price)*

*[name time(arriv) time(start) time(stopover) (price) * num(price) (\n)]
*num(station)*
```

- 返回值

```
1 or 0
```

- 说明

修改train\_id列车的信息为新信息，修改已售票的列车为非法操作

- 样例

```
modify_train abc123456 G123456 G 2 1 商务座
```

```
北京 xx:xx 08:00 00:00 ￥0.0
```

```
夏威夷 08:02 xx:xx 00:00 ￥1.5
```

```
-> 0 /* 刚才已经删掉了 */
```

## 管理相关

- 删库命令

- 格式

```
clean
```

- 返回值

```
1 or 0
```

- 说明

整个系统初始化，恢复到原始状态

- 关闭系统

- 格式

```
exit
```

- 返回值

```
BYE
```

## 数据格式

### 参数格式

说明：如未说带中文，则一定不带中文

- name

**40B**以内无空格非空字符串，带中文

- password

**20B**以内无空格非空字符串

- email

**20B**以内无空格非空字符串



- phone  
20B以内无空格非空字符串
- id  
20B以内无空格非空字符串
- privilege  
0-2数字，0代表未注册，1代表注册用户，2代表管理员
- loc  
20B以内无空格非空字符串，带中文
- date  
xxxx-xx-xx  
默认为x，换为数字，要求补0
- catalog  
10B内大写字母非空字符串，包含所有车次类型的子集
- ticket\_kind  
20B以内无空格非空字符串，带中文
- train\_id  
20B以内无空格非空字符串
- num  
整数
- time  
xx:xx  
默认为x，换为数字，要求补0
- price  
以¥开头，后接一个浮点数

## 问题说明

---

### 关于用户系统

后端评测时的用户系统仅用于存储用户数据库，并不维护“当前用户”这一概念。

前端应当完整地支持多用户同时操作以及完善且合理的用户系统。

### 关于权限

对于权限的检测应在**后端的前端**完成，故默认所有行为均在权限上已经过检验（即默认合法）。

关于权限的部分说明已在特定指令处给出。

未给出的部分，遵循“**管理员可以进行一切操作，用户只允许操作涉及到自己的部分**”这一原则。

### 关于车次

我们默认**每天的车次是相同的**，所以新建的车次和删除车次都是作用在所有日期上的。我们只需要管理**2019-06-01至2019-06-30（含首尾）内发车的列车即可**。

## 关于车票

我们默认**每种列车的每种座椅种类**，初始票数均为2000。

## 关于管理相关命令

clean 命令用于清空数据，方便进行下一轮评测（方便删库跑路）。

exit 命令用于停止交互。

## 关于系统安全

在该系统中，后端信任任何发来的数据，故如果想进行安全检查以及权限检查，应在后端的前端进行。

【前端（网页或GUI）】->【后端的前端（确认通信安全可靠、确认权限、整理信息格式）】->【后端】

## 关于头文件限制

原则上主体功能的主要部分（如数据结构设计，中转查询算法等，不包括GUI绘制和网络通信）不应**过分依赖**于外部库（如stl容器），一经发现**会导致成绩降低**，如果实在拿捏不好程度可咨询助教组。如pthread，ctime等用于锦上添花的库的使用是**支持的**。

## 关于评测机

保证评测机（后端运行环境）至少是四核的。平台**未指定**（这意味着后端需要适配多平台（不用奇奇怪怪的库问题不会很大））。

## 有没有什么推荐的前端啊

推荐写个网页，后端用python或者js脚本黏合一下；

推荐使用Qt，一站式解决GUI问题；

推荐开发新玩法。但要保证支持与后端的网络通讯。

## 去哪儿找服务器呢？

解决方案（后端放置的位置）有：

1. 本机demo, localhost
2. 使用交大宿舍ip，如果有路由器就把使用的机器设置为DMZ
3. BAT（划去）等等找个廉价的学生服务器
4. 花式使用境外小服务器

## 自学Qt有什么好的资料吗？

中科大一个较全面的指南<https://qtguide.ustclug.org/>

一个关于网络的例子<https://blog.csdn.net/rodgerjie1993/article/details/51476456>

一些关于打包的东西<http://tieba.baidu.com/p/3730103947>

## 这次的命令不像上次图书馆系统一样允许参数缺省吗？

是的，这样大家parser很好写：D

## 为了支持新的功能，我能不能增加一些命令？

理论上可以，但是要注意新增加的命令不要和原有命令冲突（就是第一个表示什么命令的词不要一样就对了）

## 前端如果只是展示的话，是不是系统任意？Windows Only也行？

是的，但是推荐使用一些跨平台的东西，比如写网页，比如Qt

## 我想写个网页，该去看点什么？

Qt相关：参考上面对应的问题

HTML, CSS, JS, TCP/IP原理 <http://www.w3school.com.cn/index.html>

Bootstrap, Python, Django, Node.js, HTTP原理 <http://www.runoob.com/>

Flask <http://flask.pocoo.org/docs/0.12/>