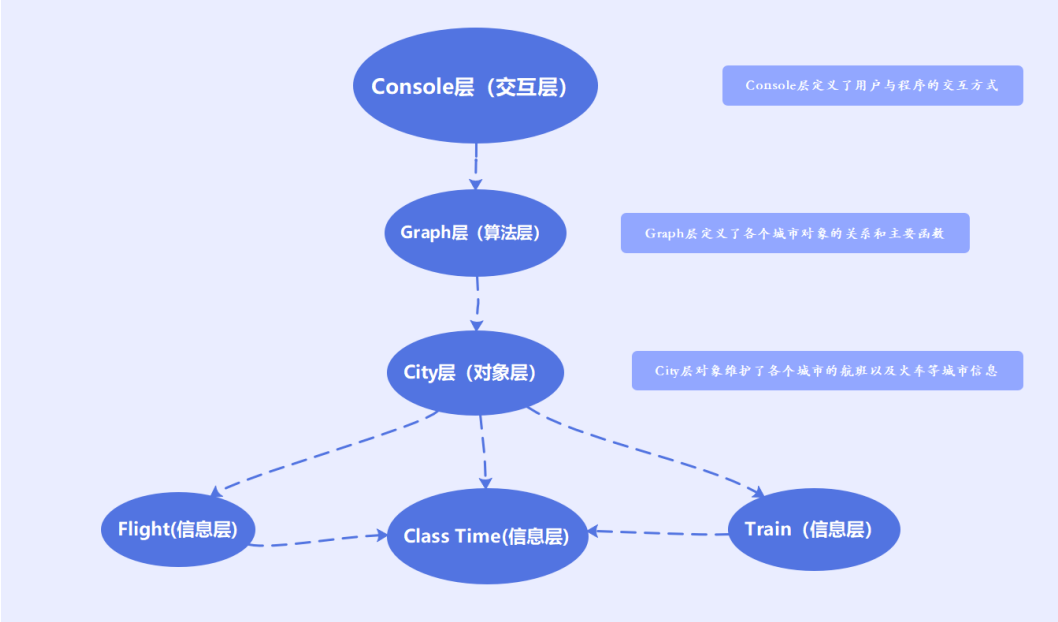
Data structure Project 报告

一、需求分析

本题最重要的需求是从实际问题出发，要求输出依据价格以及时间快慢而定的推荐路线；此程序需要能够增删航班/火车信息，满足对城市信息的编辑要求；此程序需要满足查询功能，用户输入城市名之后，程序应该在终端打印从该城市出发的所有航班及火车的详细信息（时刻表）。

二、概要设计



经过考虑我认为图邻接表是解决此问题的一个良好的数据结构。

在最底层我定义了两个存储基本信息的类：Time 类以及 FlightTrain 类，Time 储存了航班信息将会用到的时间，其成员包括年、月、日、小时以及分钟，而定义 FlightTrain 类的初衷是为了储存航班以及列车的详细信息，实际上定义 FlightTrain 类是经过优化后的处理方案。[1]

再上面一层定义了 City 类，这在整个图结构中可以视为顶点，和一般顶点数据结构不同的是，为了应对大量数据，没有使用一个链表直接储存所有的“边”（即航班火车班次）而是采用了一个二维数组分别储存从该城市到其他各个城市第一个航班/火车信息的指针，这样为接下来的算法的实现提供了诸多便利。除了相关数据成员，类中还定义了包括信息编辑在内的诸多成员函数。

再接下来定义了 Graph 类，Graph 类是所有类的友元，保证了 Graph 在算法实现过程调用底层数据的权限。这一层中除了定义了顶点集，以及当前城市数（即图的当前顶点数），

[1] 最开始做 PJ 其实是将航班信息与火车信息分别定义为用 Flight 以及 Train 储存，但考虑到二者数据成员（以及成员函数）的同构性，最终决定让二者合并，这样能精简整体的模型。

还定义了丰富的其他成员来储存主要算法的结果，以便进行高效的输出。Graph 是算法实现的层级，所以在这一层定义了许多用户交互将会直接调用到的函数比如输出最快路线、最便宜路线的 ProperPath 函数、用于信息增删编辑的 insert 函数以及 Delete 函数。

在 console 这一层定义了程序与用户之间交互的方式，其主要实现就是源文件中的 console.cpp，在这一层中有着比较完善的异常处理机制，以提高程序整体的鲁棒性。

三、详细设计

为了达到模块化编程的效果，本次 pj 程序主要包含了 8 个文件（除了 pj 报告以及说明运行环境、对各个文件描述的 readme.md 文件）device.h、device.cpp、err_out.cpp、err_out.h、constant.h、console.cpp 以及输入源文件 Flight.txt、Train.txt。

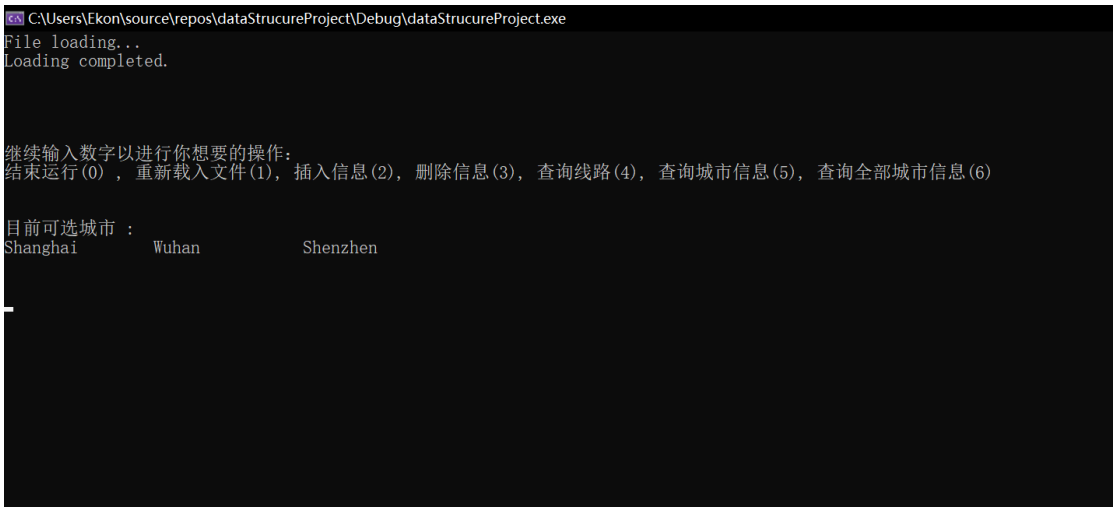
为了成功运行，需要按照实际储存路径修改 device.cpp 文件中 Load 函数读取输入源文件的路径。

四、调试分析

测试数据即储存在 Flight.txt 以及 Train.txt 中，程序运行后将不断执行用户发出的指令，直至用户决定输入 0 指令退出。程序开始运行：

一】程序初始化

程序运行开始后会 load 文件并构造出实例化的 Graph 对象，此时 Grap 及其底层类中储存了各种详细信息。程序会打印出目前支持的指令，并将以及存储的城市打印在终端以便用户选择。



二】重新载入文件

有时输入文件经过修改后，需要对 Graph 进行更新，此操作指令下原来的 Graph 会调用析构函数引起一系列底层类的自我销毁，之后再实例化一个新的 Graph 类对象并加载数据。

```
继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen


1
正在删除原始信息...



删除完毕.
File loading...
Loading completed.



继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)

目前可选城市 ：
Shanghai        Wuhan           Shenzhen
```

三】插入信息

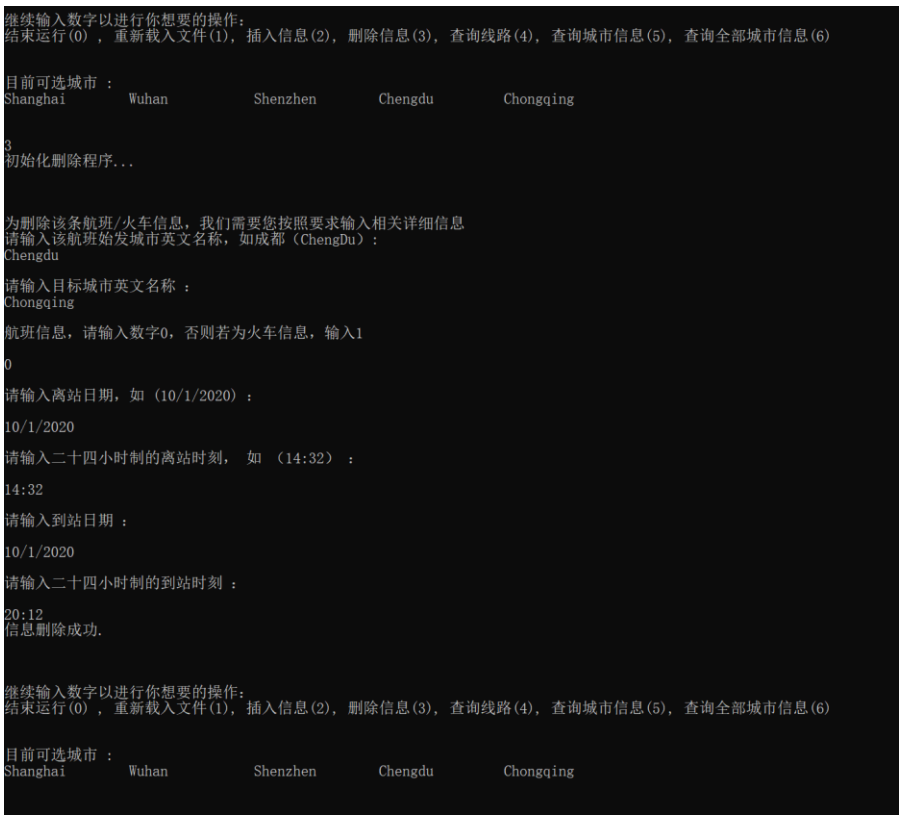
程序将会要求输入一系列航班或者火车信息，并提供相应反馈:

```
继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen


2
初始化插入程序...



请输入始发城市英文名称，如成都（ChengDu）：
Chengdu

请输入目标城市英文名称 ：
Chongqing

航班信息，请输入数字0，否则若为火车信息，输入1

0

请输入离站日期，如（10/1/2020）：

10/1/2021

请输入二十四小时制的离站时刻， 如 （14:32） ：

14:43

请输入到站日期 ：

10/1/2021

请输入二十四小时制的到站时刻 ：

17:56

请输入票价 ：

1076


信息插入成功



继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen        Chengdu         Chongqing
```

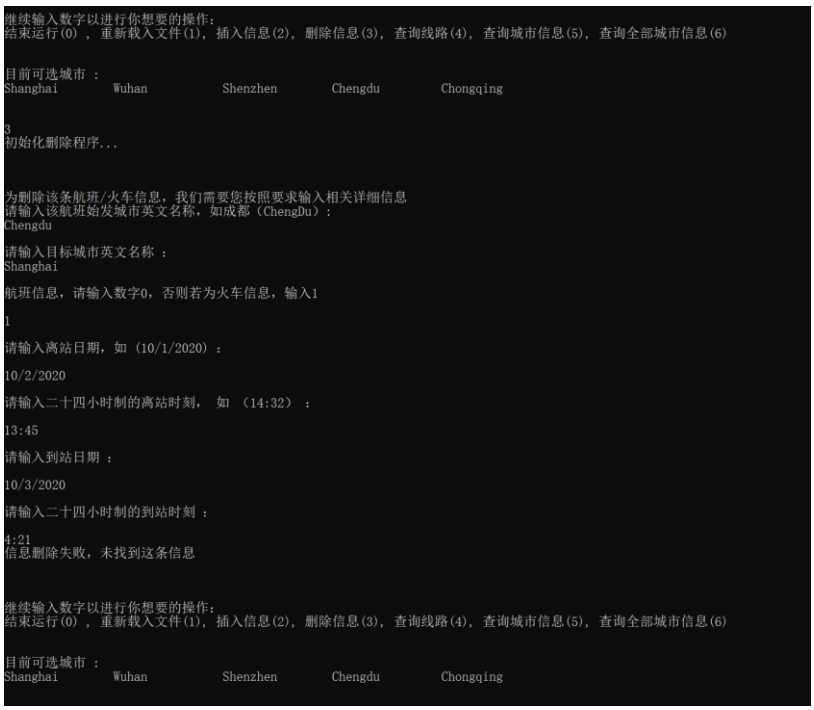
四】删除信息

用户发出删除信息指令，程序将要求用户输入想要删除航班、火车班次的具体信息，如果存在这条信息，删除并返回反馈，否则删除失败，返回失败反馈。

删除成功样例:



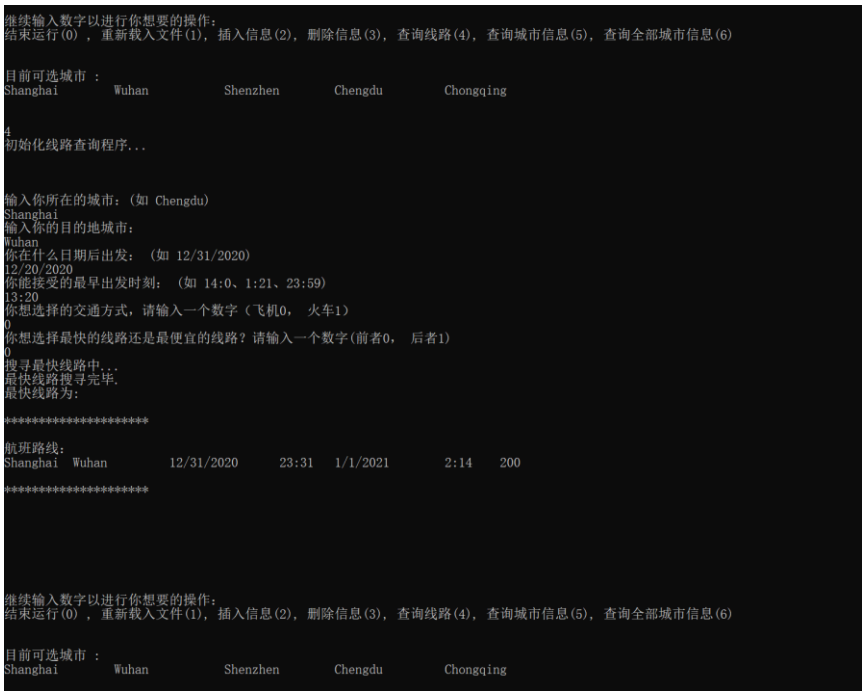
删除失败样例:



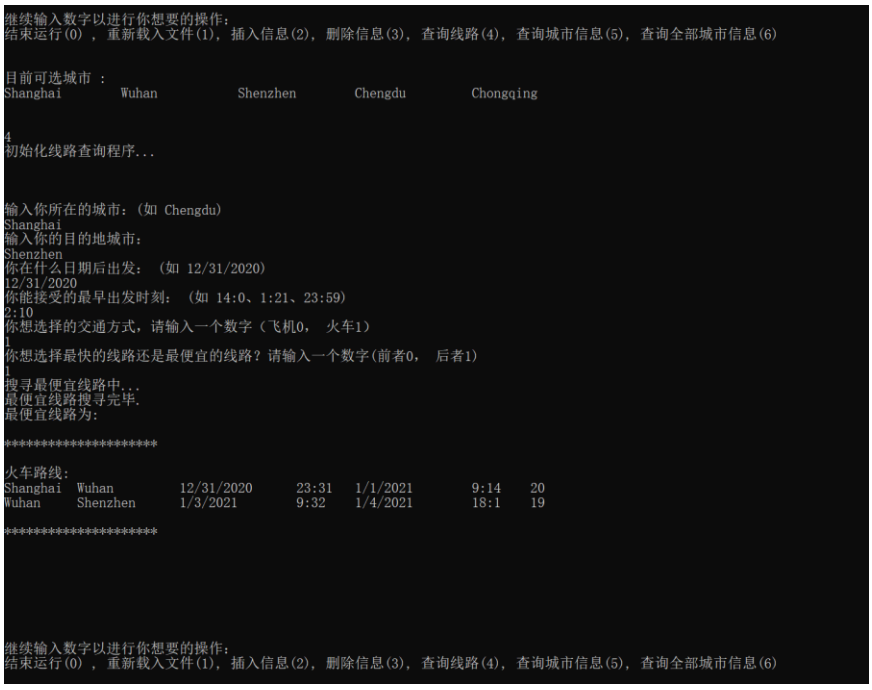
五】查询线路

用户输入相应的出发地、出发时间、目的地、交通方式以及线路优先选项，程序将判断有无满足要求的路线，如果有，则输出结果。

最快线路样例:



最便宜线路样例:



线路查询失败样例:

```
继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen        Chengdu         Chongqing


4
初始化线路查询程序...



输入你所在的城市：(如 Chengdu)
Chengdu
输入你的目的地城市：
Chongqing
你在什么日期后出发： (如 12/31/2020)
12/31/2020
你能接受的最早出发时刻： (如 14:0、1:21、23:59)
1:0
你想选择的交通方式，请输入一个数字（飞机0， 火车1）
0
你想选择最快的线路还是最便宜的线路？请输入一个数字(前者0， 后者1)
0
搜寻最快线路中...




在该时间点后，以此交通方式，从出发城市到目的地没有任何一条可行线路.
请修改目的地城市或者时间后重试.









继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen        Chengdu         Chongqing
```

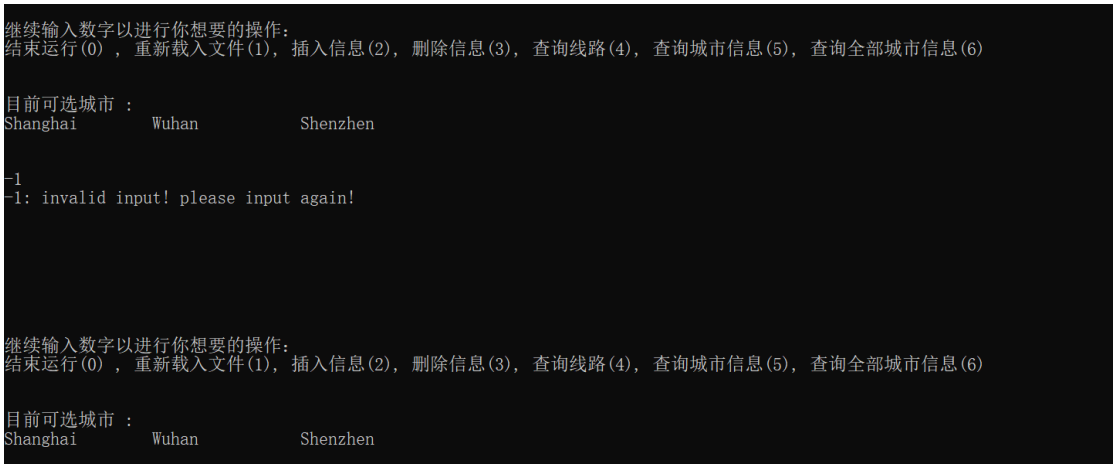
六】城市信息查询

以上海为例查询:

```
继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen        Chengdu         Chongqing


5
初始化城市查询程序...



输入你想要查询的城市：(如 Chengdu)
Shanghai
**********************************************

Shanghai 航班信息：

Shanghai  Wuhan         12/15/2020      3:32    12/15/2020      13:21   342
Shanghai  Wuhan         12/31/2020      23:31   1/1/2021        2:14    200
Shanghai  Shenzhen      1/2/2021        9:21    1/2/2021        14:56   290
Shanghai  Shenzhen      3/2/2021        3:32    3/2/2021        8:59    876
**********************************************


**********************************************

Shanghai 列车信息：

Shanghai  Wuhan         12/10/2020      4:24    12/11/2020      2:43    40
Shanghai  Wuhan         12/15/2020      3:32    12/15/2020      14:21   42
Shanghai  Wuhan         12/31/2020      23:31   1/1/2021        9:14    20
Shanghai  Shenzhen      1/2/2021        9:21    1/3/2021        19:56   90
Shanghai  Shenzhen      3/2/2021        3:32    3/3/2021        8:59    86
**********************************************


共 4条航班信息， 5 条列车信息









继续输入数字以进行你想要的操作：
结束运行(0) ， 重新载入文件(1)， 插入信息(2)， 删除信息(3)， 查询线路(4)， 查询城市信息(5)， 查询全部城市信息(6)


目前可选城市 ：
Shanghai        Wuhan           Shenzhen        Chengdu         Chongqing
```

七】指令异常处理

当输入的指令数字代码不符合要求时：



八】程序正常退出

当用户决定退出程序时：

主要算法时间复杂度分析

提供最快线路的函数是 Graph 类中的 Fpath 函数，其算法主体就是迪杰斯特拉算法，时间复杂度约为 O(n)。

提供最便宜路线的算法稍微较为困难，最开始直接想到用迪杰斯特拉算法或者 Floyd 算法直接求解，但这样得出的结果有可能导致航班班次直接相互冲突。最暴力的算法是遍历所有可能，但这样的处理会大幅降低性能。那么有没有一种算法既能提供符合逻辑的正确解，同时又能够保持良好的性能呢？答案是可以。主要改动是先记录下不用转趟的路线的最低票价，以此为基准来搜索路线，只有当路线的价钱小于这个基准值，我们才调用下一步的函数，否则我们就略过这个可能。事实证明这个策略在中等规模的数据下相当有效，在调试过程中我们发现，程序一般没有经过回溯或者只经过了很少的回溯就能找到有意设定的、不容易一下子发现的最便宜路径。这样子我们估计 Cpath 的实际平均性能应当居于 O(n)到 O(n2)之间，且在大多数情况下相当接近 O(n)。

这样我们就得到了能够较好解决问题的算法，为了进一步提升算法性能，个人认为算法中有许多处调用了 Time 类的成员函数，主要操作是对时间进行比较。如果我们不使用自己

定义的 Time 类，而是使用时间戳来储存时间数据并且作比较，应该能提升一部分算法性能。

五、课程设计总结

一学期下来确实收获了很多，数据结构给我提供了许多有力的工具来解决实际的工程问题，从简简单单的链表到复杂的 AVL 树乃至红黑树以及 B 树。数据结构提供了一种抽象的工具，而具体实现又是往往另外一回事，这一段的学习虽然将要告一段落，但它遗留下来的影响会一直常伴我。常言道：“大道至简，知易行难”，算法有时听起来很简单，但在实操过程中又总不那么轻易能做好，果然踏踏实实做事还是最终的解决途径。

这次 PJ 也是我真正第一次尝试模块化编程的实践项目，自己前前后后写了一千多行代码，删删补补最后剩下 6 个文件大概七八百行代码，一时感慨良多，程序终于跑起的那一刻十分兴奋：就像孩子终于长大的那种满足感。

最后，感谢一学期同学、老师以及助教的陪伴与付出，这几个月的收获离不开你们。

六、参考资料

《数据结构》，清华大学出版社