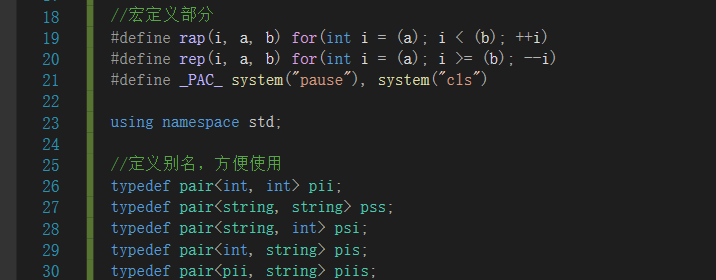
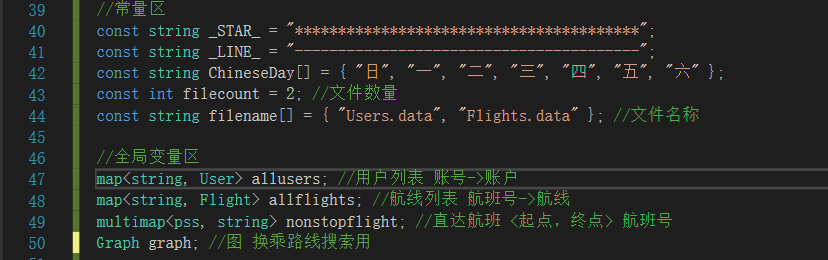
# 6.5数据结构课设代码讲解

**个人习惯：**



图方便的一些个人习惯

**全局常量、变量**



常量区：

前两个为了界面好看一点，第三个用来输出 周X，最后两个用来保存文件

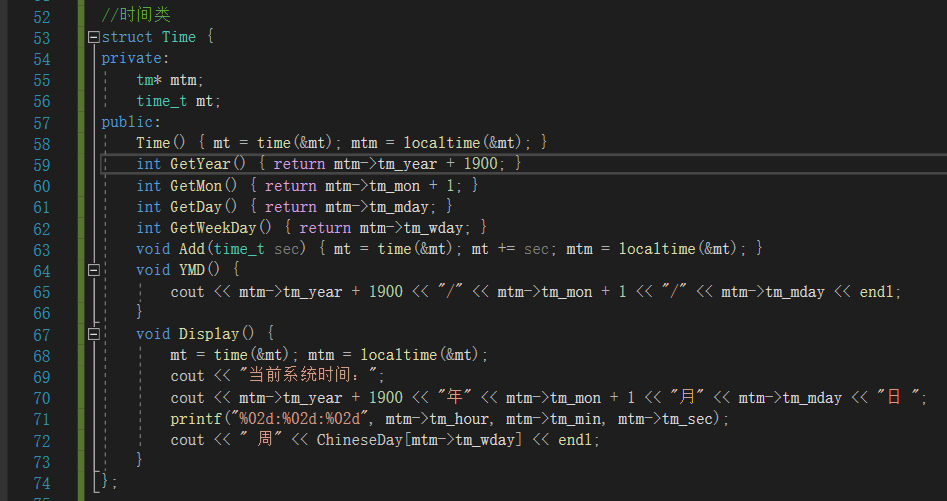
变量区：

前两个map映射分别是所有的 用户/航线 列表，账号/航班号 对应 用户/航线

第三个multimap多元映射，存直达航线，<起点，终点> 对应 航班号

第四个自定义Graph类（后面有讲），用于 换乘功能 的实现。

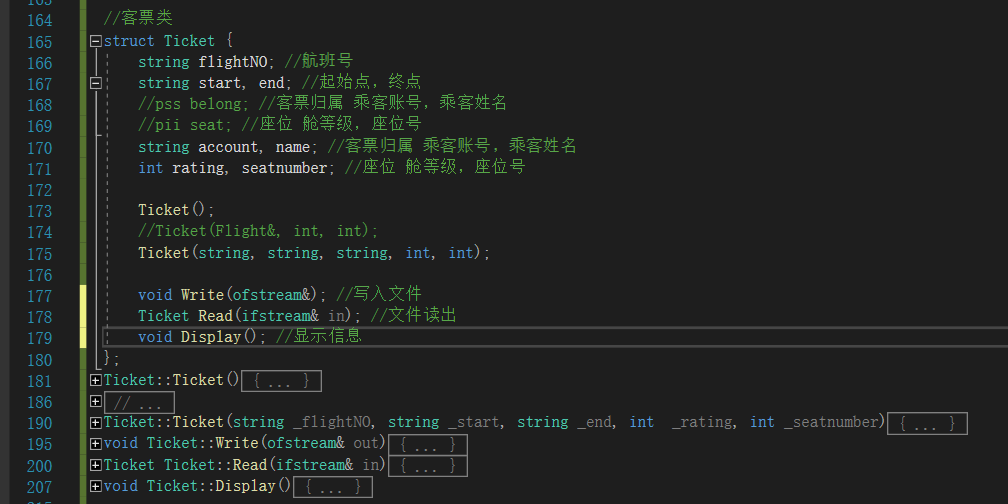
**有关 时间类 的实现：**



结合 time.h系统库实现的一个自定义时间类

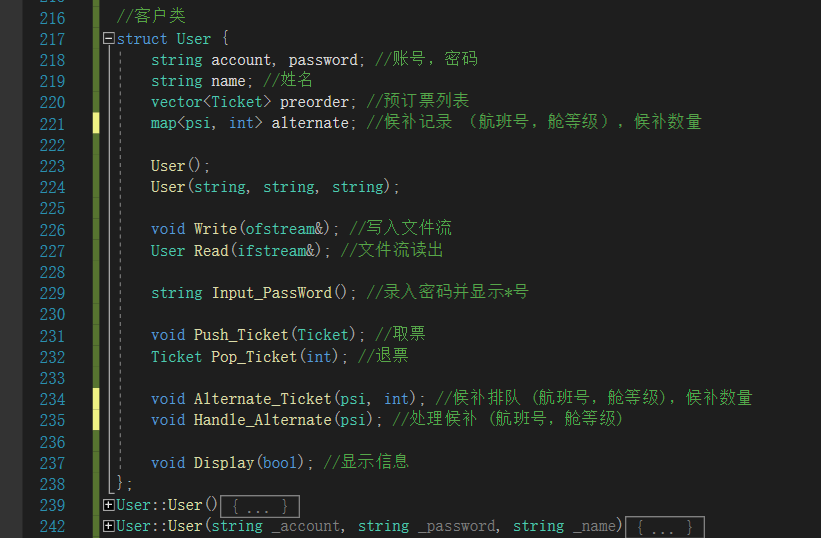
可以以中文形式输出时间

**有关 客票类 的实现：**



为了方便 乘客/航线 购票、退票等操作，于是把客票给“实例化”了。

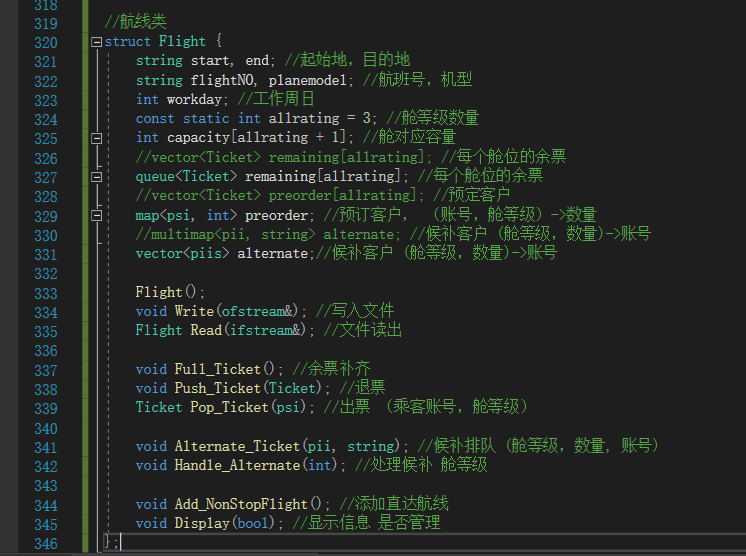
**有关 乘客类 的实现：**



说一下候补记录吧 用的map映射，<航班号，舱等级> 对应 需求数量。

考虑的是查找、删除的时间O（LogN）比较快。

**有关 航线类 的实现：**



实现过程中，最后三个成员的类型改了几次，最后确定了用

1. Queue队列 保存 X等舱的余票

2. Map映射 <客户账号，舱等级> 对应 购票数量 保存已购票的用户

3. Vector向量 <<舱等级，需求数量>，客户账号> 保存候补购票的用户

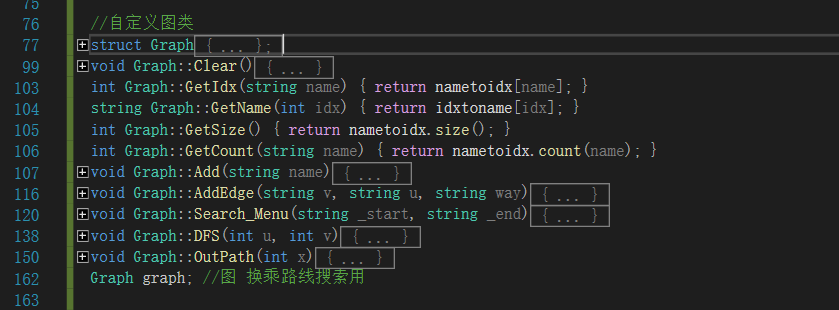
第一个是因为，不需要考虑具体用到哪张票，只需要支持出票、入票操作就行

第二个还是因为查找、删除的时间较快。

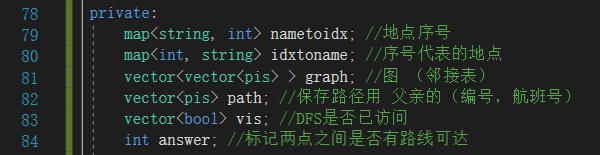
第三个本来想用multimap多元映射的，但是考虑到有一个候补时间的先后问题，之能退而求其次，用vector了（至于为什么不用Queue,因为它不支持随机访问。）

**有关 换乘功能 的实现：**

设计了一个Graph类：



成员如下：

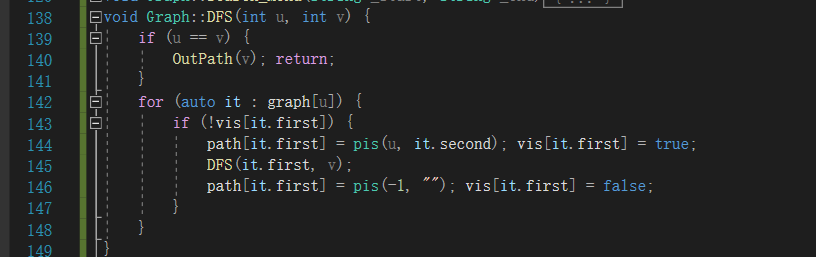


两个map映射，分别把 地点名 和 对应序号 关联起来

一个二维vector向量，用于存图的邻接表

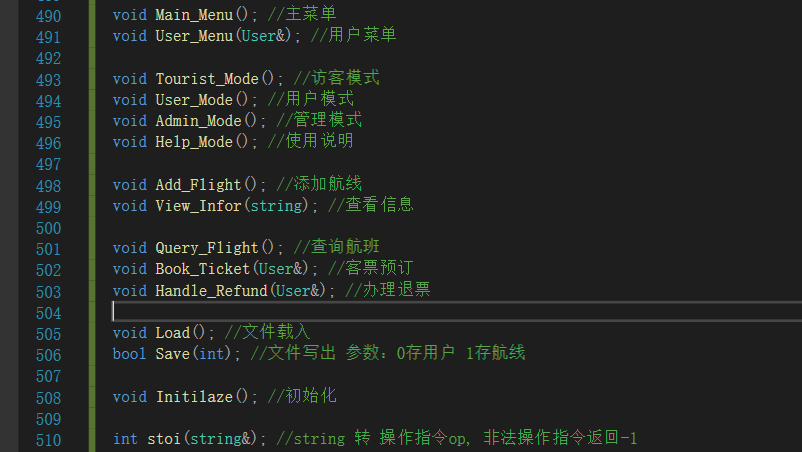
其他都有注释就不解释了。

寻路算法：

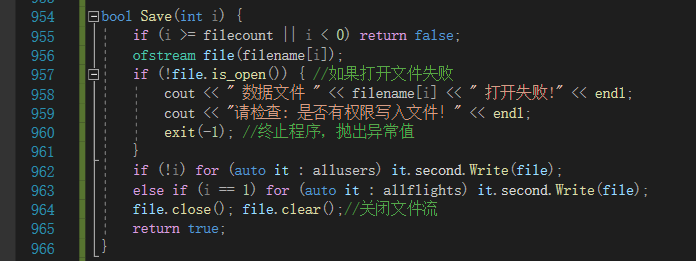


因为要输出所有的路径详情，所以采用的就是单纯的DFS暴搜

**菜单函数一览：**

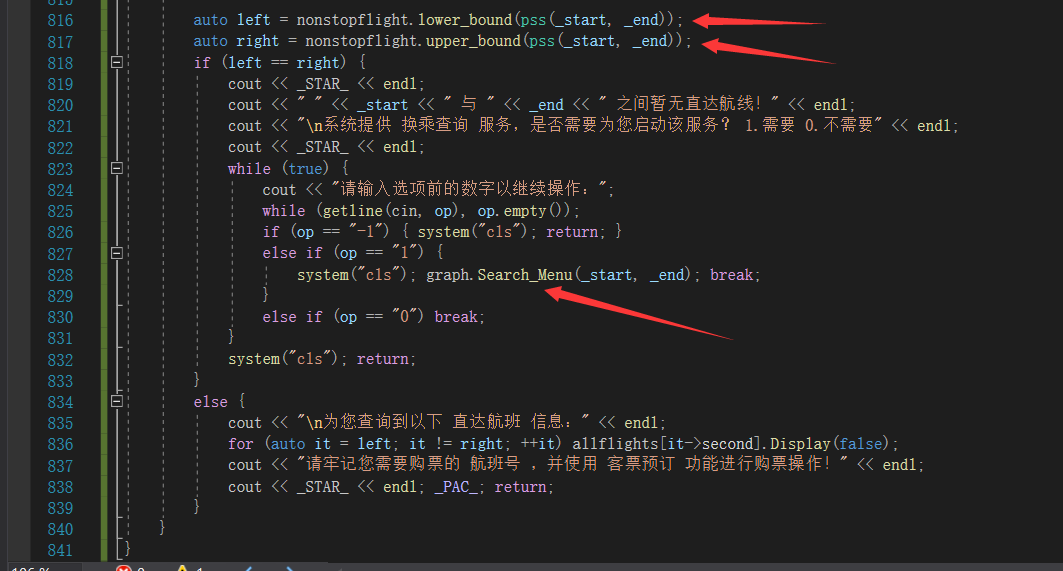


**文件存储及读入**



只保存了用户信息和航线信息，因为都为类设计了单独的读入、写出函数，所以文件写出、读入不难实现。

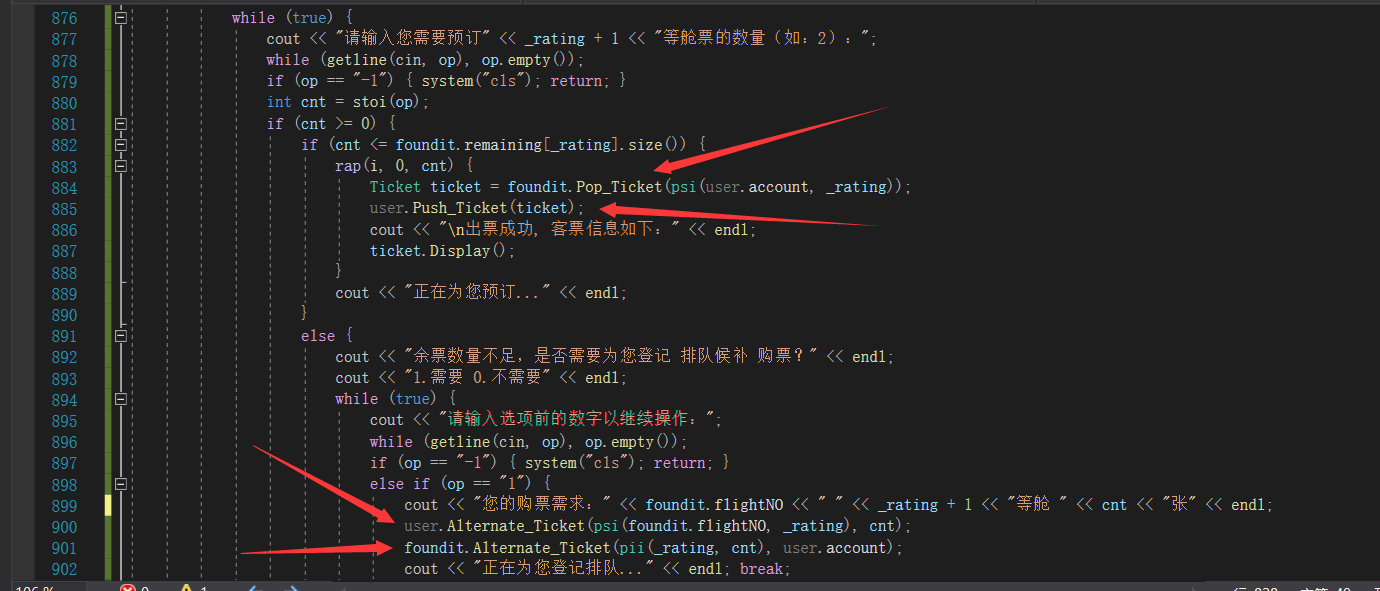
**查询航线功能：**



优先在 Multimap多元映射结构的nonstopflight里面 二分查找 直达航线

若找不到则启用Graph的DFS暴搜 换乘路线

**订票业务：**



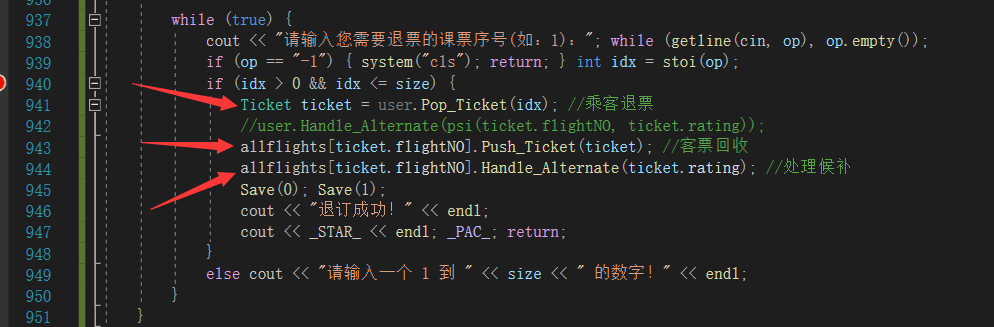
如果余票充足：

航线类弹出票，用户类接收票

如果不足：

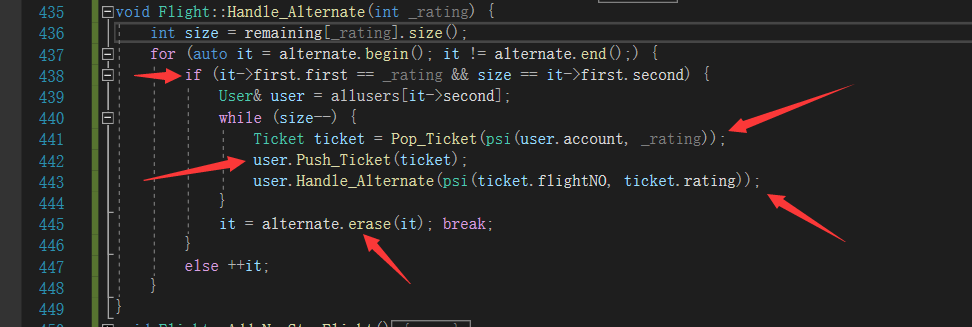
用户提出候补申请，航线类接收申请

**退票业务：**



乘客退票，用户类弹出票，航线类接收票，并处理候补订单

候补订单处理：



首先找到第一位<舱等级，需求票数>符合要求的用户，航线类将用户所需票弹出。用户接收票，并取消自己的候补申请。

最后，航线类将此条申请的用户从候补列表删除。

**A8**

**A7**

**A6**

**A5**

**A4**

**A3**

**A1**

**A2**

郑州

长沙

株洲

南京

西安

北京