|  |
| --- |
| 哈尔滨工业大学(深圳) |
| **《数据结构与算法》实验报告** |
|  |
| 实验二  栈和队列的应用  学 院: 机电工程与自动化学院   |  |  | | --- | --- | | 姓 名: | 葛旭 | | 学 号: | 190320517 | | 专 业: | 自动化 | | 日 期: | 2021-3-30 | |

# 一、问题分析

一：火车进站。即将火车进站模型转化为火车id的压栈出栈问题。

二：计算缓冲车道。将乱序的火车按顺序出站。即用队列模型来解决。

# 二、详细设计

## 2.1 设计思想

（1）判断输入的出站序列是否可由输入的进站序列通过进出栈得到：通过依次将进站的火车压栈，然后与出站的火车进行对比。如果相等就一直弹出，直到不相等为止，不相等就继续压栈，直到没有火车进站为止。

（2）判断需要几条缓冲车道：将乱序进入的火车以顺序驶出。依次遍历驶入的火车，如果该火车id小于所有队尾的火车id，则开辟新车道，如果大于队尾火车id，则选择差值最小的车道进入。

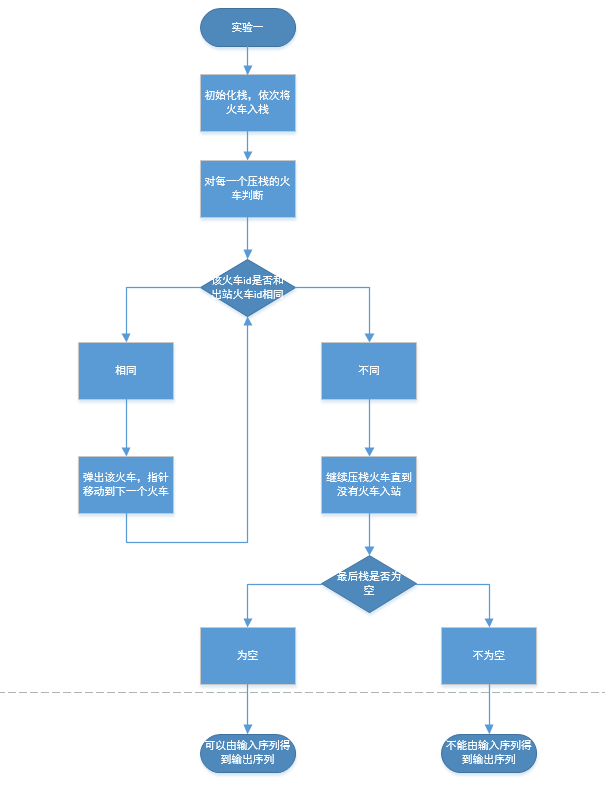
## 2.2 存储结构及操作

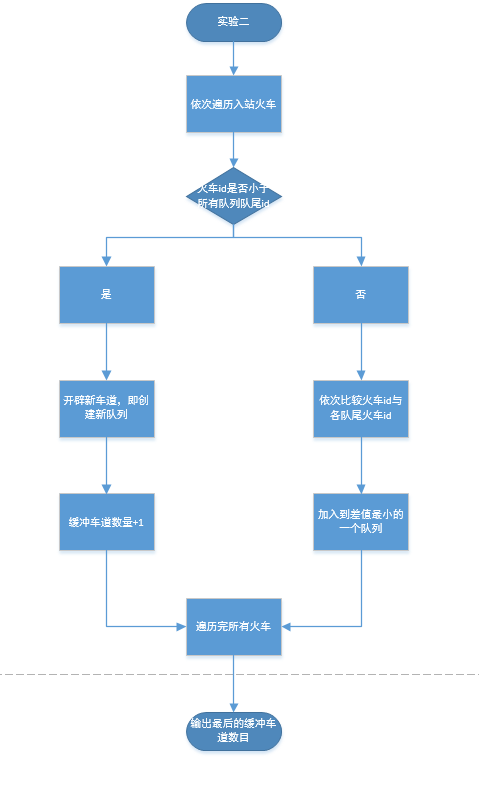
(1) 存储结构（一般为自定义的数据类型，比如单链表，栈等。）

用栈的结构来存储火车id，用队列的方式来存储火车序列，作为缓冲车道。

1. 涉及的操作（一般为自定义函数，可不写过程，但要注明该函数的含义。）
2. LinkStack\* InitStack()*//初始化栈*
3. int StackEmpty(LinkStack \*linkStack)*//判断栈是否为空*
4. int Push(LinkStack \*linkStack,int ele)*//压栈*
5. int Pop(LinkStack\* linkStack)*//弹出栈首元素*
6. int GetTop(LinkStack\* linkStack)*//返回栈顶端元素*
7. int StackLength(LinkStack\* linkStack)*//返回栈的长度*
8. int SeqLegal(int in[],int out[],int num)*//判断是否能由输入的火车序列得到输出的火车序列*
9. LinkQueue\* InitQueue()*//初始化队列*
10. int QueueEmpty(LinkQueue\* linkQueue)*//判断队列是否为空*
11. int EnQueue(LinkQueue \*linkQueue,int ele)*//进入队列*
12. int DeQueue(LinkQueue \*linkQueue)*//出队列*
13. int QueueLength(LinkQueue \*linkQueue)*//返回队列长度*
14. int GetHead(LinkQueue \*linkQueue,int\* ele)*//返回队列头部元素*
15. int GetRear(LinkQueue \*linkQueue)*//返回队列尾部元素*
16. int findClosestQueue(LinkQueue \*railQueue[],int usedQueue,int curTrain)*//找到最适合的缓冲车道*
17. int minBufferQueue(int out[],int num)*//找到最少的缓冲车道*

## 2.3 程序整体流程





# 三、用户手册

输入数据的方式；

用记事本打开Train.in文件，并把数据写入到对应的位置，程序会自动读取。

# 四、结果

# 五、总结

涉及到了栈和队列，学会了使用栈和队列解决实际问题。