C语言课程设计——公共交通行程管理系统

一、前言

1. 编写背景：

我国城市交通经历了从改革开放初期以自行车为主要交通工具，到20世纪90年代以公共汽电车为主体的发展阶段，在21世纪初，国家提出了公共交通优先发展的战略，城市交通进入了统筹协调、综合发展的新阶段。城市客运量在改革开放初期仅有132亿人次，到2018年已增至1262亿人次。近年来，城市公共汽电车车辆数与运营里程持续增长，轨道交通高速发展，传统出租数量稳中略降，互联网出行新业态涌现。城市交通呈现多元化发展。城市公共汽电车车辆数和运营线路长度持续大幅增长,截至2018年底，我国城市公共汽电车运营车辆数达67.34万辆，运营线路达60590条，运营线路总长度达119.95万公里。城市轨道交通自1969年北京市开通全国第一条城市轨道交通线路以来，截至2018年底，已有35个城市开通运营城市轨道交通，运营线路达到171条，2014-2018年，公共汽电车与出租汽车客运量逐年下降，轨道交通运输量却在逐年上升，轨道交通在城市客运中的地位越来越重要；但从绝对数来看，公共汽电车客运量占客运总量的比例仍很高，公共汽电车仍是城市客运的主要工具出租汽车客运量略有下降，但总体较稳定。运营里程达5295.1公里，完成客运量212.77亿人次，运营里程及完成客运量均居世界第一。其次，随着信息技术的发展，公交智能化深入推进，移动支付、智能公交电子站牌有效提升了顾客的乘车体验。

2020新冠疫情突然爆发，全国全世界都受到巨大的冲击与影响，就业、医疗、公共基础设施等面临的问题接踵而来，公共交通出行更是面领着巨大的问题与挑战。疫情之下严控人流密集场所，公交地铁受冷落。交通运输部颁布《客运场站和交通运输工具新冠肺炎疫情分区分级防控指南》，严控高风险地区的公交车拥挤度在每平方米≤4人，地铁满载率、拥挤度不能超过70%。恒大研究院统计得出，上海、广州等7城地铁运量在疫情期间大幅下降，2020年春节第一周客运总量仅为2019年同期的8.8%，第四周（22-25日）有所上升，但仍仅22.3%。

伴随中国新型冠状病毒肺炎(以下简称“新冠肺炎”)疫情防控向好态势进一步巩固，防控工作任务正在转向疫情防控与恢复经济社会运行两手抓，防控要求也从静态隔离转向动态防控。公共交通的出行特点，决定了其将在经济社会秩序恢复过程中担当疫情防控“主战场”角色。如果以一日出行链来还原城市中普通人的主要活动，可以发现，在家庭、办公或社交场所以及交通工具三类空间中，公共交通载运工具的传播风险相对较高。可以预见，伴随经济活动的活跃度逐步提升，素有城市血脉之称的交通系统将迎来真正意义上的持续考验。其中，公共交通的角色可能面临大幅转型，走出严格防控模式下的低活跃度状态，承担疫情防控“主战场”和经济恢复“保障线”的双重角色。

在此背景下，疫情防控与公共交通出行信息紧密结合，成为广大人民群众日常生活中不可或缺的重要安全保障，因此，编者编写了这个公共交通乘客行程系统，旨在模拟疫情管控与公共交通安全的结合，学习掌握C语言相关知识的同时深入了解公共交通在疫情爆发后的现状与未来的发展趋势，在提升自我专业能力的同时拓宽视野，增长知识。

2. 编写目的：

通过对公共交通出行服务类网站及app的使用者的调查和分析，对铁路12306及通信行程卡等应用的使用，对一些技术指标的学习和网络资料查询，我们编写出这一份终期报告。

本报告对于整个“公共交通乘客行程系统”进行了全面的用户需求和功能分析。包括可行性分析，需求分析，系统功能设计，代码实现，软件亮点和不足，集成测试等等。本报告明确了本软件系统架构设计，软件结构与数据结构设计，各模块之间的接口和调用，系统界面设计，系统功能设计，具体算法设计以及整个软件的源代码。

同时，该项报告也对代码进行一定程度的解释与概括，增强了后期测试人员对于软件的调试和验收的可读性与可修改性。

本报告的预期受众为全年龄群体包括所有有公共交通出行需求的用户。

3. 参考资料：

1. 王士元. C高级实用程序设计. 北京: 清华大学出版社. 1996年。
2. 周纯杰，何顶新等. 程序设计与应用（用C/C++编程）. 北京: 机械工业出版社. 2008年。

4. 参考软件：

铁路12306、通信行程卡等

二、任务概述

1. 目标功能

本公共交通乘客行程模拟系统可以实现同类app、网站的大多基础功能，在欢迎界面进行登陆类型的选择或者注册。

1. 用户进入后先进行注册或登录，登陆后进行身份证信息与电话号码的绑定，绑定完成后可以进行个人健康状况、身份信息等个人信息的查看，可以进行乘车前的登记，查询个人车次并查询行程及途径站点，还可以对个人的乘车记录进行查询，搜索自己乘坐过的列车及到达过的城市。
2. 管理员登陆后可以进行对疫情状况的调查与管控，管理员可以模拟输入乘客信息进行对乘车人员身体健康状况的标记，还可以查询已经进行标记的阳性乘客，查询阳性乘客乘坐的列车，并查询与阳性乘客有相关接触的密接乘客。

2. 编写规范

1. 命名规范

函数的命名一律以小写英文为规范，文件与变量的命名实现英文释义与其功能的对应，数据结构等的命名以上述参考资料为标准进行规范化编程。

2. 注释

a. 函数功能在函数原型后注明。
b. 难以理解的算法和流程给出相应的注释。

3. 内存管理

a. 保证对指针的malloc和free一一对应
b. 保证文件的fopen和fclose一一对应

4. 页面控制函数编写规范

三、运行环境和配置

1. 硬件接口

处理器：Intel Pentium 166 MX 或以上

硬盘：空间150MB以上。

屏幕适配器：VGA接口。

系统运行内存：要求32MB以上。

2. 软件接口

开发软件工具：Borland C++ 3.1。

文字编辑工具：Notepad++、Borland C++ 3.1。

数据库：文本存储（记事本）。

操作系统：DOS WINDOWS 9X/ME/2000/XP/WINDOWS 10等。

3. 控制

该系统通过鼠标与键盘直接进行控制。用户将鼠标移至需要操作的功能区进行点击，同时通过键盘来完成登陆、注册、搜索、填写信息等输入功能。点击返回箭头即可返回上一页面。通过中断技术来获取鼠标的位置与键盘的输入功能。

四、需求分析

公共交通乘客行程模拟系统主要为了实现出行记录查询功能，出行登记功能，乘客健康状况登记功能，各种信息查询和浏览功能。具体包括：

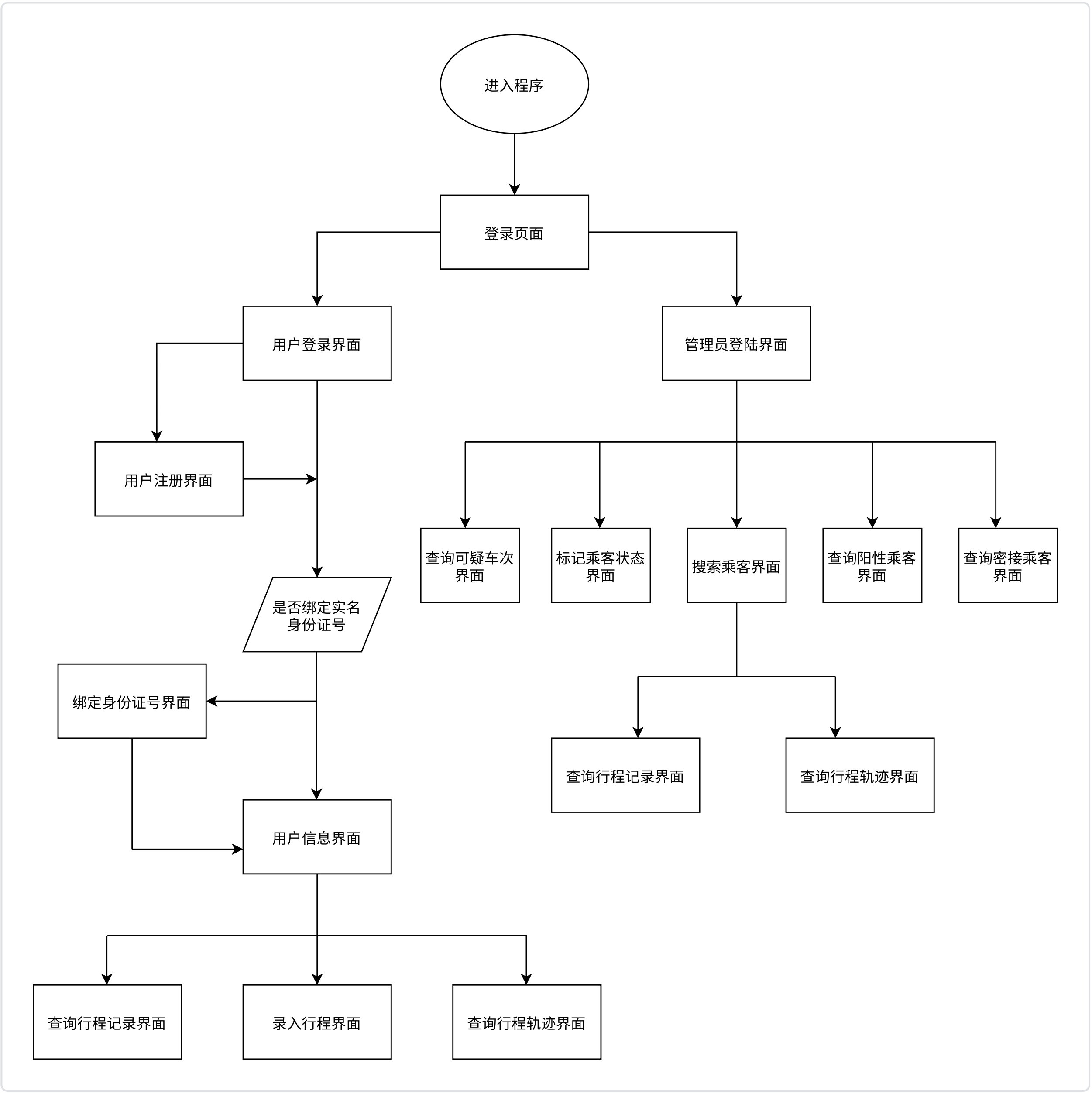
1. 为保证疫情期间乘客的个人健康安全，每个乘客都可以查看乘客个人信息，查询个人健康状况的更新，公共交通现面临严峻的防疫安全问题，乘客需在出行前确认自身身体健康状况以免对公众安全带来风险，以保证每一位使用公共交通的乘客的安全问题，健康状况处于非正常的乘客将被暂时拒绝使用公共交通设施以免为广大乘客带来传染风险。
2. 为保证每一位使用公共交通出行的乘客的健康状况，确认使用公共交通进行出行的乘客的安全状况，乘客需要在上车前进行乘车登记。乘客应提前了解个人乘坐列车的起点站终点站与途径站点并了解列车出行日期，乘客希望避免出行过程出现错误或者面临与感染者同乘而被感染的风险，因此乘客需提前确认好个人所乘列车的乘车时间和列车班次，并提前确认起点、目的地与途径站点。确认完成后凭借登记乘车凭证才可登上列车，未进行乘车登记或者乘车登记未通过的乘客都无法登车。用户希望对个人乘车记录进行查看以确认自身是否同感染者有过时空交集以对自身健康状况进行确认。同时，用户通过查看列车途径路线以确定行车途中乘客经过的地区方便进一步对自身健康状况进行确认。
3. 管理员需要对乘客的信息进行检查和管理时，通过管理员权限对乘客的身体健康状况进行录入，同时为了管理员更方便的对患病的乘客进行管理，我们设计了查询查询全体阳性乘客的功能，以便于政府机构、医院等及时搜索并获取信息对阳性感染者进行控制与隔离，将疫情期间的防疫措施落实到位。同时，如果能查询阳性患者乘坐过的车辆，将会进一步便利管理员与政府机关等对疫情防控的管理，所以我们加入了对可疑车次的查询，以便搜索乘坐过相同车次的乘客并对其健康状况进行管理与排查。并将与阳性乘客有相关接触的乘客采集，进行对密接乘客的管理与排查，方便进行下一步的疫情管控措施。

本系统需要完成的具体任务如下：

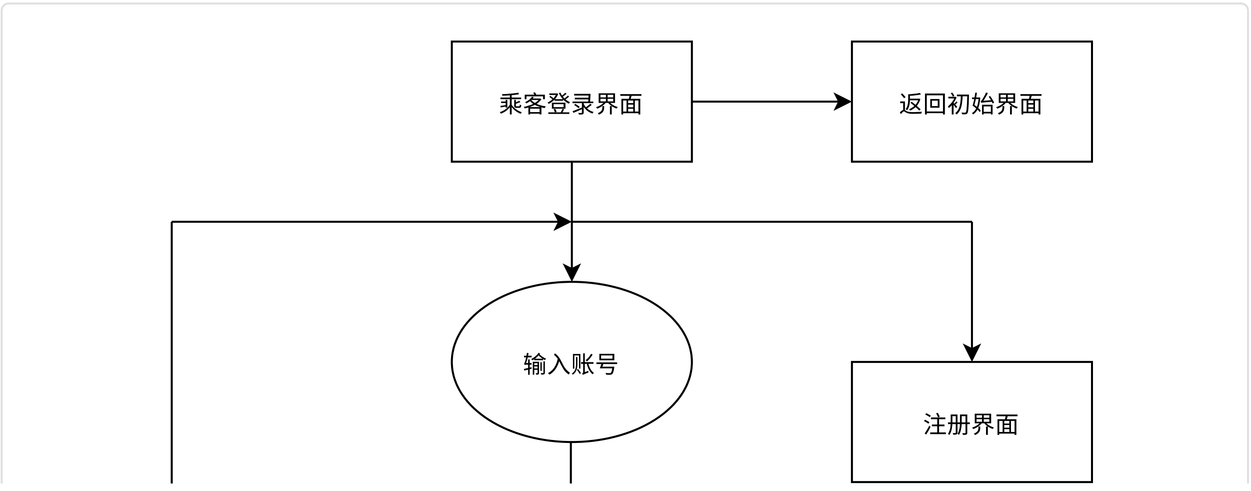
1. 登录注册。用户通过注册获得与身份证唯一绑定的账号，并可登录该账号进行操作。每个用户的用户信息、乘车信息等都和账号绑定。
2. 乘车登记功能。用户可在此界面选择称作班次，确认乘车时间及路线，在登车前完成登记,用户输入列车车次后可以选择起点站与终点站，将乘客起点站与终点站录入已完成乘车前的乘车具体登记。
3. 乘车记录查询功能。用户可在此界面查询个人乘坐的车次及路径，系统将展示乘客乘坐车次及乘坐车次的起点站、终点站。
4. 轨迹查询功能。用户可以在此界面查看个人乘坐列车的途径路线，并查看路线上的途径站点。
5. 管理员录入功能。管理员输入乘客身份证号确认信息后选择乘客健康状况对乘客的身体家康状况录入系统，方便乘客及管理员等进行查看。
6. 阳性乘客查询功能。根据已录入的信息和乘客的健康状况，对列车阳性乘客进行统一查询，其中可查看乘客的详细信息如感染症状、姓名、身份证号、电话、性别等与个人信息绑定的重要信息。
7. 可疑列车查询功能。系统判定阳性患者所乘坐过的列车为可疑列车，系统通过查询阳性乘客的乘车记录以确定可能带有病毒的列车。查询的结果包括列车名称，列车路线及铁路干线途经城市。
8. 密接乘客查询功能。管理员在此界面查询与阳性患者乘坐同一列车的乘客的个人身份信息，包括乘客的身份证号，电话，姓名等个人信息。
9. 搜索功能。

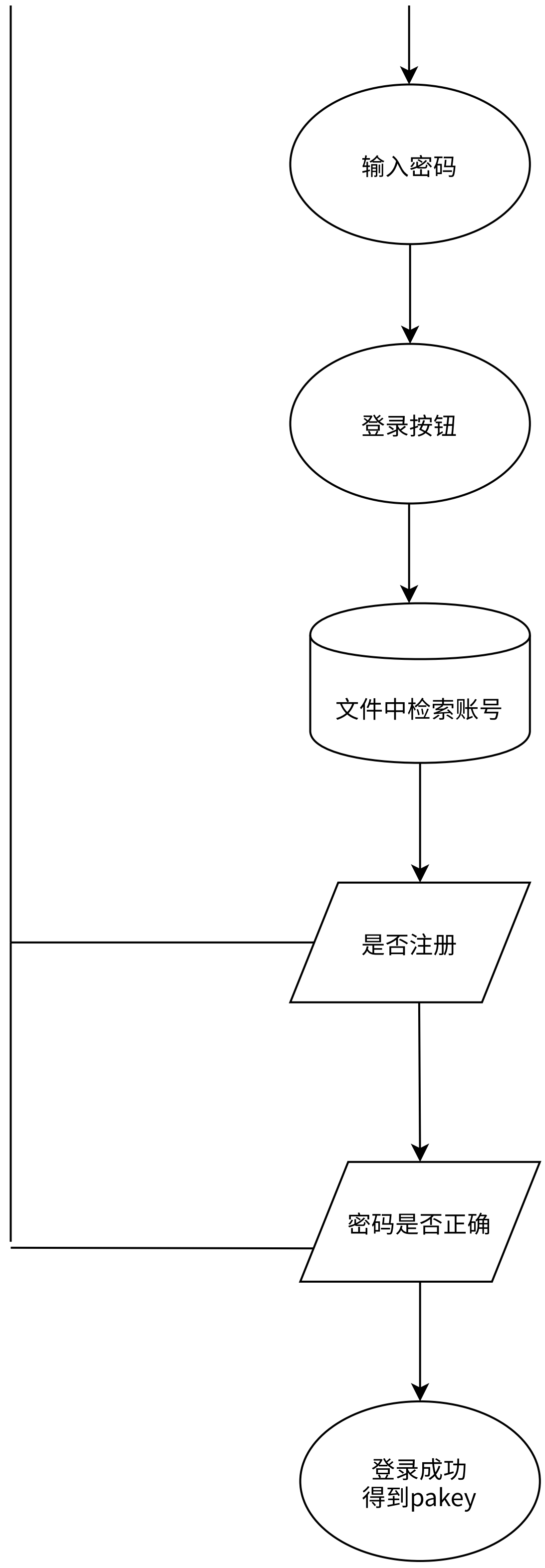
五、系统设计

1. 主要框架



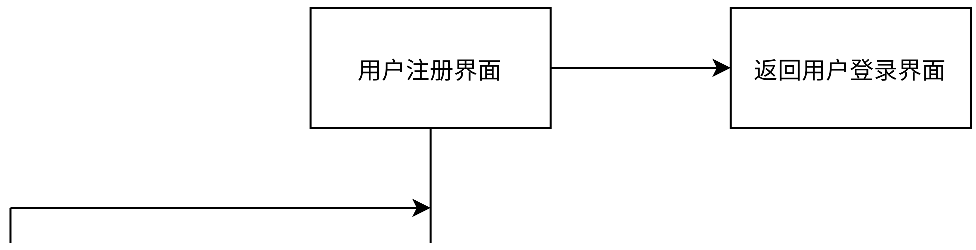
2. 用户登录



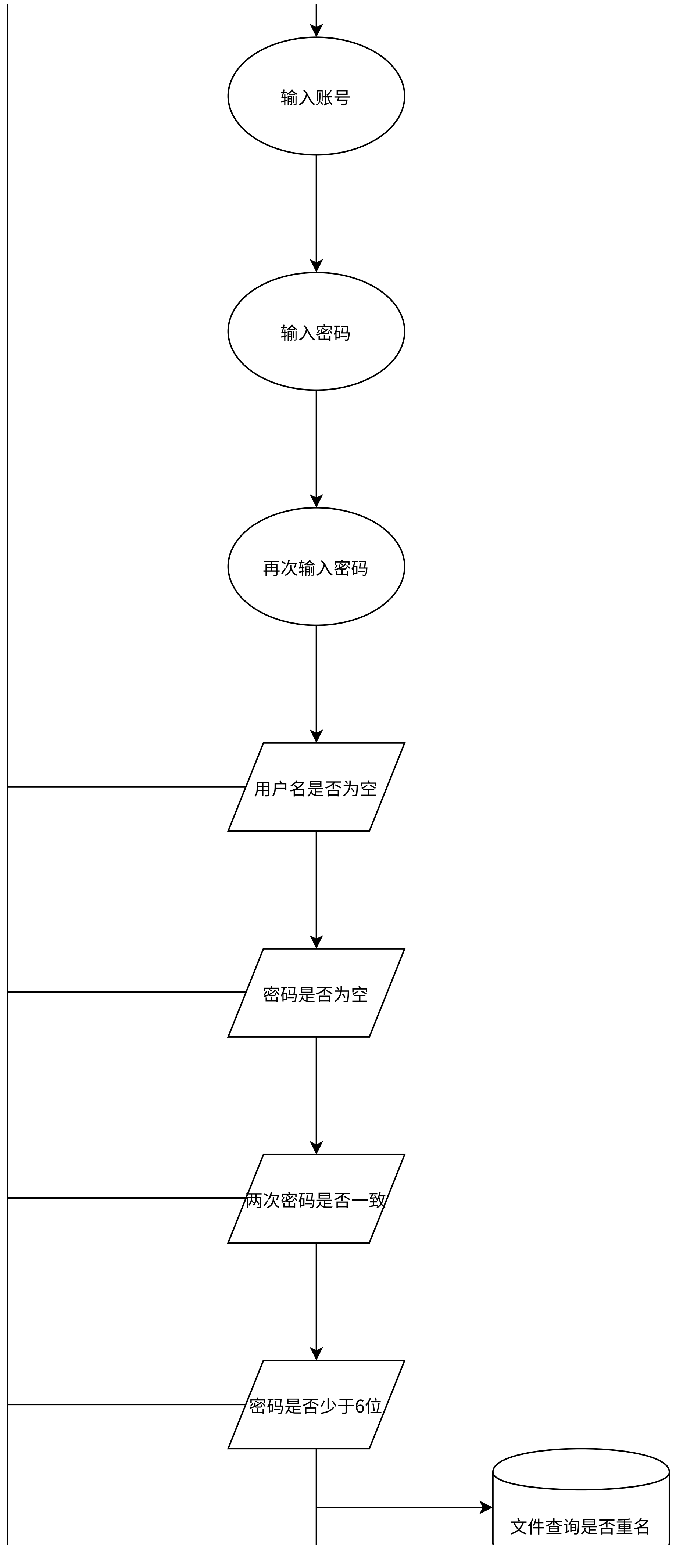


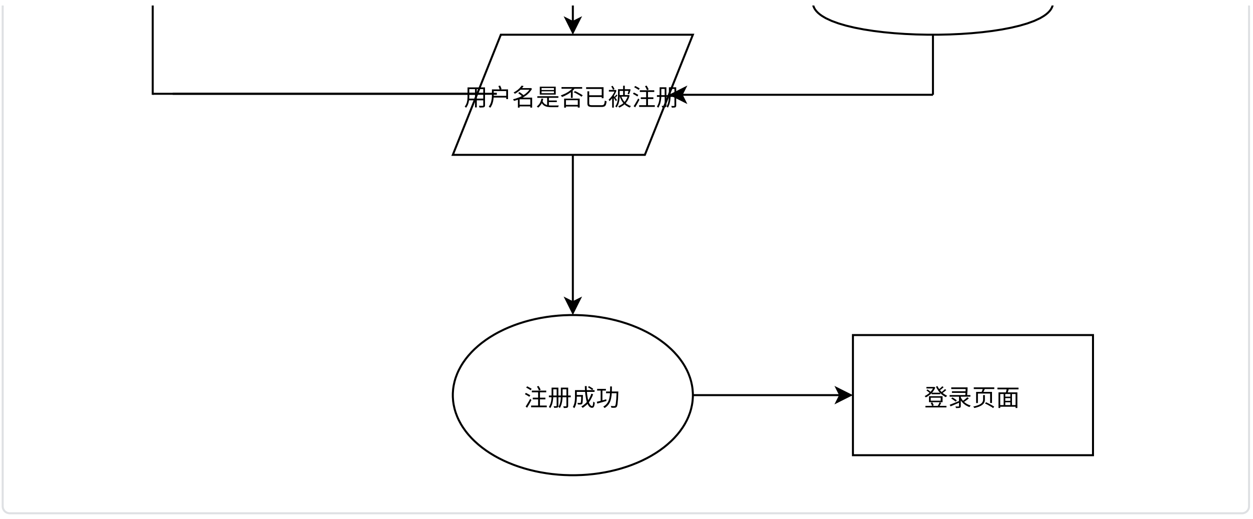
输入密码
登录按钮
文件中检索账号
是否注册
密码是否正确
登录成功
得到pakey

3. 用户注册



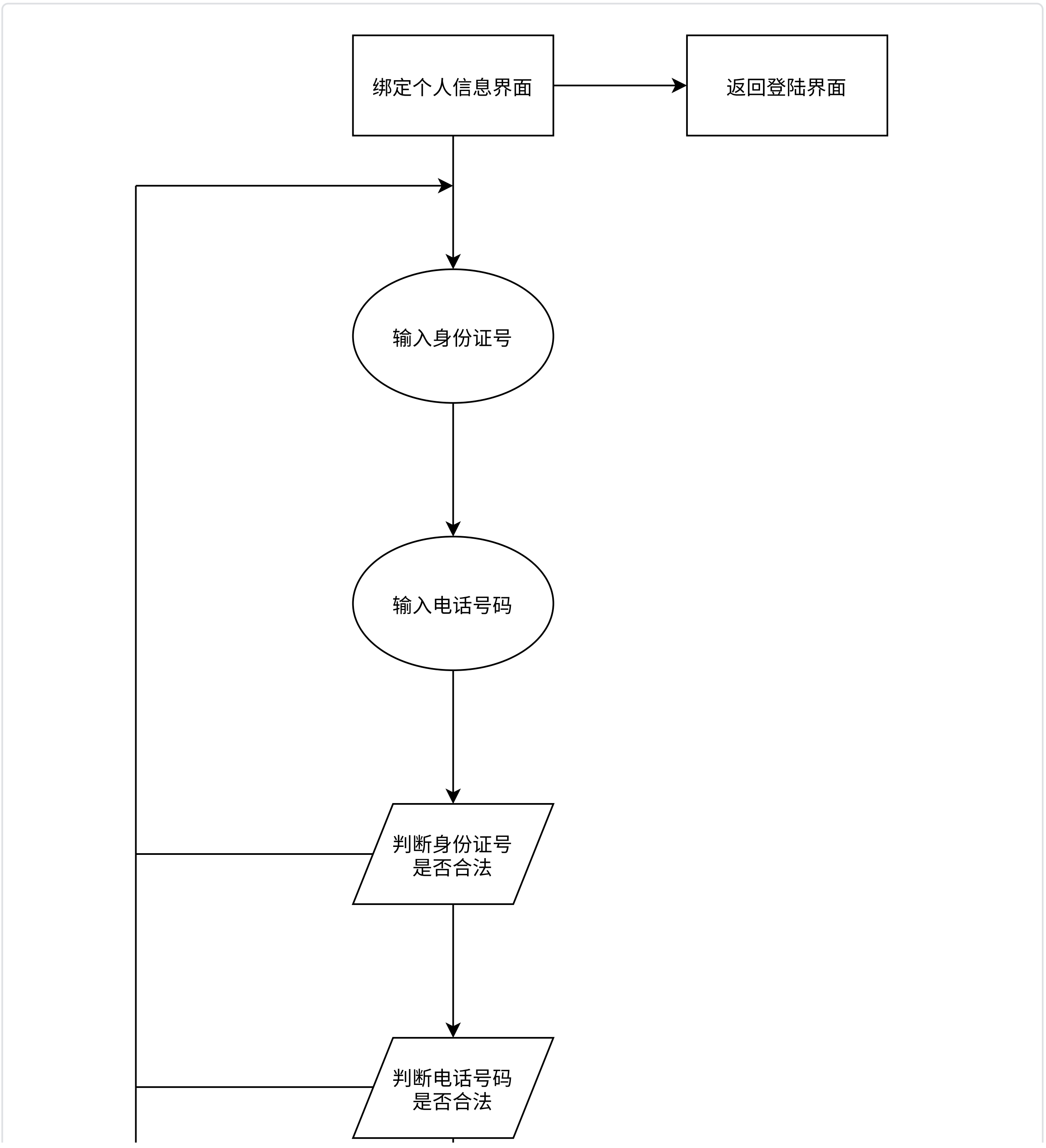
用户注册界面
返回用户登录界面


输入账号
输入密码
再次输入密码
用户名是否为空
密码是否为空
两次密码是否一致
密码是否少于6位
文件查询是否重名

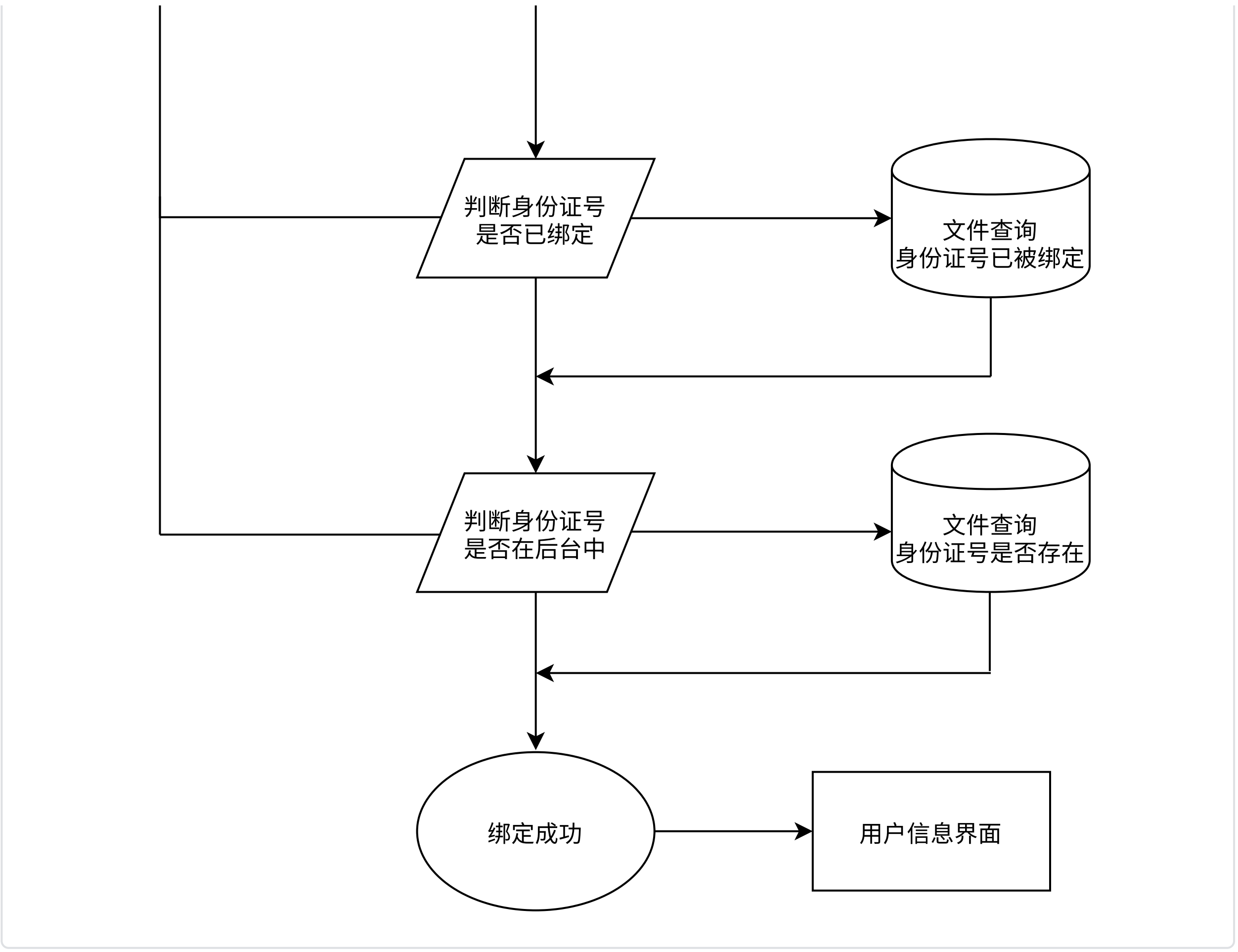


用户名是否已被注册
注册成功
登录页面

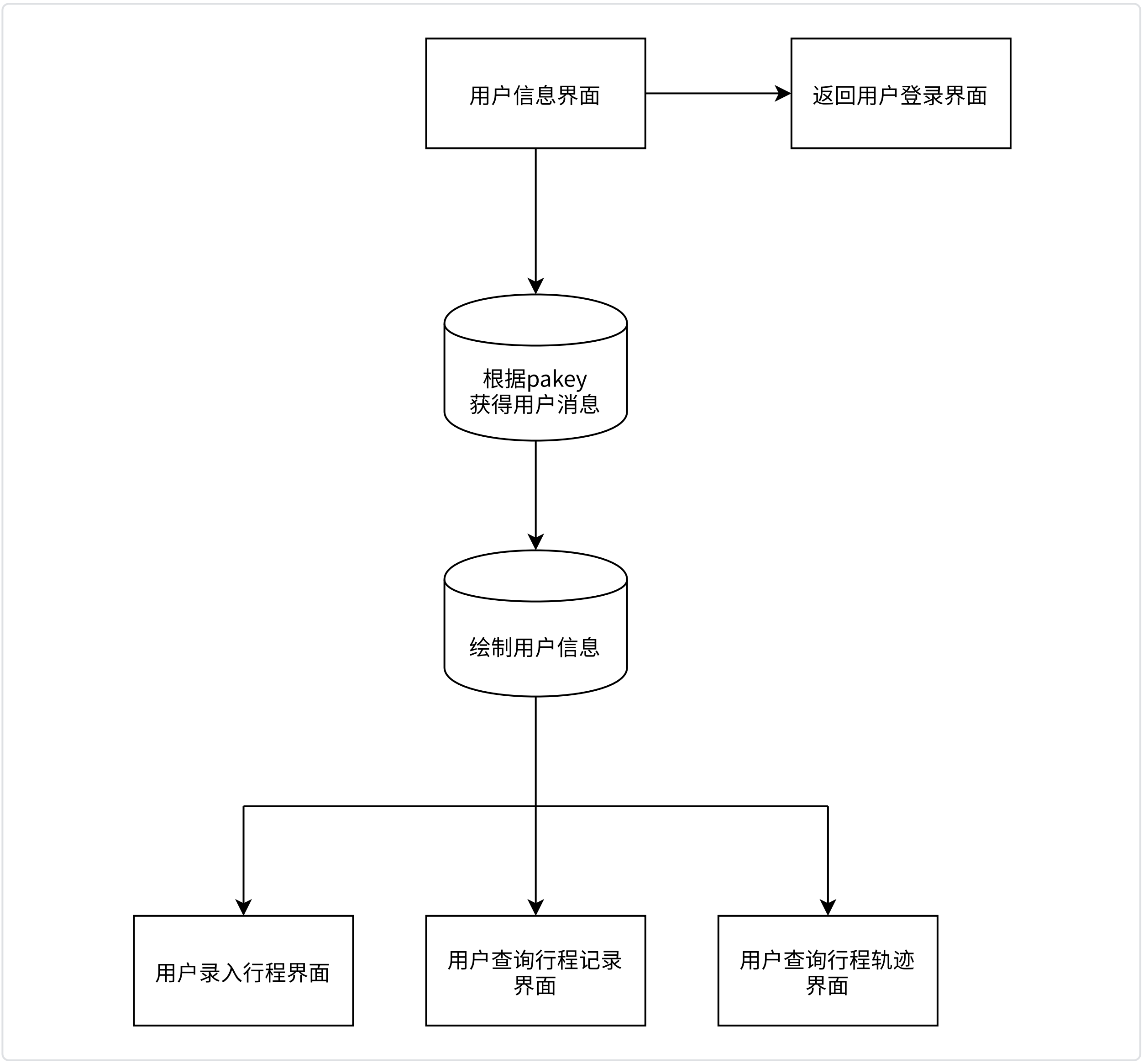
4. 用户绑定身份信息



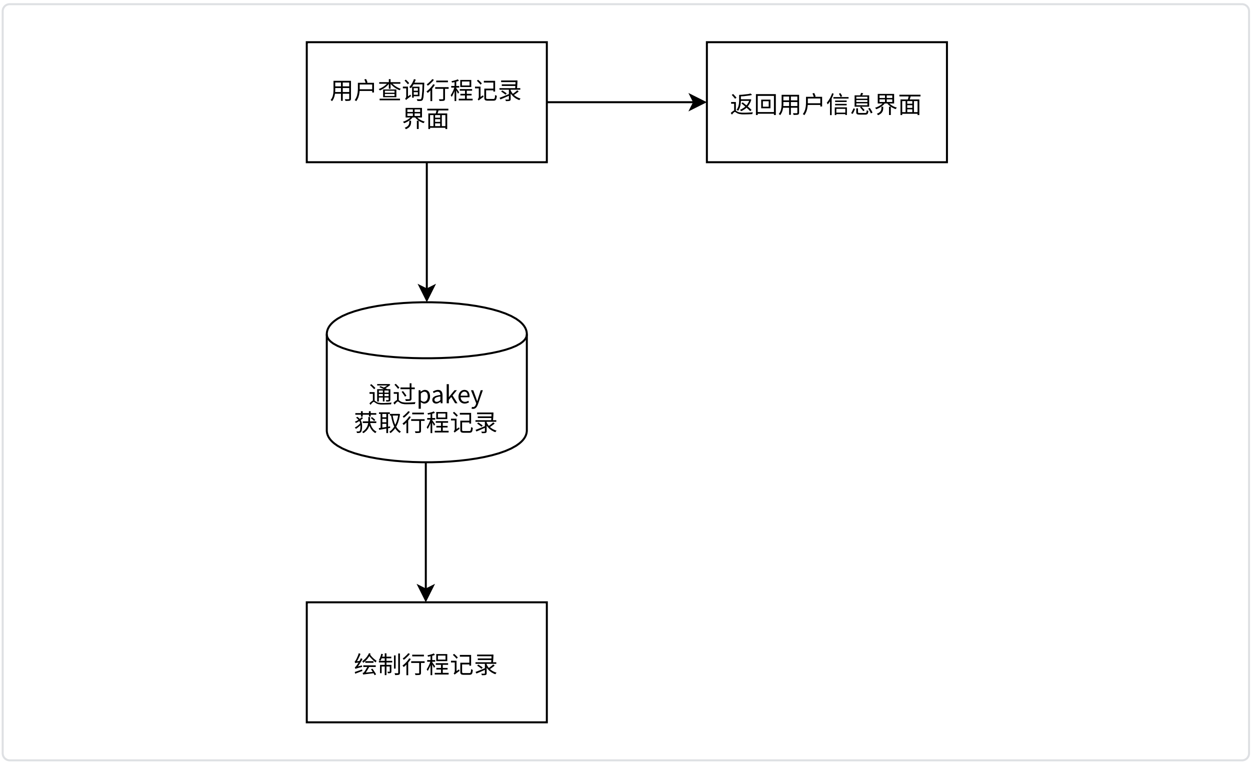
绑定个人信息界面
返回登陆界面
输入身份证号
输入电话号码
判断身份证号是否合法
判断电话号码是否合法


判断身份证号
是否已绑定
文件查询
身份证号已被绑定
判断身份证号
是否在后台中
文件查询
身份证号是否存在
绑定成功
用户信息界面

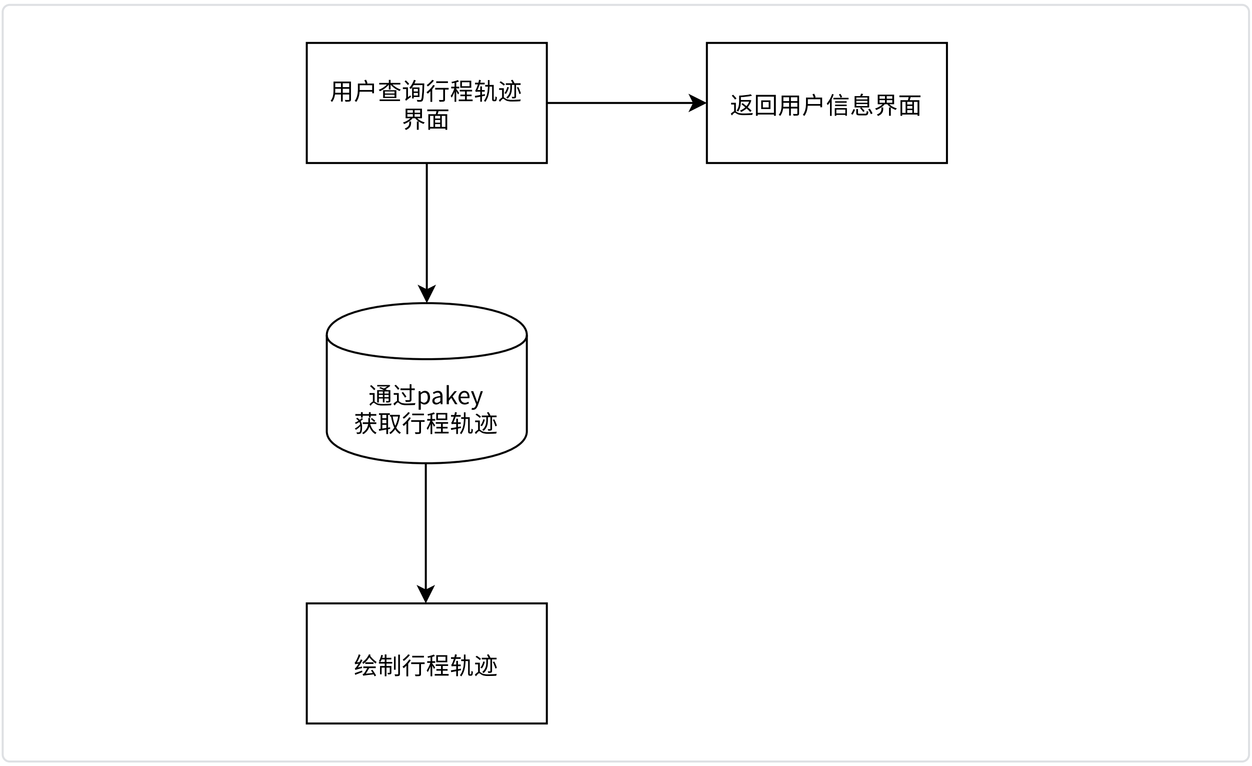
5. 用户信息界面


用户信息界面
返回用户登录界面
根据pakey获得用户消息
绘制用户信息
用户录入行程界面
用户查询行程记录界面
用户查询行程轨迹界面

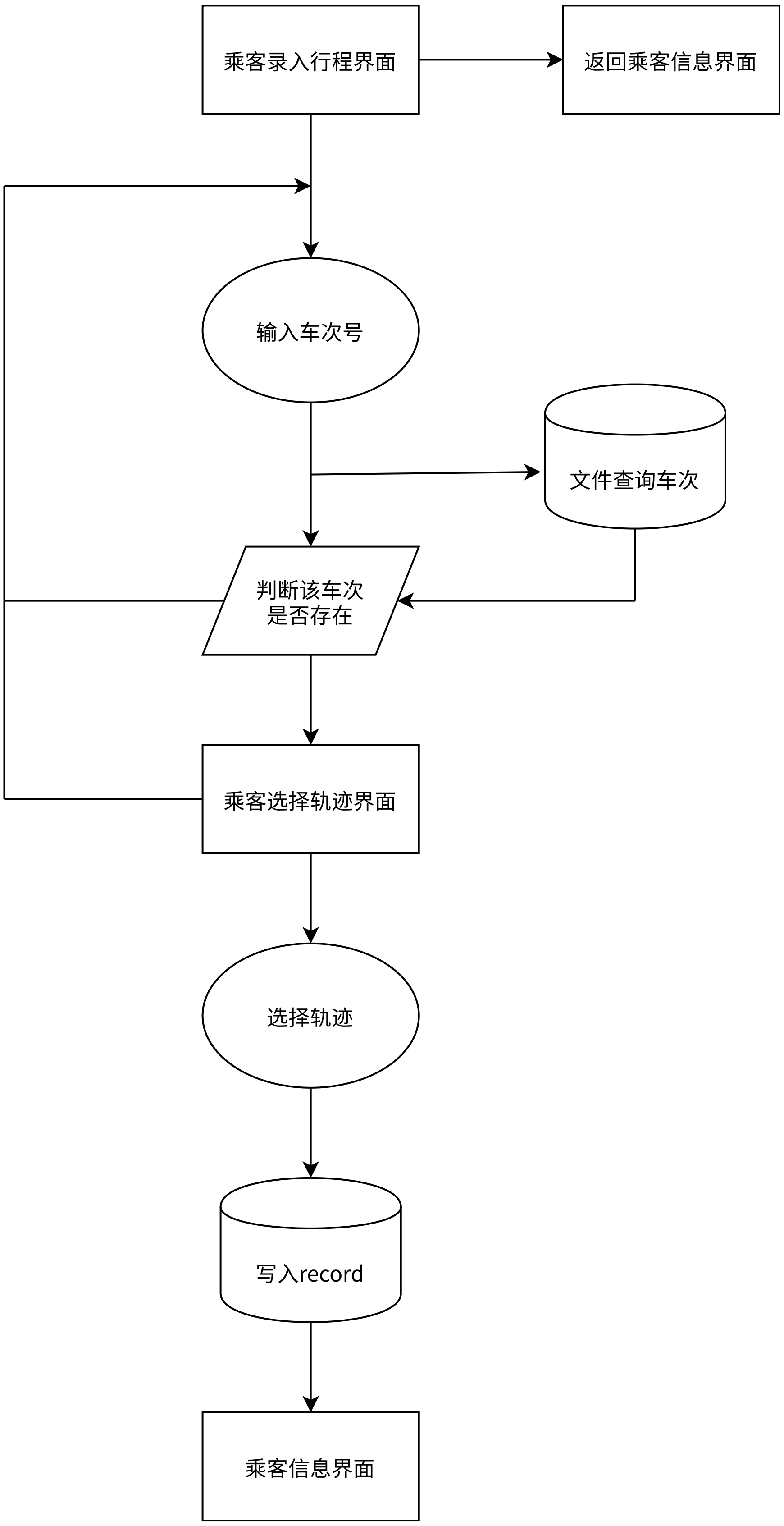
6. 用户查询行程记录



7. 用户查询行程轨迹



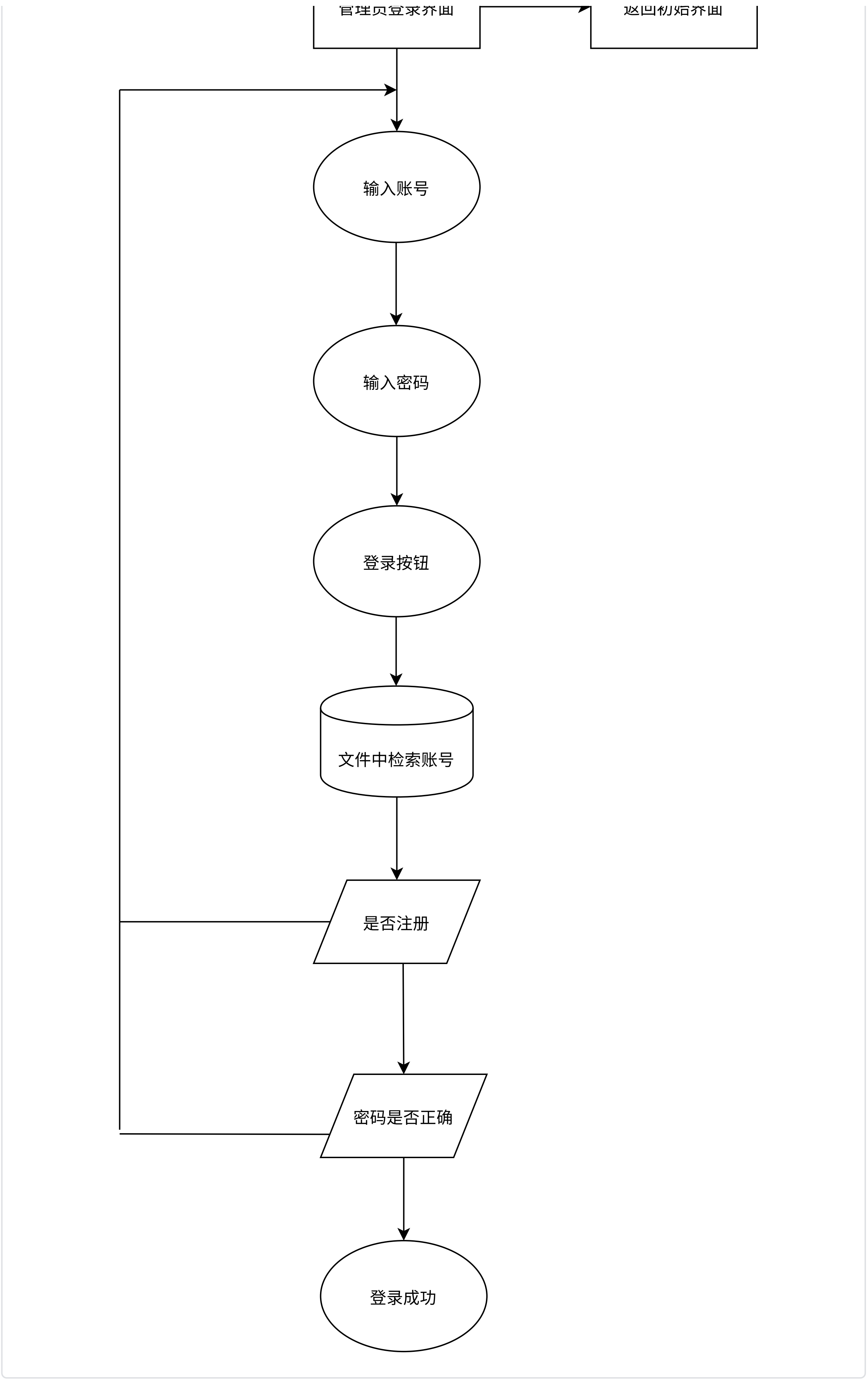
8. 用户录入行程



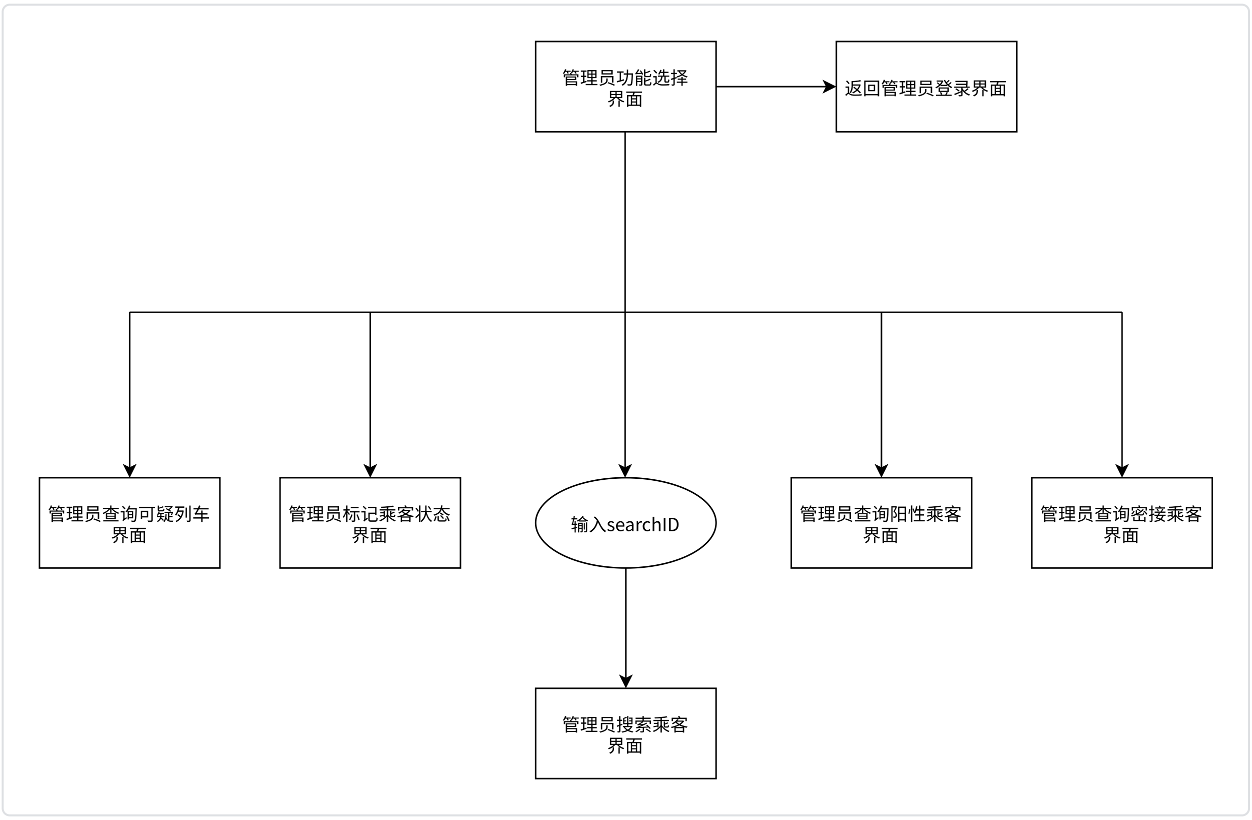
乘客录入行程界面
返回乘客信息界面
输入车次号
文件查询车次
判断该车次是否存在
乘客选择轨迹界面
选择轨迹
写入record
乘客信息界面

9. 管理员登录

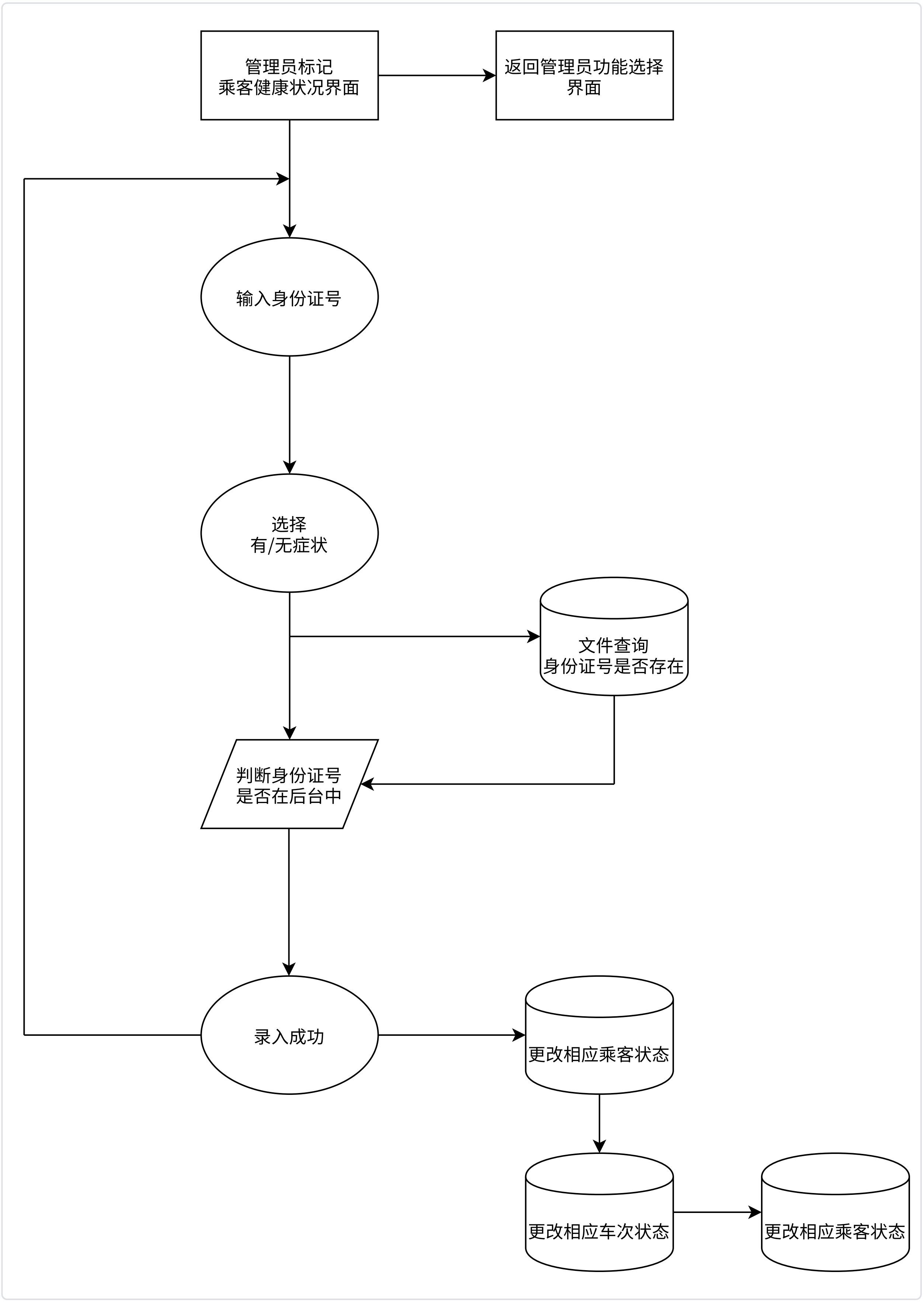
管理员登录界面
返回初始界面


管理员登录界面
返回初始界面
输入账号
输入密码
登录按钮
文件中检索账号
是否注册
密码是否正确
登录成功

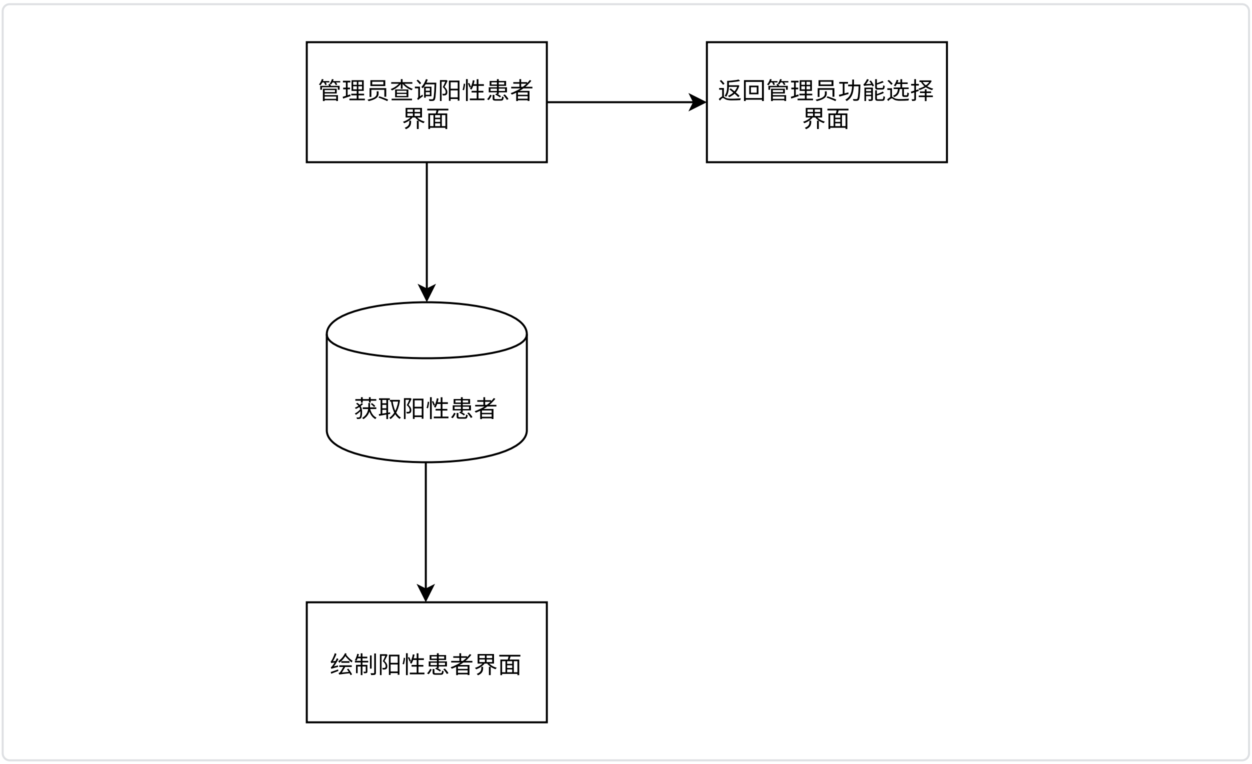
10. 管理员功能选择界面



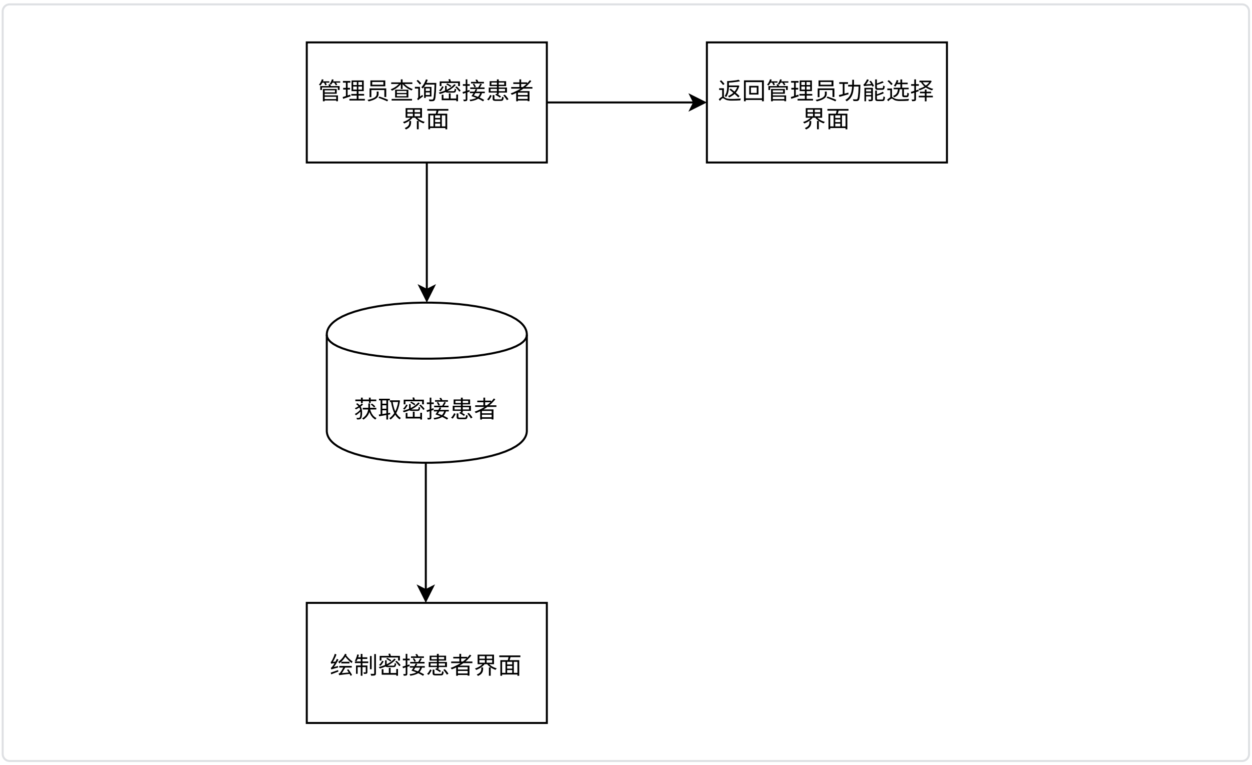
11. 管理员标记乘客健康状况



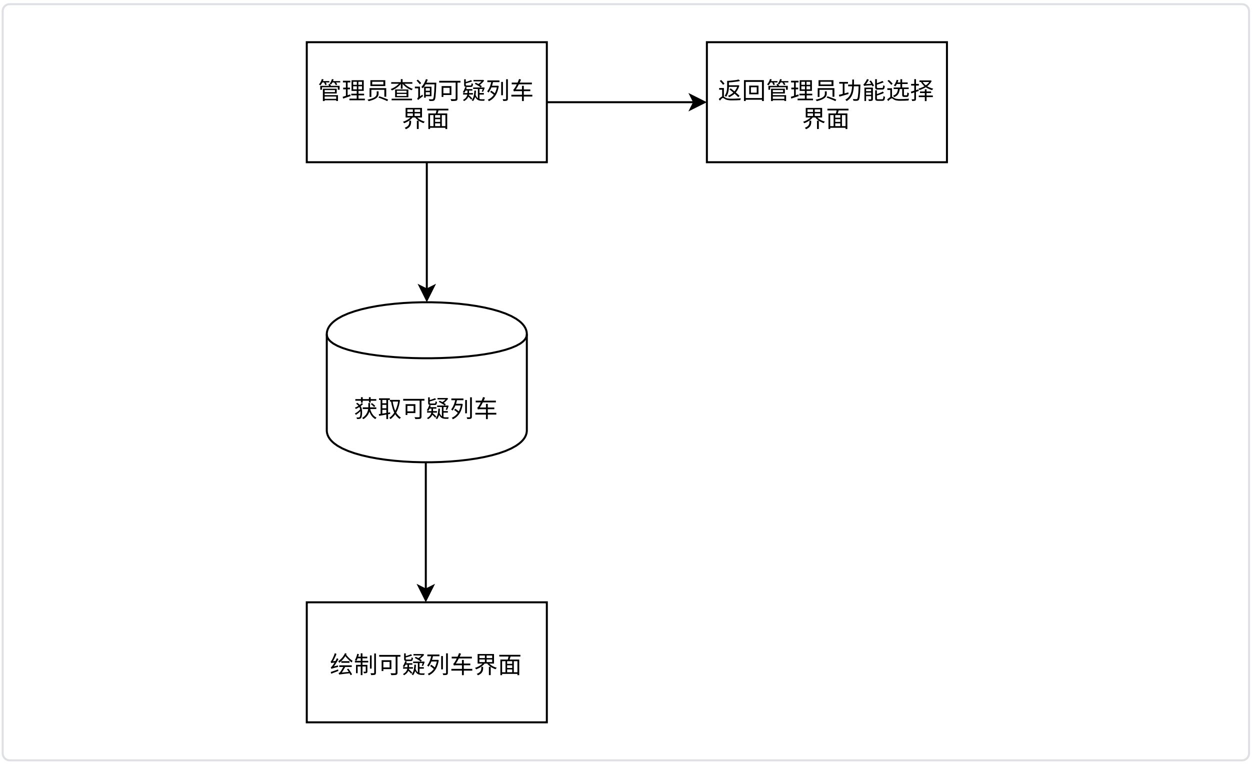
12. 管理员查询阳性患者



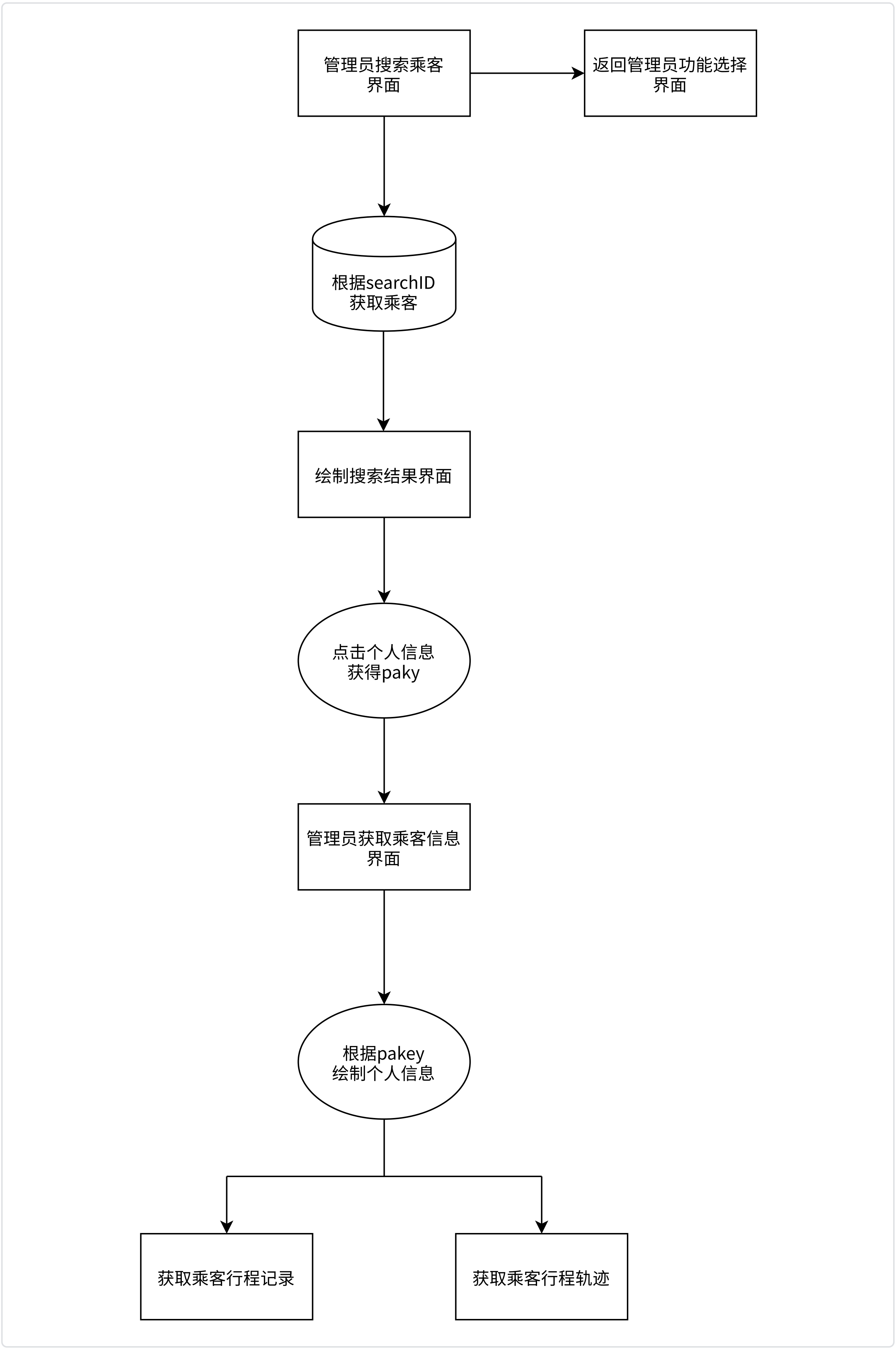
13. 管理员查询密接患者



14. 管理员查询可疑列车



15. 管理员搜索乘客


管理员搜索乘客界面
返回管理员功能选择界面
根据searchID获取乘客
绘制搜索结果界面
点击个人信息获得paky
管理员获取乘客信息界面
根据pakey绘制个人信息
获取乘客行程记录
获取乘客行程轨迹

六、界面设计

七、数据结构设计

本程序共定义了6种结构体，声明于data.h中

```C
#ifndef _DATA_H_
#define _DATA_H_

typedef struct administrator
{
    int adkey;
    char name[20];
    char password[20];
}ADMINISTRATOR;

typedef struct user
{
    int pakey;
    char name[50];
    char password[50];
}USER;

typedef struct passenger{
    int pakey;
    char ID[20];
    char tel[20];
    int sex;                // 1 男 2 女
    int age;
    int status;                //初始化为0，0为健康，1为有症状感染，2为无症状感染，3为密接
}PASSENGER;

typedef struct record
{
    int pakey;
    int trkey;
    char track[6];
}RECORD;

typedef struct train
{
    int trkey;
    char trainname[10];
    int date;
```

```c
    int date;
    int status;
}TRAIN;

typedef struct pagetrain{
    int trkey;
    char trainname[10];
    int date;
    int status;
}PAGETRAIN;

#endif
```

1. administrator

变量名	类型	解释说明
adkey	int	存储管理员的键
name	char[]	存储管理员用户名
password	char[]	存储管理员密码

2. user

变量名	类型	解释说明
pakey	int	存储乘客的键
name	char[]	存储乘客用户名
password	char[]	存储乘客密码

3. passenger

变量名	类型	解释说明
pakey	int	存储乘客的键
ID	char[]	存储乘客身份证号
tel	char[]	存储乘客电话号码

sex	int	存储乘客性别 1为男性，2为女性
age	int	存储乘客年龄
status	int	存储乘客健康状态 0为健康，1为有症状 2为无症状，3为密接

4. record

变量名	类型	解释说明
pakey	int	存储对应乘客的键
trkey	int	存储对应车次的键
track	char[]	存储详细行程轨迹

5. train

变量名	类型	解释说明
trkey	int	存储车次的键
trainname	char[]	存储车次名称
date	int	存储日期，date为距2020.1.1的天数
status	int	存储车次状态，0为正常，1为异常

6. pagetrain

变量名	类型	解释说明
trkey	int	存储车次的键
trainname	char[]	存储车次名称
date	int	存储日期，date为距2020.1.1的天

		数
status	int	存储车次状态，0为正常，1为异常

八、算法说明

1. 分页器功能

2. 基于动态规划获取最小相同字串长度的搜索功能

九、函数原型

1. config.h

C

```c
#ifndef _config_h_
#define _config_h_

int judgeID(char *ID);
int judgetel(char *tel);
int leapyear(int yy);
int getdate(int date,int *yy,int *mm,int *dd);
int postdate(int yy,int mm,int dd,int *date);
char *datestring(int date);
int getsexbyID(char ID[20]);
int getagebyID(char ID[20]);
void gettodaydate(int *yy,int *mm,int *dd);
int xpos(char c);
int ypos(char c);
void gettrackstring(char *trainname,int i,char *desttrack);
int samestringmax(char *a,char *b);

#endif
```

2. control.h

C

```c
#ifndef _control_h_
#define _control_h_
```

```c

int inbarword16(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
int pressbarword16(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);

int inbarword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
int pressbarword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);

int inbarword32(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
int pressbarword32(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);

int inbarword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
int pressbarword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);

int inbarwordframe(int left,int top,int barcolor,int width,char *s,int wordcolor,int framecolor);
int pressbarwordframe(int left,int top,int barcolor,int width,char *s,int wordcolor,int framecolor);

int incircle(int x,int y,int radius);
int presscircle(int x,int y,int radius);

int inreturnbutton(void);
int pressreturnbutton(void);

int insearchbuttuon(void);
int presssearchbutton(void);

int inpalabelu(void);
int presspalabelu(void);

int inpalabelm(void);
int presspalabelm(void);

int inpalabell(void);
int presspalabell(void);

int inleftarrow(void);
int pressleftarrow(void);

int inrightarrow(void);
int pressrightarrow(void);

int intrack1(void);
int presstrack1(void);

int intrack2(void);
int presstrack2(void);

int intrack3(void);
int presstrack3(void);

int intrack4(void);
int presstrack4(void);

int intrack5(void);
int presstrack5(void);

int intrack6(void);
int presstrack6(void);

int intrack7(void);
int presstrack7(void);

int intrack8(void);
int presstrack8(void);

int intrack9(void);
int presstrack9(void);

int intrack10(void);
int presstrack10(void);

#endif
```

3. drawpage.h

```c
#ifndef _DRAWPAGE_H_
#define _DRAWPAGE_H_

void drawstart();
void drawadlogin();
void drawpalogin();
void drawparegister();
void drawpabind();
void drawpamessage(struct passenger *pa);
void drawpapostrecord();
void drawpagetrecord(struct pagetrain *ptrnu,struct pagetrain *ptrnm,struct pagetrain *ptrnl,int currentpage,int countpage);
void drawpagettrack();
void drawstationselect(char *trainname);
void drawadlabel();
void drawadmanager();
void drawadhealth();
void drawadgetpospa(struct passenger *pau,struct passenger *pam,struct passenger *pal,int currentpage,int countpage);
void drawadgetcttpa(struct passenger *pau,struct passenger *pam,struct passenger *pal,int currentpage,int countpage);
void drawadgetpostr(struct pagetrain *ptrnu,struct pagetrain *ptrnm,struct pagetrain *ptrnl,int currentpage,int countpage);
void drawadsearch(struct passenger *pau,struct passenger *pam,struct passenger *pal,int currentpage,int countpage);

#endif
```

4. file.h

```c
#ifndef _file_h_
#define _file_h_

void initfile(void);
int writeuser(char *name,char *password);
int loginuser(char *name,char *password);
int existusername(char *name);
int writeadmin(char *name,char *password);
int loginadmin(char *name,char *password);
int writepassenger(int pakey,char *ID,char *tel,int sex,int age,int status);
int existpassengerID(char *ID);
int getpassengerbystatus(int status,struct passenger destpa[50]);
int getpassengerbysearch(char *searchID,struct passenger destpa[20]);
int getpassengerbypakey(int pakey,struct passenger *destpa);
int getpakeybyID(char *ID,int *pakey);
int updatestatusbypakey(int pakey,int status);
int writetrain(char *trainname,int yy,int mm,int dd);
int gettrainbytrkey(int trkey,struct train *desttrn);
int gettrainbystatus(int status,struct pagetrain desttrn[20]);
int updatestatusbytrkey(int trkey,int status);
int getpostedtrkey(int date,char *trainname,int *trkey);

#endif
```

5. file2.h

```c
#ifndef _file2_h_
#define _file2_h_

int writerecord(int pakey,int trkey);
int writerecordv1(int pakey,int trkey,char *track);
int getrecordbypakey(int pakey,int desttrkeyset[50]);
int getrecordbytrkey(int trkey,int destpakeyset[50]);
int gettrackbypakey(int pakey,char trackset[10][10]);

#endif
```

6. filetxt.h

C++

```cpp
#ifndef _filetxt_h_
#define _filetxt_h_

int getcnamebyID(char *ID,char *cname);
int getstartbytrainname(char *trainname,char *start);
int getendbytrainname(char *trainname,char *end);

#endif
```

7. graphpro.h

C

```c
#ifndef _GRAPHPRO_H_
#define _GRAPHPRO_H_

void drawtop(void);
void barword16(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
void barword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
void barword32(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
void barword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor);
void barwordframe(int left,int top,int barcolor,int n,char *s,int wordcolor,int framecolor);
void returnbutton(int color);
void searchbutton(int color);
void draw_date(int x, int y,int wordcolor);
void loginok(void);
void loginfail(void);
void loginpasswordwrong(void);
void loginnoregister(void);
void registerok(void);
void registerrepeat(void);
void registerpasswrong(void);
void registerpasszero(void);
void registernamezero(void);
void registerpassshort(void);
void bindok(void);
void bindIDrepeat(void);
void bindIDwrong(void);
void bindtelwrong(void);
void labelok(void);
void labelIDzero(void);
void statusabnormal(void);
void recordzero(void);
void trackrecordrepeat(void);
void trackrecordok(void);
void palabelu(struct passenger *pa);
void palabelm(struct passenger *pa);
void palabell(struct passenger *pa);
void trlabelu(struct pagetrain *trn);
void trlabelm(struct pagetrain *trn);
void trlabell(struct pagetrain *trn);
void leftarrow(int color);
void rightarrow(int color);
void pagenumber(int currentpage,int countpage);
void putnum(int x,int y,int num);
void todaydate(void);
void recordmessage(char* trainname);
void position1(int x,int y,int *s);
void position2(int x,int y,int *s);
void drawmap(void);
void mapline(char *s,int color);
void track1(char *trainname,int color);
void track2(char *trainname,int color);
void track3(char *trainname,int color);
void track4(char *trainname,int color);
void track5(char *trainname,int color);
void track6(char *trainname,int color);
void track7(char *trainname,int color);
void track8(char *trainname,int color);
void track9(char *trainname,int color);
void track10(char *trainname,int color);

#endif
```

8. page2.h

C++

```
#ifndef _page2_h_
#define _page2_h_

void pagestart(int *page);
void pagepalogin(int *page,int *pakey);
void pageadlogin(int *page);

#endif
```

9. pagead.h

C++

```
#ifndef _pagead_h_
#define _pagead_h_

void pageadmanager(int *page,char searchID[20]);
void pageadlabel(int *page);
void pageadgetpospa(int *page);
void pageadgetcttpa(int *page);
void pageadgetpostrn(int *page);
void pageadsearch(int *page,char searchID[20],int *pakey);
void pageadpamessage(int *page,int *pakey);
void pageadpagettrack(int *page,int *pakey);
void pageadpagetrecord(int *page,int *pakey);

#endif
```

10. pagepa.h

```C
#ifndef _pagepa_h_
#define _pagepa_h_

void pageparegister(int *page);
void pagepabind(int *page,int *pakey);
void pagepamessage(int *page,int *pakey);
void pagepapostrecord(int *page,int *pakey,int *trkey);
void pagepagetrecord(int *page,int *pakey);
void pageposttrack(int *page,int *pakey,int *trkey);
void pagepagettrack(int *page,int *pakey);

#endif
```

11. public.h

```C
#ifndef _public_h_
#define _public_h_

#include<graphics.h>
#include<time.h>
#include<bios.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include "hz.h"
#include "graphpro.h"
#include "mouse.h"
#include "control.h"
#include "drawpage.h"
#include "input.h"
#include "page2.h"
#include "pagead.h"
#include "pagepa.h"
#include "data.h"
#include "file.h"
#include "file2.h"
#include "filetxt.h"
#include "config.h"

#endif
```

12. hz.h

C++

```cpp
#ifndef __HZ_H__
#define __HZ_H__

void puthz(int x, int y,char *s,int flag,int part,int color);

#endif
```

13. input.h

C

```c
#ifndef _input_h_
#define _input_h_

void Input_Vis(char* ip,int x,int y,int lim,int color,int wordcolor);
void Input_Invis(char* ip,int x,int y,int lim,int color,int wordcolor);


#endif
```

14. mouse.h

```c
#ifndef _mouse_h_
#define _mouse_h_

int mouse_press(int x1, int y1, int x2, int y2);//如果在框中点击，则返回1；在框中未点击，则返回2；不在框中则返回0
void mouse(int,int);//设计鼠标
void mouseinit(void);//初始化
//void mou_pos(int*,int*,int*);//更改鼠标位置
void mread(int *,int *,int*);//改坐标不画
void save_bk_mou(int x,int y);//存鼠标背景
void clrmous(int x,int y);//清除鼠标
void drawmous(int x,int y);//画鼠标
void newmouse(int *nx,int *ny,int *nbuttons);    //更新鼠标

extern int MouseX;
extern int MouseY;
extern int MouseS;
extern int press;
extern union REGS regs;
#endif
```

十、感想体会

1. 李洋

日月如梭，白驹过隙，c语言课程设计的日子很快到来了。回想过去的几个月不禁让我们感慨万分，初开始面对c课设的迷惘无助仿佛还在昨天。从大一初入学校就开始听说关于c课设的种种传闻与故事，曾经只是轻松一笑，到我们亲自来完成这份任务时却才感受到了真正的压力与紧张感。初学c课设的我完全无法将课本与程序设计相结合，无从下手的我焦急万分、不知所措，好在许多学长学姐和我的队友为我提供了很多的帮助，我在慢慢摸索中一步步学习c语言程序设计。c课设是一项好大的工程，初次面对它让我有些喘不过气，但在一步步前进中，我跌跌撞撞的迈上了c语言程序设计这条艰巨的路。

在设计之初，我纠结与课本内容的讲解与学习，但在一步步摸索中，我试着在书写代码的过程中更好地领悟c课设的构架与逻辑。我和队友开始对c课设进行第一步的需求分析，在需求分析后，我们逐步确立了自己程序的设计方向与大体思路，我们又从网络上参考铁路12306等相关的app与网站，从更多方面考虑用户会想要使用一个包含什么样功能的系统来便利自己的生活。接着我参考了一些学长学姐的代码，对程序设计的框架有了一个更加明确直观的认识，从图形界面的绘制到界面跳转的完成，从按钮功能的实现到文件结构体的完善。在代码结构逐步完善的同时，我们的信心也不断增强。

同时我的队友帮助我开始使用git进行线上的代码传输，使得我们能随时随地共享我们更新的代码。c课设是一项浩大的工程，在c课设的完成过程中我更深的领悟到了团队合作的魅力与重要性，

在队友的帮助下我们共同努力共同进步，虽然时刻有压力伴在心中，但在鼓励与坚持中我们共同进步，不断地攻克难关，我在不断地激励自己不能拖对方后腿，同时有问题我们也会共同努力共同将其解决。

课设让我对问题的解决有了更深一步的了解。之前在学习c语言的过程中，代码出现问题后便通过系统提供的报错来对代码进行修改与完善，但在c语言课程设计的过程中，许多问题不会出现在error之中，这就需要我们通过调试来对bug进行修复与完善，刚开始的我对此一筹莫展，只能通过目力来寻找可能存在的错误与问题，但在逐渐地对调试的学习中，我渐渐学会了从问题出现打大概位置对代码进行调试以进行进一步的修改，尽管有时候会被一个不知名的bug卡住进度许久，但在bug解决运行成功的那一瞬间所来带的喜悦比设计程序带来的更为强烈。

c课设在为我们带来压力的同时也带来了动力，这是一项很考验我意志力与坚持力的艰巨任务，c课设在占用了大量时间的同时极为考验我们对学习时间与编写程序时间的平衡能力，尤其是在面临课设验收和微积分，物理期中考试同时到来的节点上，更为考验我们对时间的管理能力和我们对压力的处理能力。在巨大的压力之下，我内心也曾有过灰心与恐惧，但幸运的是，我的队友为我提供了巨大的帮助，在他的帮助下，我不断处理各方面压力，不断将其均衡。

这次c课设让我学到了很多，无论是需求分析，对bug的处理能力，还是团队合作完成任务的过程都让我受益匪浅。

2. 张子陆

马上就要迎来c课设的验收，而为一个bug苦苦皱眉一晚的日子仿佛还在昨天，c课设完成的过程中，有收获，有压力，有喜悦，也有痛苦，但我们还是坚持走到了最后一步。一路坚持走来的收获最多的也许不是代码的学习与理解，更大的收获是在克服困难路上的坚持与不断挑战自我的突破，c课设的完成让我收益颇多，让我在突破中成长了自我。

开始做c课设的时候遇到了许多困难，无法合理分配课业学习与c课设的时间安排，

十一、源代码

1. main.c

C

```c
#include"public.h"

int main(){
    int gd =VGA;
    int gm =VGAHI;
    char S[30];
    int page=0;
    FILE *log;
    int pakeymain;
    int trkeymain;
    int i;
    char searchID[20];
    struct train trainmain1;
    struct train trainmain2;
    int yy,mm,dd;
    int trkeyset[50];
    int pakeyset[50];
    char trackset[10][10];

    initfile();
    initgraph(&gd,&gm,".\\BGI");
    mouseinit();
    setcolor(BLUE);

    while (1)
    {
        switch (page)
        {
            case 0:
                pagestart(&page);
                break;
            case 1:
                pagepalogin(&page,&pakeymain);
                break;
            case 2:
                pageadlogin(&page);
                break;
            case 3:
                pageparegister(&page);
                break;
            case 4:
                pagepabind(&page,&pakeymain);
                break;
            case 5:
                pagepamessage(&page,&pakeymain);
                break;
            case 6:
                pagepapostrecord(&page,&pakeymain,&trkeymain);
                break;
            case 7:
                pagepagetrecord(&page,&pakeymain);
                break;
            case 8:
                pageposttrack(&page,&pakeymain,&trkeymain);
                break;
            case 9:
                pagepagettrack(&page,&pakeymain);
                break;
            case 21:
                pageadmanager(&page,searchID);
                break;
            case 211:
                pageadlabel(&page);
                break;
            case 212:
                pageadgetpospa(&page);
                break;
            case 213:
                pageadgetpostrn(&page);
                break;
            case 214:
                pageadgetcttpa(&page);
                break;
            case 215:
                pageadsearch(&page,searchID);
                break;
        }
    }
}
```

2. config.c

```C
int judgetel(char *tel){
    if(strlen(tel)!=11){
        return 0;
    }
    if(*tel!='1'){
        return 0;
    }
    return 1;
}

int leapyear(int yy){
    if((yy%4==0&&yy%100!=0)||(yy%400==0))return 1;
    else return 0;
}

int getdate(int date,int *yy,int *mm,int *dd){
    int year=2020;
    int month;
    int day;

    if(date<=0){
        return 0;
    }

    date++;
    while(date>365){
        if(leapyear(year)==1&&date>366){
            date-=366;
            year++;
        }
        else if(leapyear(year)!=1){
            date-=365;
            year++;
        }
    }
    if(leapyear(year)==1){
        if(date>=1&&date<=31){
            month=1;
            day=date-0;
        }
        if(date>=32&&date<=60){
            month=2;
            day=date-31;
        }
        if(date>=61&&date<=91){
            month=3;
            day=date-60;
        }
        if(date>=92&&date<=121){
            month=4;
            day=date-91;
        }
        if(date>=122&&date<=152){
            month=5;
            day=date-121;
        }
        if(date>=153&&date<=182){
            month=6;
            day=date-152;
        }
        if(date>=183&&date<=213){
            month=7;
            day=date-182;
        }
        if(date>=214&&date<=244){
            month=8;
            day=date-213;
        }
        if(date>=245&&date<=274){
            month=9;
            day=date-244;
        }
        if(date>=275&&date<=305){
            month=10;
            day=date-274;
        }
        if(date>=306&&date<=335){
            month=11;
            day=date-305;
        }
        if(date>=336&&date<=366){
            month=12;
            day=date-335;
        }
    }
    else{
        if(date>=1&&date<=31){
            month=1;
            day=date-0;
        }
        if(date>=32&&date<=59){
            month=2;
            day=date-31;
        }
        if(date>=60&&date<=90){
            month=3;
            day=date-59;
        }
        if(date>=91&&date<=120){
            month=4;
            day=date-90;
        }
        if(date>=121&&date<=151){
            month=5;
            day=date-120;
        }
        if(date>=152&&date<=181){
            month=6;
            day=date-151;
        }
        if(date>=182&&date<=212){
            month=7;
            month=7;
            day=date-181;
        }
        if(date>=213&&date<=243){
            month=8;
            day=date-212;
        }
        if(date>=244&&date<=273){
            month=9;
            day=date-243;
        }
        if(date>=274&&date<=304){
            month=10;
            day=date-273;
        }
        if(date>=305&&date<=334){
            month=11;
            day=date-304;
        }
        if(date>=335&&date<=365){
            month=12;
            day=date-334;
        }
    }
    *yy=year;
    *mm=month;
    *dd=day;
    return 1;
}

int postdate(int yy,int mm,int dd,int *date){
    if(yy>=2028||yy<=2019){
        *date=-1;
        return 0;
    }
    if(mm<=0||mm>=13){
        *date=-1;
        return 0;
    }
    if(leapyear(yy)==1&&mm==2){
        if(dd<=0||dd>=30){
            *date=-1;
            return 0;
        }
    }
    if(leapyear(yy)!=1&&mm==2){
        if(dd<=0||dd>=29){
            *date=-1;
            return 0;
        }
    }
    if(mm==1||mm==3||mm==5||mm==7||mm==8||mm==10||mm==12){
        if(dd<=0||dd>=32){
            *date=-1;
            return 0;
        }
    }
    if(mm==4||mm==6||mm==9||mm==11){
       if(dd<=0||dd>=31){
            *date=-1;
            return 0;
        }
    }

    *date=0;
    *date+=dd;
    mm--;
    while(mm>0){
        if(mm==1||mm==3||mm==5||mm==7||mm==8||mm==10||mm==12){
            *date+=31;
            mm--;
        }
        if(mm==4||mm==6||mm==9||mm==11){
            *date+=30;
            mm--;
        }
        if(leapyear(yy)==1&&mm==2){
            *date+=29;
            mm--;
        }
        if(leapyear(yy)!=1&&mm==2){
            *date+=28;
            mm--;
        }
    }
    while(yy>2020){
        if(leapyear(yy)==1){
            *date+=366;
            yy--;
        }
        else{
            *date+=365;
            yy--;
        }
    }
    return 1;
```

```c
}

char *datestring(int date){
    int yy,mm,dd;
    char ystring[5];
    char mstring[5];
    char dstring[5];
    char *dests;

    memset(ystring,'\0',sizeof(ystring));
    memset(mstring,'\0',sizeof(mstring));
    memset(dstring,'\0',sizeof(dstring));

    getdate(date,&yy,&mm,&dd);

    itoa(yy,ystring,10);
    itoa(mm,mstring,10);
    itoa(dd,dstring,10);

    //todo:!!!
    strcpy(dests,ystring);
    strcat(dests,".");
    strcat(dests,mstring);
    strcat(dests,".");
    strcat(dests,dstring);

    return dests;
}

int getsexbyID(char ID[20]){
    int x;

    x=ID[16]-'0';
    if(x%2==0){
        return 2;
    }
    else{
       return 1;
    }
}

int getagebyID(char ID[20]){
    int age;
    int IDyy,IDmm,IDdd;
    int yy,mm,dd;
    gettodaydate(&yy,&mm,&dd);
    IDyy=(ID[6]-'0')*1000+(ID[7]-'0')*100+(ID[8]-'0')*10+(ID[9]-'0');
    IDmm=(ID[10]-'0')*10+(ID[11]-'0');
    IDmm=(ID[10]-'0')*10+(ID[11]-'0');
    IDdd=(ID[12]-'0')*10+(ID[13]-'0');

    age=yy-IDyy;
    if(mm<IDmm)age--;
    if(mm==IDmm&&dd<IDdd)age--;

    return age;
}

void gettodaydate(int *yy,int *mm,int *dd){
    struct tm* ptr;
    time_t lt;

    time(&lt);
    ptr = localtime(&lt);


    *yy=ptr->tm_year + 1900;
    *mm=ptr->tm_mon + 1;
    *dd=ptr->tm_mday;

    *yy=2022;
    *mm=4;
    *dd=17;

    return;
}

int xpos(char c){
    if(c=='a')return 425;
    if(c=='b')return 325;
    if(c=='c')return 155;
    if(c=='d')return 55;
    if(c=='e')return 95;
    if(c=='f')return 305;
    if(c=='g')return 465;
    if(c=='h')return 485;
    if(c=='i')return 505;
    if(c=='j')return 565;
    if(c=='k')return 209;
    if(c=='l')return 229;
    if(c=='m')return 385;
    if(c=='n')return 200;
}

int ypos(char c){
    if(c=='a')return 120;
    if(c=='b')return 165;
    if(c=='c')return 210;
    if(c=='d')return 270;
    if(c=='e')return 310;
    if(c=='f')return 270;
    if(c=='g')return 240;
    if(c=='h')return 280;
    if(c=='i')return 250;
    if(c=='j')return 252;
    if(c=='k')return 420;
    if(c=='l')return 440;
    if(c=='m')return 330;
    if(c=='n')return 350;
}

void gettrackstring(char *trainname,int i,char *desttrack){
    if(strcmp(trainname,"G562")==0){
        if(i==1)strcpy(desttrack,"ab");return;
        if(i==2)strcpy(desttrack,"abg");return;
        if(i==3)strcpy(desttrack,"abgi");return;
        if(i==4)strcpy(desttrack,"abgij");return;
        if(i==5)strcpy(desttrack,"bg");return;
        if(i==6)strcpy(desttrack,"bgi");return;
        if(i==7)strcpy(desttrack,"bgij");return;
        if(i==8)strcpy(desttrack,"gi");return;
        if(i==9)strcpy(desttrack,"gij");return;
        if(i==10)strcpy(desttrack,"ij");return;
    }
    if(strcmp(trainname,"G567")==0){
        if(i==1)strcpy(desttrack,"jh");return;
        if(i==2)strcpy(desttrack,"jhm");return;
        if(i==3)strcpy(desttrack,"jhml");return;
        if(i==4)strcpy(desttrack,"jhmlk");return;
        if(i==5)strcpy(desttrack,"hm");return;
        if(i==6)strcpy(desttrack,"hml");return;
        if(i==7)strcpy(desttrack,"hmlk");return;
        if(i==8)strcpy(desttrack,"ml");return;
        if(i==9)strcpy(desttrack,"mlk");return;
        if(i==10)strcpy(desttrack,"lk");return;
    };
    if(strcmp(trainname,"G751")==0){
        if(i==1)strcpy(desttrack,"mf");return;
        if(i==2)strcpy(desttrack,"mfg");return;
        if(i==3)strcpy(desttrack,"mfgb");return;
        if(i==4)strcpy(desttrack,"mfgba");return;
        if(i==5)strcpy(desttrack,"fg");return;
        if(i==6)strcpy(desttrack,"fgb");return;
        if(i==7)strcpy(desttrack,"fgba");return;
        if(i==8)strcpy(desttrack,"gb");return;
        if(i==9)strcpy(desttrack,"gba");return;
        if(i==10)strcpy(desttrack,"ba");return;
    }
    if(strcmp(trainname,"G768")==0){
        if(i==1)strcpy(desttrack,"ke");return;
        if(i==2)strcpy(desttrack,"ked");return;
        if(i==3)strcpy(desttrack,"kedc");return;
        if(i==4)strcpy(desttrack,"kedcb");return;
        if(i==5)strcpy(desttrack,"ed");return;
        if(i==6)strcpy(desttrack,"edc");return;
        if(i==7)strcpy(desttrack,"edcb");return;
        if(i==8)strcpy(desttrack,"dc");return;
        if(i==9)strcpy(desttrack,"dcb");return;
        if(i==10)strcpy(desttrack,"cb");return;
    }
    if(strcmp(trainname,"G267")==0){
        if(i==1)strcpy(desttrack,"lk");return;
        if(i==2)strcpy(desttrack,"lkn");return;
        if(i==3)strcpy(desttrack,"lknf");return;
        if(i==4)strcpy(desttrack,"lknfa");return;
        if(i==5)strcpy(desttrack,"kn");return;
        if(i==6)strcpy(desttrack,"knf");return;
        if(i==7)strcpy(desttrack,"knfa");return;
        if(i==8)strcpy(desttrack,"nf");return;
        if(i==9)strcpy(desttrack,"nfa");return;
        if(i==10)strcpy(desttrack,"fa");return;
    }
    if(strcmp(trainname,"G186")==0){
        if(i==1)strcpy(desttrack,"gf");return;
        if(i==2)strcpy(desttrack,"gfn");return;
        if(i==3)strcpy(desttrack,"gfne");return;
        if(i==4)strcpy(desttrack,"gfned");return;
        if(i==5)strcpy(desttrack,"fn");return;
        if(i==6)strcpy(desttrack,"fne");return;
        if(i==7)strcpy(desttrack,"fned");return;
        if(i==8)strcpy(desttrack,"ne");return;
        if(i==9)strcpy(desttrack,"ned");return;
        if(i==10)strcpy(desttrack,"ed");return;
    }
    if(strcmp(trainname,"G379")==0){
        if(i==1)strcpy(desttrack,"hm");return;
        if(i==2)strcpy(desttrack,"hmf");return;
        if(i==3)strcpy(desttrack,"hmfc");return;
        if(i==4)strcpy(desttrack,"hmfcd");return;
        if(i==5)strcpy(desttrack,"mf");return;
        if(i==6)strcpy(desttrack,"mfc");return;
        if(i==7)strcpy(desttrack,"mfcd");return;
        if(i==8)strcpy(desttrack,"fc");return;
        if(i==9)strcpy(desttrack,"fcd");return;
        if(i==10)strcpy(desttrack,"cd");return;
    }
    if(strcmp(trainname,"G467")==0){
        if(i==1)strcpy(desttrack,"cf");return;
        if(i==2)strcpy(desttrack,"cfm");return;
        if(i==3)strcpy(desttrack,"cfmh");return;
        if(i==4)strcpy(desttrack,"cfmhj");return;
        if(i==5)strcpy(desttrack,"fm");return;
        if(i==6)strcpy(desttrack,"fmh");return;
        if(i==7)strcpy(desttrack,"fmhj");return;
        if(i==8)strcpy(desttrack,"mh");return;
        if(i==9)strcpy(desttrack,"mhj");return;
        if(i==10)strcpy(desttrack,"hj");return;
    }
    if(strcmp(trainname,"G685")==0){
        if(i==1)strcpy(desttrack,"im");return;
        if(i==2)strcpy(desttrack,"ima");return;
        if(i==3)strcpy(desttrack,"imab");return;
        if(i==4)strcpy(desttrack,"imabc");return;
        if(i==5)strcpy(desttrack,"ma");return;
        if(i==6)strcpy(desttrack,"mab");return;
        if(i==7)strcpy(desttrack,"mabc");return;
        if(i==8)strcpy(desttrack,"ab");return;
        if(i==9)strcpy(desttrack,"abc");return;
        if(i==10)strcpy(desttrack,"bc");return;
    }
    if(strcmp(trainname,"G335")==0){
        if(i==1)strcpy(desttrack,"af");return;
        if(i==2)strcpy(desttrack,"afn");return;
        if(i==3)strcpy(desttrack,"afnk");return;
        if(i==4)strcpy(desttrack,"afnkl");return;
        if(i==5)strcpy(desttrack,"fn");return;
        if(i==6)strcpy(desttrack,"fnk");return;
        if(i==7)strcpy(desttrack,"fnkl");return;
        if(i==8)strcpy(desttrack,"nk");return;
        if(i==9)strcpy(desttrack,"nkl");return;
        if(i==10)strcpy(desttrack,"kl");return;
    }
}

int samestringmax(char *a,char *b){
    int lengtha=0;
    int lengthb=0;
    char atemp[20]={'\0'};
    char btemp[20]={'\0'};
    int res[20][20]={0};
    int maxres=0;
    int i,j;

    while(*a!='\0'){
        atemp[lengtha]=*a;
        a++;
        lengtha++;
    }
    while(*b!='\0'){
        btemp[lengthb]=*b;
        b++;
        lengthb++;
    }

    for(i=0;i<lengtha;i++){
        for(j=0;j<lengthb;j++){
            if(atemp[i]==btemp[j]){
                res[i+1][j+1]=res[i][j]+1;
                maxres=max(maxres,res[i+1][j+1]);
            }
        }
    }

    return maxres;
}
```

3. control.c

```
C++

#include"mouse.h"

int inbarword16(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*16+4+left;
    int bottom=20+top;

    if(left==0){
        if(MouseX>320-8*width-2&&MouseY>top&&MouseX<320+8*width+2&&MouseY<bottom)return 1;
        else return 0;
    }

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
```

```c
}
int pressbarword16(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*16+4+left;
    int bottom=20+top;

    if(left==0){
        if(mouse_press(320-8*width-2,top,320+8*width+2,bottom)==1)return 1;
        else if(mouse_press(320-8*width-2,top,320+8*width+2,bottom)==2)return 2;
        else return 0;
    }
    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int inbarword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*24+6+left;
    int bottom=30+top;

    if(left==0){
        if(MouseX>320-12*width-3&&MouseY>top&&MouseX<320+12*width+3&&MouseY<bottom)return 1;
        else return 0;
    }

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
}
int pressbarword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*24+6+left;
    int bottom=30+top;

    if(left==0){
        if(mouse_press(320-12*width-3,top,320+12*width+3,bottom)==1)return 1;
        else if(mouse_press(320-12*width-3,top,320+12*width+3,bottom)==2)return 2;
        else return 0;
    }
    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int inbarword32(int left,int top,int barcolor,int width,int n,char *s,int word
color){
    int right=width*32+8+left;
    int bottom=40+top;

    if(left==0){
        if(MouseX>320-16*width-4&&MouseY>top&&MouseX<320+16*width+4&&MouseY<bottom)return 1;
        else return 0;
    }

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
}
int pressbarword32(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*32+8+left;
    int bottom=40+top;

    if(left==0){
        if(mouse_press(320-16*width-4,top,320+16*width+4,bottom)==1)return 1;
        else if(mouse_press(320-16*width-4,top,320+16*width+4,bottom)==2)return 2;
        else return 0;
    }
    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int inbarword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*48+12+left;
    int bottom=60+top;

    if(left==0){
        if(MouseX>320-24*width-6&&MouseY>top&&MouseX<320+24*width+6&&MouseY<bottom)return 1;
        else return 0;
    }

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
}
int pressbarword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*48+12+left;
    int bottom=60+top;

```

```c
    if(left==0){
        if(mouse_press(320-24*width-6,top,320+24*width+6,bottom)==1)return 1;
        else if(mouse_press(320-24*width-6,top,320+24*width+6,bottom)==2)return 2;
        else return 0;
    }
    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int inbarwordframe(int left,int top,int barcolor,int width,char *s,int wordcolor,int framecolor){
    int right=width*32+8+left;
    int bottom=40+top;

    if(left==0){
        if(MouseX>320-16*width-4&&MouseY>top&&MouseX<320+16*width+4&&MouseY<bottom)return 1;
        else return 0;
    }

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
}
int pressbarwordframe(int left,int top,int barcolor,int width,char *s,int wordcolor,int framecolorr){
    int right=width*32+8+left;
    int bottom=40+top;

    if(left==0){
        if(mouse_press(320-16*width-4,top,320+16*width+4,bottom)==1)return 1;
        else if(mouse_press(320-16*width-4,top,320+16*width+4,bottom)==2)return 2;
        else return 0;
    }
    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int incircle(int x,int y,int radius){
    int left=x-radius;
    int right=x+radius;
    int top=y-radius;
    int bottom=y+radius;

    if(MouseX>left&&MouseY>top&&MouseX<right&&MouseY<bottom)return 1;
    else return 0;
}
int presscircle(int x,int y,int radius){
    int left=x-radius;
    int right=x+radius;
    int top=y-radius;
    int bottom=y+radius;

    if(mouse_press(left,top,right,bottom)==1)return 1;
    else if(mouse_press(left,top,right,bottom)==2)return 2;
    else return 0;
}

int inreturnbutton(void){
    if(MouseX>10&&MouseY>5&&MouseX<33&&MouseY<30)return 1;
    else return 0;
}
int pressreturnbutton(void){
    if(mouse_press(10,5,33,30)==1)return 1;
    else if(mouse_press(10,5,33,30)==2)return 2;
    else return 0;
}

int insearchbuttuon(void){
    if(MouseX>520&&MouseY>100&&MouseX<550&&MouseY<140)return 1;
    else return 0;
}
int presssearchbutton(void){
    if(mouse_press(510,100,550,140)==1)return 1;
    else if(mouse_press(510,100,550,140)==2)return 2;
    else return 0;
}

int inpalabelu(void){
    if(MouseX>60&&MouseY>130&&MouseX<580&&MouseY<210)return 1;
    else return 0;
}
int presspalabelu(void){
    if(mouse_press(60,130,580,210)==1)return 1;
    else if(mouse_press(60,130,580,210)==2)return 2;
    else return 0;
}

int inpalabelm(void){
    if(MouseX>60&&MouseY>230&&MouseX<580&&MouseY<310)return 1;
    else return 0;
}
```

```
int presspalabelm(void){
    if(mouse_press(60,230,580,310)==1)return 1;
    else if(mouse_press(60,230,580,310)==2)return 2;
    else return 0;
}

int inpalabell(void){
    if(MouseX>60&&MouseY>330&&MouseX<580&&MouseY<410)return 1;
    else return 0;
}
int presspalabell(void){
    if(mouse_press(60,330,580,410)==1)return 1;
    else if(mouse_press(60,330,580,410)==2)return 2;
    else return 0;
}

int inleftarrow(void){
    if(MouseX>250&&MouseY>430&&MouseX<270&&MouseY<450)return 1;
    else return 0;
}
int pressleftarrow(void){
    if(mouse_press(250,430,270,450)==1)return 1;
    else if(mouse_press(250,430,270,450)==2)return 2;
    else return 0;
}

int inrightarrow(void){
    if(MouseX>370&&MouseY>430&&MouseX<390&&MouseY<450)return 1;
    else return 0;
}
int pressrightarrow(void){
    if(mouse_press(370,430,390,450)==1)return 1;
    else if(mouse_press(370,430,390,450)==2)return 2;
    else return 0;
}

int intrack1(void){
    if(MouseX>(110-3)&&MouseY>190&&MouseX<(110+24*6+3)&&MouseY<(190+24))return 1;
    else return 0;
}
int presstrack1(void){
    if(mouse_press(110,190,110+24*6+6,190+24)==1)return 1;
    else if(mouse_press(110,190,110+24*6+6,190+24)==2)return 2;
    else return 0;
}

int intrack2(void){
```

```c
int intrack2(void){
    if(MouseX>(110-3)&&MouseY>246&&MouseX<(110+24*6+3)&&MouseY<(246+24))return 1;
    else return 0;
}
int presstrack2(void){
    if(mouse_press(110,246,110+24*6+6,246+24)==1)return 1;
    else if(mouse_press(110,246,110+24*6+6,246+24)==2)return 2;
    else return 0;
}

int intrack3(void){
    if(MouseX>(110-3)&&MouseY>302&&MouseX<(110+24*6+3)&&MouseY<(302+24))return 1;
    else return 0;
}
int presstrack3(void){
    if(mouse_press(110,302,110+24*6+6,302+24)==1)return 1;
    else if(mouse_press(110,302,110+24*6+6,302+24)==2)return 2;
    else return 0;
}

int intrack4(void){
    if(MouseX>(110-3)&&MouseY>358&&MouseX<(110+24*6+3)&&MouseY<(358+24))return 1;
    else return 0;
}
int presstrack4(void){
    if(mouse_press(110,358,110+24*6+6,358+24)==1)return 1;
    else if(mouse_press(110,358,110+24*6+6,358+24)==2)return 2;
    else return 0;
}

int intrack5(void){
    if(MouseX>(110-3)&&MouseY>414&&MouseX<(110+24*6+3)&&MouseY<(414+24))return 1;
    else return 0;
}
int presstrack5(void){
    if(mouse_press(110,414,110+24*6+6,414+24)==1)return 1;
    else if(mouse_press(110,414,110+24*6+6,414+24)==2)return 2;
    else return 0;
}

int intrack6(void){
    if(MouseX>(380-3)&&MouseY>190&&MouseX<(380+24*6+3)&&MouseY<(190+24))return 1;
    else return 0;
```

```
}
int presstrack6(void){
    if(mouse_press(380,190,380+24*6+6,190+24)==1)return 1;
    else if(mouse_press(380,190,380+24*6+6,190+24)==2)return 2;
    else return 0;
}

int intrack7(void){
    if(MouseX>(380-3)&&MouseY>246&&MouseX<(380+24*6+3)&&MouseY<(246+24))return 1;
    else return 0;
}
int presstrack7(void){
    if(mouse_press(380,246,380+24*6+6,246+24)==1)return 1;
    else if(mouse_press(380,246,380+24*6+6,246+24)==2)return 2;
    else return 0;
}

int intrack8(void){
    if(MouseX>(380-3)&&MouseY>302&&MouseX<(380+24*6+3)&&MouseY<(302+24))return 1;
    else return 0;
}
int presstrack8(void){
    if(mouse_press(380,302,380+24*6+6,302+24)==1)return 1;
    else if(mouse_press(380,302,380+24*6+6,302+24)==2)return 2;
    else return 0;
}

int intrack9(void){
    if(MouseX>(380-3)&&MouseY>358&&MouseX<(380+24*6+3)&&MouseY<(358+24))return 1;
    else return 0;
}
int presstrack9(void){
    if(mouse_press(380,358,380+24*6+6,358+24)==1)return 1;
    else if(mouse_press(380,358,380+24*6+6,358+24)==2)return 2;
    else return 0;
}

int intrack10(void){
    if(MouseX>(380-3)&&MouseY>414&&MouseX<(380+24*6+3)&&MouseY<(414+24))return 1;
    else return 0;
}
int presstrack10(void){
    if(mouse_press(380,414,380+24*6+6,414+24)==1)return 1;
    else if(mouse_press(380,414,380+24*6+6,414+24)==2)return 2;
    else return 0;
}
```

4. drawpage.c

```c
#include"public.h"

void drawstart(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();

    barword32(0,90,BLUE,7,7,"请选择登录方式",LIGHTGRAY);
    barword32(0,200,BLUE,5,4,"乘客登录",LIGHTGRAY);
    barword32(0,300,BLUE,5,5,"管理员登录",LIGHTGRAY);
    return;
}

void drawadlogin(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();

    barword32(0,85,BLUE,6,5,"管理员登录",LIGHTGRAY);
    barword32(60,170,BLUE,3,2,"账号",LIGHTGRAY);
    barword32(60,250,BLUE,3,2,"密码",LIGHTGRAY);
    barwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED);
    barwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED);
    barword32(0,350,BLUE,3,2,"登录",LIGHTGRAY);
    returnbutton(LIGHTGRAY);

    return;
 }

void drawpalogin(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();

    barword32(0,85,BLUE,5,4,"乘客登录",LIGHTGRAY);
    barword32(60,170,BLUE,3,2,"账号",LIGHTGRAY);
    barword32(60,250,BLUE,3,2,"密码",LIGHTGRAY);
    barwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED);
    barwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED);
    barword32(0,350,BLUE,3,2,"登录",LIGHTGRAY);
    barword16(450,420,BLUE,5,5,"新用户注册",LIGHTGRAY);
    returnbutton(LIGHTGRAY);
    return;
}

void drawparegister(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,85,BLUE,5,4,"乘客注册",LIGHTGRAY);
    barword32(60,170,BLUE,3,2,"账号",LIGHTGRAY);
    barwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED);

    barword32(60,250,BLUE,3,2,"密码",LIGHTGRAY);
    barwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED);

    barwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED);
    puthz(190,335,"再次输入密码",32,32,LIGHTGRAY);
    barword32(0,400,BLUE,2,2,"注册",LIGHTGRAY);
}

void drawpabind(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,120,BLUE,6,6,"绑定个人信息",WHITE);
    puthz(340-10.5*16,70,"该账号尚未绑定个人信息，请绑定个人信息",16,16,RED);

    barword32(60,200,BLUE,4,4,"身份证号",WHITE);
    barwordframe(210,200,WHITE,11,"",LIGHTGRAY,RED);
    barword32(60,300,BLUE,4,4,"电话号码",WHITE);
    barwordframe(210,300,WHITE,11,"",LIGHTGRAY,RED);

    barword32(0,400,BLUE,2,2,"绑定",WHITE);
}

void drawpamessage(struct passenger *pa){
    char cname[10];

    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,70,RED,6,4,"乘客信息",WHITE);
    setfillstyle(1,WHITE);
    bar(50,140,590,352);

    puthz(70,150,"姓名",32,32,LIGHTGRAY);
    puthz(250,150,"性别",32,32,LIGHTGRAY);
    puthz(420,150,"年龄",32,32,LIGHTGRAY);
    puthz(70,200,"身份证号",32,32,LIGHTGRAY);
    puthz(70,250,"电话",32,32,LIGHTGRAY);
    puthz(70,300,"健康状况",32,32,LIGHTGRAY);

    barword32(65,400,BLUE,4,4,"登记乘车",LIGHTGRAY);
    barword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY);
    barword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY);

    getcnamebyID(pa->ID,cname);
    puthz(140,150,cname,32,32,LIGHTGRAY);

    if(pa->sex==1){
        puthz(350,150,"男",32,32,LIGHTGRAY);
    }
    else if(pa->sex==2){
        puthz(350,150,"女",32,32,LIGHTGRAY);
    }
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);
    putnum(515,145,pa->age);
    outtextxy(205,195,pa->ID);
    outtextxy(200,245,pa->tel);
    if(pa->status==0){
        puthz(300,300,"无异常",32,32,GREEN);
    }
    if(pa->status==1){
        puthz(300,300,"新冠肺炎患者",32,32,RED);
    }
    if(pa->status==2){
        puthz(300,300,"无症状感染者",32,32,RED);
    }
    if(pa->status==3){
        puthz(300,300,"密切接触者",32,32,YELLOW);
    }
}

void drawpapostrecord(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);
```

```c
    returnbutton(LIGHTGRAY);

    barword32(0,70,BLUE,4,4,"登记乘车",RED);
    todaydate();

    barword32(110,185,WHITE,2,2,"车次",LIGHTGRAY);
    setcolor(BLUE);
    settextstyle(1,0,5);
    outtextxy(240,170,"G");
    barwordframe(280,185,WHITE,3,"",0,RED);
    barword32(430,185,BLUE,2,2,"确认",WHITE);
}

void drawpagetrecord(struct pagetrain *ptrnu,struct pagetrain *ptrnm,struct pagetrain *ptrnl,int currentpage,int countpage){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,65,WHITE,4,4,"乘车记录",RED);
    if(ptrnu!=NULL)trlabelu(ptrnu);
    if(ptrnm!=NULL)trlabelm(ptrnm);
    if(ptrnl!=NULL)trlabell(ptrnl);

    leftarrow(BLUE);
    rightarrow(BLUE);
    pagenumber(currentpage,countpage);

    return ;
}

void drawpagettrack(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    drawmap();
}

void drawadlabel(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    puthz(320-32*3.5,80,"请录入阳性患者",32,32,GREEN);
    barword32(80,150,GREEN,4,4,"身份证号",WHITE);
    barwordframe(230,150,WHITE,11,"",0,RED);

    barword24(80,240,GREEN,8,8,"是否无症状感染者",WHITE);
    puthz(330,240,"是",24,24,GREEN);
    puthz(430,240,"否",24,24,GREEN);
    setcolor(GREEN);
    circle(380,252,10);
    circle(480,252,10);

    barword32(0,320,GREEN,4,4,"确认录入",WHITE);
    return;
}

void drawregister(){        //乘客登记信息界面

    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();

    barword32(75,150,BLUE,4,2,"姓名",WHITE);
    barwordframe(250,150,WHITE,10,"",0,RED);
    barword32(75,200,BLUE,4,4,"身份证号",WHITE);
    barwordframe(250,200,WHITE,10,"",0,RED);
    barword32(75,300,BLUE,4,3,"电话",WHITE);
    barwordframe(250,300,WHITE,10,"",0,RED);
    barword32(0,400,BLUE,2,2,"注册",WHITE);

    return;
}

void drawstationselect(char *trainname){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,60,BLUE,4,4,"车次信息",WHITE);
    puthz(360,130,"请选择出发地与目的地",24,24,RED);
    puthz(60,130,"列车号",32,32,RED);
    setcolor(RED);
    settextstyle(1,0,3);
    outtextxy(200,130,trainname);
    track1(trainname,WHITE);
    track2(trainname,WHITE);
    track3(trainname,WHITE);
    track4(trainname,WHITE);
    track5(trainname,WHITE);
    track6(trainname,WHITE);
    track7(trainname,WHITE);
    track8(trainname,WHITE);
    track9(trainname,WHITE);
    track10(trainname,WHITE);
}

void drawadmanager(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    searchbutton(WHITE);
    barwordframe(100,100,WHITE,11,"",WHITE,RED);
    barword32(100,200,CYAN,6,6,"标记乘客状况",WHITE);
    barword32(350,200,DARKGRAY,6,6,"查询阳性乘客",WHITE);
    barword32(350,300,BLUE,6,6,"查询密接乘客",WHITE);
    barword32(100,300,BROWN,6,6,"查询可疑车次",WHITE);
}

void drawadhealth(){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();

    puthz(25,150,"请输入身份证号",32,32,WHITE);
    barwordframe(250,150,WHITE,11,"",0,RED);
    barword32(320-32*6-10,300,BLUE,6,6,"查询健康状况",LIGHTGRAY);
    barword32(320+10,300,RED,6,6,"查询出行记录",LIGHTGRAY);
}

void drawadgetpospa(struct passenger *pau,struct passenger *pam,struct passeng
er *pal,int currentpage,int countpage){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,65,WHITE,6,6,"阳性乘客信息",RED);
    if(pau!=NULL)palabelu(pau);
    if(pam!=NULL)palabelm(pam);
    if(pal!=NULL)palabell(pal);

    leftarrow(BLUE);
    rightarrow(BLUE);
    pagenumber(currentpage,countpage);
```

```c

    return ;
}

void drawadgetcttpa(struct passenger *pau,struct passenger *pam,struct passenger *pal,int currentpage,int countpage){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,65,WHITE,6,6,"密接乘客信息",RED);
    if(pau!=NULL)palabelu(pau);
    if(pam!=NULL)palabelm(pam);
    if(pal!=NULL)palabell(pal);

    leftarrow(BLUE);
    rightarrow(BLUE);
    pagenumber(currentpage,countpage);

    return ;
}

void drawadgetpostr(struct pagetrain *ptrnu,struct pagetrain *ptrnm,struct pagetrain *ptrnl,int currentpage,int countpage){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,65,WHITE,6,6,"阳性车次信息",RED);
    if(ptrnu!=NULL)trlabelu(ptrnu);
    if(ptrnm!=NULL)trlabelm(ptrnm);
    if(ptrnl!=NULL)trlabell(ptrnl);

    leftarrow(BLUE);
    rightarrow(BLUE);
    pagenumber(currentpage,countpage);
}

void drawadsearch(struct passenger *pau,struct passenger *pam,struct passenger *pal,int currentpage,int countpage){
    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,65,WHITE,4,4,"搜索结果",RED);
    if(pau!=NULL)palabelu(pau);
    if(pam!=NULL)palabelm(pam);
    if(pal!=NULL)palabell(pal);

    leftarrow(BLUE);
    rightarrow(BLUE);
    pagenumber(currentpage,countpage);
}

void drawadpamessage(struct passenger *pa){
    char cname[10];

    cleardevice();
    setbkcolor(LIGHTGRAY);
    drawtop();
    returnbutton(LIGHTGRAY);

    barword32(0,70,RED,6,4,"乘客信息",WHITE);
    setfillstyle(1,WHITE);
    bar(50,140,590,352);

    puthz(70,150,"姓名",32,32,LIGHTGRAY);
    puthz(250,150,"性别",32,32,LIGHTGRAY);
    puthz(420,150,"年龄",32,32,LIGHTGRAY);
    puthz(70,200,"身份证号",32,32,LIGHTGRAY);
    puthz(70,250,"电话",32,32,LIGHTGRAY);
    puthz(70,300,"健康状况",32,32,LIGHTGRAY);

    barword32(65,400,BLUE,4,4,"登记乘车",RED);
    barword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY);
    barword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY);

    getcnamebyID(pa->ID,cname);
    puthz(140,150,cname,32,32,LIGHTGRAY);

    if(pa->sex==1){
        puthz(350,150,"男",32,32,LIGHTGRAY);
    }
    else if(pa->sex==2){
        puthz(350,150,"女",32,32,LIGHTGRAY);
    }
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);
    putnum(515,145,pa->age);
    outtextxy(205,195,pa->ID);
    outtextxy(200,245,pa->tel);
    if(pa->status==0){
        puthz(300,300,"无异常",32,32,GREEN);
    }
    if(pa->status==1){
        puthz(300,300,"新冠肺炎患者",32,32,RED);
    }
    if(pa->status==2){
        puthz(300,300,"无症状感染者",32,32,RED);
    }
    if(pa->status==3){
        puthz(300,300,"密切接触者",32,32,YELLOW);
    }
}
```

5. file.c

C

```c
#include"public.h"

void initfile(void){
    FILE *log,*fadministrator,*fuser,*ftrain,*frecord,*fpassenger;

    log=fopen(".\\LOGGER","w+");

    fadministrator=fopen(".\\DB\\ADMIN","wb+");
    fuser=fopen(".\\DB\\USER","wb+");
    ftrain=fopen(".\\DB\\TRAIN","wb+");
    frecord=fopen(".\\DB\\RECORD","wb+");
    fpassenger=fopen(".\\DB\\PASSENG","wb+");

    fclose(log);
    fclose(fadministrator);
    fclose(fuser);
    fclose(ftrain);
    fclose(frecord);
    fclose(fpassenger);

    writeuser("xyd","993995");
    writeuser("zzl","993995");
    writeadmin("xyd","993995");
    writeadmin("zzl","993995");
    writetrain("G562",2022,4,17);
    writetrain("G186",2022,4,17);
    writetrain("G379",2022,4,17);
    writetrain("G335",2022,4,17);
    writepassenger(1,"320602200308222547","15062744586",2,18,1);
    writepassenger(2,"320602200211035913","18921673386",1,19,0);
    writerecordv1(2,2,"fne");
    writerecordv1(2,3,"fcd");
    writerecordv1(2,4,"nkl");
}

int writeuser(char *name,char *password){
    FILE* fuser;
    FILE* log;
    struct user* usr;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fuser=fopen(".\\DB\\USER", "rb+" ))==NULL)
    {
        fprintf(log,"\nerr:open user fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fuser,0,SEEK_END);
    n=ftell(fuser)/sizeof(USER);

    if((usr=(USER*)malloc(sizeof(USER)))==NULL){
        fprintf(log,"\nerr:no space for user!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    usr->pakey=n+1;
    strcpy(usr->name,name);
    strcpy(usr->password,password);

    fseek(fuser,0,SEEK_END);
    fwrite(usr,sizeof(USER),1,fuser);
    fflush(fuser);

    if (usr!=NULL)
    {
        free(usr);
        usr=NULL;
    }

```

```c
    if (fclose(fuser)!=0)
    {
        fprintf(log,"\nerr:close user fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:write user pakey %d name %s password %s success!",n+1,name,password);

    fclose(log);
    return 1;
}

//pakey:登录成功 0:文件操作错误 -1: 密码错误 -2: 用户未注册 -3: 未知错误
int loginuser(char *name,char *password){
    FILE* user;
    FILE* log;
    struct user* usr;
    int i;
    int n;
    int pakey;

    log=fopen(".\\LOGGER","a+");

    if((user=fopen(".\\DB\\USER", "rb+" ))==NULL)
    {
        fprintf(log,"\nerr:open user fail!");
        delay(3000);
        exit(1);
        fclose(log);
        return 0;
    }
    fseek(user,0,SEEK_END);
    n=ftell(user)/sizeof(USER);

    for(i=0;i<n;i++){
        if ((usr=(USER*)malloc(sizeof(USER)))==NULL){
            fprintf(log,"\nerr:no space for user!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(user,i*sizeof(USER),SEEK_SET);
        fread(usr,sizeof(USER),1,user);

        if(strcmp(name,usr->name)==0){
            if(strcmp(password,usr->password)==0){
                pakey=usr->pakey;

                if (usr!=NULL){
                    free(usr);
                    usr=NULL;
                }
                if (fclose(user)!= 0){
                    fprintf(log,"\nerr:close user fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }

                fprintf(log,"\ninfo:user %s login success!",name);

                fclose(log);
                return pakey;

            }
            else if(strcmp(password,usr->password)!=0){
                if (usr!=NULL){
                    free(usr);
                    usr=NULL;
                }
                if (fclose(user)!= 0){
                    fprintf(log,"\nerr:close user fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }
                fprintf(log,"\ninfo:user %s login fail:password wrong!",name);

                fclose(log);
                return -1;
            }
        }
        if (usr!=NULL){
            free(usr);
            usr=NULL;
        }

    }
    if(i==n){
        if (usr!=NULL){
            free(usr);
            usr=NULL;
        }
        if (fclose(user)!= 0){
            fprintf(log,"\nerr:close user fail!");
            delay(3000);
            exit(1);
            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:user %s login fail:no register!",name);

        fclose(log);
        return -2;
    }

    fprintf(log,"\nerr:user %s login unknown fail!",name);

    fclose(log);
    return -3;
}

//1: 存在 -1: 不存在
int existusername(char *name){
    FILE* fuser;
    FILE* log;
    struct user* usr;
    int i;
    int n;
    log=fopen(".\\LOGGER","a+");

    if((fuser=fopen(".\\DB\\USER", "rb+"))==NULL){
        fprintf(log,"\nerr:open user fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fuser,0,SEEK_END);
    n=ftell(fuser)/sizeof(USER);
    n=ftell(fuser)/sizeof(USER);

    for(i=0;i<n;i++){
        if((usr=(USER*)malloc(sizeof(USER)))==NULL){
            fprintf(log,"\nerr:no space for user!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(fuser,i*sizeof(USER),SEEK_SET);
        fread(usr,sizeof(USER),1,fuser);
        if(strcmp(name,usr->name)==0){
            if(usr != NULL){
                free(usr);
                usr = NULL;
            }
            if(fclose(fuser) != 0){
                printf(log,"\nerr:close user fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            if(usr!= NULL){
                free(usr);
                usr = NULL;
            }
            fclose(log);
            return 1;
        }
        if(usr!= NULL){
            free(usr);
            usr = NULL;
        }
    }

    if(fclose(fuser)!= 0){
        printf(log,"\nerr:close user fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
```

```c
    }

    fclose(log);
    return -1;
}

int writeadmin(char *name,char *password){
    FILE* fadministrator;
    FILE* log;
    struct administrator* admin;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fadministrator=fopen(".\\DB\\ADMIN", "rb+" ))==NULL)
    {
        fprintf(log,"\nerr:open administrator fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fadministrator,0,SEEK_END);
    n=ftell(fadministrator)/sizeof(ADMINISTRATOR);

    if((admin=(ADMINISTRATOR*)malloc(sizeof(ADMINISTRATOR)))==NULL){
        fprintf(log,"\nerr:no space for administrator!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    admin->adkey=n+1;
    strcpy(admin->name,name);
    strcpy(admin->password,password);

    fseek(fadministrator,0,SEEK_END);
    fwrite(admin,sizeof(ADMINISTRATOR),1,fadministrator);
    fflush(fadministrator);

    if (admin!=NULL)
    {
        free(admin);
        admin=NULL;
    }

    if (fclose(fadministrator)!=0)
    {
        fprintf(log,"\nerr:close administrator fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:write administrator adkey %d name %s password %s success!",n+1,name,password);

    fclose(log);
    return 1;
}

//1:登录成功 0:文件操作错误 -1: 密码错误 -2: 用户未注册 -3: 未知错误
int loginadmin(char *name,char *password){
    FILE* fadministrator;
    FILE* log;
    struct administrator* admin;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fadministrator=fopen(".\\DB\\ADMIN", "rb+" ))==NULL)
    {
        fprintf(log,"\nerr:open administrator fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }
    fseek(fadministrator,0,SEEK_END);

    n=ftell(fadministrator)/sizeof(ADMINISTRATOR);

    for(i=0;i<n;i++){
        if ((admin=(ADMINISTRATOR*)malloc(sizeof(ADMINISTRATOR)))==NULL){
            fprintf(log,"\nerr:no space for administrator!");
            delay(3000);
            exit(1);
            fclose(log);
            return 0;
        }
        fseek(fadministrator,i*sizeof(ADMINISTRATOR),SEEK_SET);
        fread(admin,sizeof(ADMINISTRATOR),1,fadministrator);

        if(strcmp(name,admin->name)==0){
            if(strcmp(password,admin->password)==0){
                if (admin!=NULL){
                    free(admin);
                    admin=NULL;
                }
                if (fclose(fadministrator)!= 0){
                    fprintf(log,"\nerr:close administrator fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }

                fprintf(log,"\ninfo:admministrator %s login success!",name);

                fclose(log);
                return 1;

            }
            else if(strcmp(password,admin->password)!=0){
                if (admin!=NULL){
                    free(admin);
                    admin=NULL;
                }
                if (fclose(fadministrator)!= 0){
                    fprintf(log,"\nerr:close administrator fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }
                fprintf(log,"\ninfo:administrator %s login fail:password wrong!",name);

                fclose(log);
                return -1;
            }
        }
        if (admin!=NULL){
            free(admin);
            admin=NULL;
        }
    }
    if(i==n){
        if (admin!=NULL){
            free(admin);
            admin=NULL;
        }
        if (fclose(fadministrator)!= 0){
            fprintf(log,"\nerr:close administrator fail!");
            delay(3000);
            exit(1);
            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:administrator %s login fail:no register!",name);

        fclose(log);
        return -2;
    }

    fprintf(log,"\nerr:administrator %s login unknown fail!",name);

    fclose(log);
    return -3;
}

int writepassenger(int pakey,char *ID,char *tel,int sex,int age,int status){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        fprintf(log,"\nerr:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    if((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
        fprintf(log,"\nerr:no space for passenger!");
        delay(3000);
```

```c
        exit(1);

        fclose(log);
        return 0;
    }

    memset(pa,'\0',sizeof(pa));

    pa->pakey=pakey;
    pa->age=age;
    pa->sex=sex;
    pa->status=status;
    strcpy(pa->ID,ID);
    strcpy(pa->tel,tel);

    fseek(fpassenger,0,SEEK_END);
    fwrite(pa,sizeof(PASSENGER),1,fpassenger);
    fflush(fpassenger);

    if (pa != NULL){
        free(pa);
        pa = NULL;
    }

    if (fclose(fpassenger)!=0){
        fprintf(log,"\nerr:close passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:write passenger pakey %d ID %s tel %s sex %d age %d status %d success!",pakey,ID,tel,sex,age,status);
    fclose(log);
    return 1;
}

//1.存在  -1.不存在
int existpassengerID(char *ID){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");
    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG","rb+"))==NULL){
        fprintf(log,"\neer:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for passenger!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);
        if(strcmp(ID,pa->ID)==0){
            if(pa!= NULL){
                free(pa);
                pa= NULL;
            }

            if(fclose(fpassenger) != 0){
                printf(log,"\nerr:close passenger fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            fclose(log);
            return 1;
        }
        if(pa!= NULL){
            free(pa);
            pa= NULL;
        }
```

```c
    }

    if(fclose(fpassenger)!= 0){
        printf(log,"\nerr:close passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return -1;
}

//-1: 没有该pakey 1: 成功获取
int getpassengerbypakey(int pakey,struct passenger *destpa){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        fprintf(log,"\nerr:open passenger fail!");
        fclose(log);

        delay(3000);
        exit(1);

        return 0;                                                    //???
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if ((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for passenger!");
            fclose(log);

            delay(3000);
            exit(1);

            return 0;
        }
        }

        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);

        if(pa->pakey==pakey){
            destpa->pakey=pa->pakey;
            destpa->age=pa->age;
            destpa->status=pa->status;
            destpa->sex=pa->sex;
            strcpy(destpa->ID,pa->ID);
            strcpy(destpa->tel,pa->tel);

            if (pa!=NULL){
                free(pa);
                pa=NULL;
            }

            if (fclose(fpassenger)!= 0){
                fprintf(log,"\nerr:close passenger fail!");
                fclose(log);

                delay(3000);
                exit(1);

                return 0;
            }

            fprintf(log,"\ninfo:get passenger by pakey %d success!",pakey);

            fclose(log);
            return 1;
        }

        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
    }
    if(i==n){
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
        if (fclose(fpassenger)!= 0){
            fprintf(log,"\nerr:close passenger fail!");
            fclose(log);

            delay(3000)
            delay(3000);
            exit(1);

            return 0;
        }

        fprintf(log,"\ninfo:get passenger by pakey %d fail:no bind passenger (no pakey in passenger)!",pakey);

        fclose(log);
        return -1;
    }
}

//返回获得的个数
int getpassengerbystatus(int status,struct passenger destpa[20]){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;
    int j=0;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        printf("\nerr:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                           //???
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if ((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for passenger!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);

        if(pa->status==status||(status==1&&pa->status==2)){
            destpa[j].pakey=pa->pakey;
            destpa[j].sex=pa->sex;
            destpa[j].age=pa->age;
            destpa[j].status=pa->status;
            strcpy(destpa[j].ID,pa->ID);
            strcpy(destpa[j].tel,pa->tel);

            j++;
            fprintf(log,"\ninfo:get passenger:pakey %d ID %s tel %s sex %d age %d by status %d success!"
                       ,pa->pakey,pa->ID,pa->tel,pa->sex,pa->age,pa->status);

        }
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
    }

    if (fclose(fpassenger)!= 0){
        fprintf(log,"\nerr:close passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return j;
}

int getpassengerbysearch(char *searchID,struct passenger destpa[20]){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;
    int j=0;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        printf("\nerr:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if ((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for passenger!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);

        if(samestringmax(searchID,pa->ID)>=6){
            destpa[j].pakey=pa->pakey;
            destpa[j].sex=pa->sex;
            destpa[j].age=pa->age;
            destpa[j].status=pa->status;
            strcpy(destpa[j].ID,pa->ID);
            strcpy(destpa[j].tel,pa->tel);

            j++;
            fprintf(log,"\ninfo:get passenger:pakey %d ID %s tel %s sex %d age %d by search %s success!"
                       ,pa->pakey,pa->ID,pa->tel,pa->sex,pa->age,searchID);
        }
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
    }

    if (fclose(fpassenger)!= 0){
        fprintf(log,"\nerr:close passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
```

```c
        return 0;
    }

    fclose(log);
    return j;
}

//-1: 没有相应的ID 1: 操作成功
int getpakeybyID(char *ID,int *pakey){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        fprintf(log,"\nerr:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if ((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for passenger!");
            delay(3000);
            exit(1);

            fclose(log);

            return 0;
        }
        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);
        // fprintf(log,"\ndebug:%s",pa->ID);                               //ok
        if(strcmp(pa->ID,ID)==0){

            *pakey=pa->pakey;

            if (pa!=NULL){
                free(pa);
                pa=NULL;

            }

            if (fclose(fpassenger)!= 0){
                fprintf(log,"\nerr:close passenger fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            fprintf(log,"\ninfo:get pakey by ID %s success:pakey %d!",ID,*pake
y);
            fclose(log);
            return 1;
        }
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
    }
    if(i==n){
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
        if (fclose(fpassenger)!= 0){
            fprintf(log,"\nerr:close passenger fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:get pakey by ID %s fail:no ID in passenger!",ID);
        return -1;
    }
}

//-2: 相同状态 -1: 没有pakey 1: 操作成功
int updatestatusbypakey(int pakey,int status){
    FILE* fpassenger;
    FILE* log;
    struct passenger *pa;
    int i;
    int n;
```

```c
    int n;

    log=fopen(".\\LOGGER","a+");

    if((fpassenger=fopen(".\\DB\\PASSENG", "rb+" ))==NULL){
        fprintf(log,"\nerr:open passenger fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(fpassenger,0,SEEK_END);
    n=ftell(fpassenger)/sizeof(PASSENGER);

    for(i=0;i<n;i++){
        if ((pa=(PASSENGER*)malloc(sizeof(PASSENGER)))==NULL){
            fprintf(log,"\nerr:no space for user!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
        fread(pa,sizeof(PASSENGER),1,fpassenger);


        if(pa->pakey==pakey){
            if(pa->status==status){
                fprintf(log,"\ninfo:update status %d by pakey %d fail:same status %d!",status,pakey,status);

                if (pa!=NULL){
                    free(pa);
                    pa=NULL;
                }

                if (fclose(fpassenger)!= 0){
                    fprintf(log,"\nerr:close passenger fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }

                fclose(log);
                return -2;
            }

            if(status==3&&(pa->status==1||pa->status==2)){
                fprintf(log,"\ninfo:update status %d by pakey %d fail:ctt to pos!",status,pakey);

                if (pa!=NULL){
                    free(pa);
                    pa=NULL;
                }

                if (fclose(fpassenger)!= 0){
                    fprintf(log,"\nerr:close passenger fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }

                fclose(log);
                return -2;
            }

            pa->status=status;
            fseek(fpassenger,i*sizeof(PASSENGER),SEEK_SET);
            fwrite(pa,sizeof(PASSENGER),1,fpassenger);

            if (fclose(fpassenger)!= 0){
                fprintf(log,"\nerr:close passenger fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            if (pa!=NULL){
                free(pa);
                pa=NULL;
            }

            fprintf(log,"\ninfo:update status %d by pakey %d success!",status,pakey);
            fclose(log);
            return 1;
```

```c
        }
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
    }
    if(i==n){
        if (pa!=NULL){
            free(pa);
            pa=NULL;
        }
        if (fclose(fpassenger)!= 0){
            fprintf(log,"\nerr:close passenger fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:update status by pakey %d fail:no pakey in passenger!",pakey);
        fclose(log);
        return -1;
    }
}

//-1: 日期错误
//fprintf
int writetrain(char *trainname,int yy,int mm,int dd){
    FILE* ftrain;
    FILE* log;
    struct train* trn;
    int n;
    int date;

    trn=(struct train*)malloc(sizeof(TRAIN));
    log=fopen(".\\LOGGER","a+");

    if((ftrain=fopen(".\\DB\\TRAIN", "rb+" ))==NULL){
        printf("\nerr:open train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                 //???
    }

    fseek(ftrain,0,SEEK_END);
    fseek(ftrain,0,SEEK_END);
    n=ftell(ftrain)/sizeof(TRAIN);

    if((trn=(TRAIN*)malloc(sizeof(TRAIN)))==NULL){
        printf("\nerr:no space for train!");
        delay(3000);
        exit(1);
        return 0;
    }

    trn->trkey=n+1;
    strcpy(trn->trainname,trainname);
    //todo:加入判断trainname
    if(postdate(yy,mm,dd,&date)!=1){
        fprintf(log,"\ninfo:write train fail:wrong date!");

        if (fclose(ftrain)!=0) {
        fprintf(log,"\nerr:close train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
        }

        fclose(log);
        free(trn);
        return -1;
    }
    //todo:?
    trn->date=date;
    trn->status=0;

    fseek(ftrain,0,SEEK_END);
    fwrite(trn,sizeof(TRAIN),1,ftrain);
    fflush(ftrain);                         //todo

    if (trn != NULL)
    {
        free(trn);
        trn = NULL;
    }

    if (fclose(ftrain)!=0)                                                    //关闭文件
    {
        fprintf(log,"\nerr:close train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:write train trkey %d trainname %s date %d.%d.%d success!",n+1,trainname,yy,mm,dd);

    fclose(log);
    free(trn);
    return 1;
}

//-1: 没有该trkey 1: 成功获取
int gettrainbytrkey(int trkey,struct train *desttrn){
    FILE* ftrain;
    FILE* log;
    struct train *trn;
    int i;
    int n;

    trn=(struct train *)malloc(sizeof(TRAIN));
    log=fopen(".\\LOGGER","a+");

    if((ftrain=fopen(".\\DB\\TRAIN", "rb+" ))==NULL){
        fprintf(log,"\nerr:open train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    fseek(ftrain,0,SEEK_END);
    n=ftell(ftrain)/sizeof(TRAIN);

    for(i=0;i<n;i++){
        if ((trn=(TRAIN*)malloc(sizeof(TRAIN)))==NULL){
            fprintf(log,"\nerr:no space for train!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(ftrain,i*sizeof(TRAIN),SEEK_SET);
        fread(trn,sizeof(TRAIN),1,ftrain);

        if(trn->trkey==trkey){
            desttrn->trkey=trn->trkey;
            desttrn->date=trn->date;
            desttrn->status=trn->status;
            strcpy(desttrn->trainname,trn->trainname);

            if (trn!=NULL){
                free(trn);
                trn=NULL;
            }

            if (fclose(ftrain)!= 0){
                fprintf(log,"\nerr:close train fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            fprintf(log,"\ninfo:get train by trkey %d success!",trkey);

            fclose(log);
            free(trn);
            return 1;
        }
        if (trn!=NULL){
            free(trn);
            trn=NULL;
        }
    }

    if(i==n){
        if (trn!=NULL){
            free(trn);
            trn=NULL;
        }

        if (fclose(ftrain)!= 0){
            fprintf(log,"\nerr:close train fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fprintf(log,"\ninfo:get train by trkey %d fail:no trkey in train!",trkey);

        fclose(log);
        free(trn);
        return -1;
    }
}

//返回获得的个数
int gettrainbystatus(int status,struct pagetrain desttrn[20]){
    FILE* ftrain;
    FILE* log;
    struct train* trn;
    int i;
    int j=0;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((ftrain=fopen(".\\DB\\TRAIN", "rb+" ))==NULL){
        printf("\nerr:open train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                  //???
    }

    fseek(ftrain,0,SEEK_END);
    n=ftell(ftrain)/sizeof(TRAIN);

    for(i=0;i<n;i++){
        if ((trn=(TRAIN*)malloc(sizeof(TRAIN)))==NULL){
            fprintf(log,"\nerr:no space for train!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(ftrain,i*sizeof(TRAIN),SEEK_SET);
        fread(trn,sizeof(TRAIN),1,ftrain);

        if(trn->status==status){
            desttrn[j].date=trn->date;
            desttrn[j].status=trn->status;
            desttrn[j].trkey=trn->trkey;
            strcpy(desttrn[j].trainname,trn->trainname);

            j++;
            fprintf(log,"\ninfo:get train:trkey %d trainname %s date %d by status %d success!"
                       ,trn->trkey,trn->trainname,trn->date,trn->status);
        }
        if (trn!=NULL){
            free(trn);
            trn=NULL;
        }
    }

    if (fclose(ftrain)!= 0){
        fprintf(log,"\nerr:close train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return j;
}

int updatestatusbytrkey(int trkey,int status){
    FILE* ftrain;
    FILE* log;
    struct train *trn;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((ftrain=fopen(".\\DB\\TRAIN", "rb+" ))==NULL){
        fprintf(log,"\nerr:open train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(ftrain,0,SEEK_END);
    fseek(ftrain,0,SEEK_END);
    n=ftell(ftrain)/sizeof(TRAIN);

    for(i=0;i<n;i++){
        if ((trn=(TRAIN*)malloc(sizeof(TRAIN)))==NULL){
            fprintf(log,"\nerr:no space for train!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(ftrain,i*sizeof(TRAIN),SEEK_SET);
        fread(trn,sizeof(TRAIN),1,ftrain);

        if(trn->trkey==trkey){
            if(trn->status==status){
                fprintf(log,"\ninfo:update status %d by trkey %d fail:same status %d!",status,trkey,status);

                if (fclose(ftrain)!= 0){
                    fprintf(log,"\nerr:close user fail!");
                    delay(3000);
                    exit(1);

                    fclose(log);
                    return 0;
                }

                fclose(log);
                return -2;
            }

            trn->status=status;

            fseek(ftrain,i*sizeof(TRAIN),SEEK_SET);
            fwrite(trn,sizeof(TRAIN),1,ftrain);
            fflush(ftrain);

            if (trn!=NULL){
                free(trn);
                trn=NULL;
            }
            if (fclose(ftrain)!= 0){
                fprintf(log,"\nerr:close user fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            fprintf(log,"\ninfo:update status %d by trkey %d success!",status,trkey);
            fclose(log);
            return 1;
        }
        if (trn!=NULL){
            free(trn);
            trn=NULL;
        }
    }

    if(i==n){
        if (trn!=NULL){
            free(trn);
            trn=NULL;
        }
        if (fclose(ftrain)!= 0){
            fprintf(log,"\nerr:close train fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:update status by trkey %d fail:notrkey in train!",trkey);

        fclose(log);
        return -1;
    }
}

int getpostedtrkey(int date,char *trainname,int *trkey){
    FILE* ftrain;
    FILE* log;
    struct train *trn;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((ftrain=fopen(".\\DB\\TRAIN","rb+"))==NULL){
        fprintf(log,"\neer:open train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fseek(ftrain,0,SEEK_END);
    n=ftell(ftrain)/sizeof(TRAIN);

    for(i=0;i<n;i++){
        if((trn=(TRAIN*)malloc(sizeof(TRAIN)))==NULL){
            fprintf(log,"\nerr:no space for train!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }

        fseek(ftrain,i*sizeof(TRAIN),SEEK_SET);
        fread(trn,sizeof(TRAIN),1,ftrain);
        if((strcmp(trainname,trn->trainname)==0)&&(date==trn->date)){
            *trkey=trn->trkey;

            if(trn!= NULL){
                free(trn);
                trn= NULL;
            }

            if(fclose(ftrain) != 0){
                printf(log,"\nerr:close train fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

            fprintf(log,"\ninfo:get trkey %d by trainname %s date %d success!",*trkey,trainname,date);

            fclose(log);
            return 1;
        }

        if(trn!= NULL){
            free(trn);
            trn= NULL;
```

```c
            trn= NULL;
        }
    }

    if(fclose(ftrain)!= 0){
        printf(log,"\nerr:close train fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:get trkey by trainname %s date %d fail!",trainname,date);

    fclose(log);
    return -1;
}
```

6. file2.c

C

```c
#include"public.h"

//-1 ??
int writerecord(int pakey,int trkey){
    FILE* frecord;
    FILE* log;
    struct record* rcd;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((frecord=fopen(".\\DB\\RECORD", "rb+"))==NULL){
        fprintf(log,"\nerr:open record fail! %s",strerror(errno));
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    fseek(frecord,0,SEEK_END);
    n=ftell(frecord)/sizeof(RECORD);

    for(i=0;i<n;i++){
        if ((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL){
            fprintf(log,"\nerr:no space for record!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(frecord,i*sizeof(RECORD),SEEK_SET);
        fread(rcd,sizeof(RECORD),1,frecord);

        if(rcd->pakey==pakey&&rcd->trkey==trkey){
            if (rcd!=NULL){
                free(rcd);
                rcd=NULL;
            }

            if (fclose(frecord)!=0){
                printf("\nerr:close record fail!");
                delay(3000);
                exit(1);
                return 0;
            }

            fprintf(log,"\ninfo:write record pakey %d trkey %d fail:already wr
itten!",pakey,trkey);
            fclose(log);
            return -1;
        }

        if (rcd!=NULL){
            free(rcd);
            rcd=NULL;
        }
    }

    if((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL)
    {
        fprintf(log,"\nerr:no space for record!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    rcd->pakey=pakey;
    rcd->trkey=trkey;

    fseek(frecord,0,SEEK_END);
    fwrite(rcd,sizeof(RECORD),1,frecord);
    fflush(frecord);

    if (rcd != NULL)
    {
        free(rcd);
        rcd = NULL;
    }

    if (fclose(frecord)!=0)
    {
        printf("\nerr:close record fail!");
        delay(3000);
        exit(1);
        return 0;
    }

    fprintf(log,"\ninfo:write record pakey %d trkey %d success!",pakey,trkey);

    fclose(log);
    return 1;
}

int writerecordv1(int pakey,int trkey,char *track){
    FILE* frecord;
    FILE* log;
    struct record* rcd;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((frecord=fopen(".\\DB\\RECORD", "rb+"))==NULL){
        fprintf(log,"\nerr:open record fail! %s",strerror(errno));
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    fseek(frecord,0,SEEK_END);
    n=ftell(frecord)/sizeof(RECORD);

    for(i=0;i<n;i++){
        if ((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL){
            fprintf(log,"\nerr:no space for record!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(frecord,i*sizeof(RECORD),SEEK_SET);
        fread(rcd,sizeof(RECORD),1,frecord);

        if(rcd->pakey==pakey&&rcd->trkey==trkey){
            if (rcd!=NULL){
                free(rcd);
                rcd=NULL;
            }

            if (fclose(frecord)!=0){
                printf("\nerr:close record fail!");
                delay(3000);
                exit(1);
                return 0;
            }

            fprintf(log,"\ninfo:write record pakey %d trkey %d fail:already wr
itten!",pakey,trkey);
            fclose(log);
            return -1;
        }

        if (rcd!=NULL){
            free(rcd);
            rcd=NULL;
        }
    }

    if((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL)
    {
        fprintf(log,"\nerr:no space for record!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    memset(rcd->track,'\0',sizeof(rcd->track));
    rcd->pakey=pakey;
    rcd->trkey=trkey;
    strcpy(rcd->track,track);

    fseek(frecord,0,SEEK_END);
    fwrite(rcd,sizeof(RECORD),1,frecord);
    fflush(frecord);

    if (rcd != NULL)
    {
        free(rcd);
        rcd = NULL;
    }

    if (fclose(frecord)!=0)
    {
        printf("\nerr:close record fail!");
        delay(3000);
        exit(1);
        return 0;
    }

    fprintf(log,"\ninfo:write record pakey %d trkey %d track %s success!",pakey,trkey,track);

    fclose(log);
    return 1;
}

//>0 trkey个数  0 错误 -1 没有trkey
int getrecordbypakey(int pakey,int desttrkeyset[50]){
    FILE* frecord;
    FILE* log;
    struct record* rcd;
    int i;
    int n;
    int j=0;

    log=fopen(".\\LOGGER","a+");

    if((frecord=fopen(".\\DB\\RECORD", "rb+" ))==NULL){
        fprintf(log,"\nerr:open record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
        return 0;                                                   //???
    }

    fseek(frecord,0,SEEK_END);
    n=ftell(frecord)/sizeof(RECORD);

    for(i=0;i<n;i++){
        if ((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL){
            fprintf(log,"\nerr:no space for record!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(frecord,i*sizeof(RECORD),SEEK_SET);
        fread(rcd,sizeof(RECORD),1,frecord);

        if(rcd->pakey==pakey){
            desttrkeyset[j]=rcd->trkey;
            j++;

            if (rcd!=NULL){
                free(rcd);
                rcd=NULL;
            }
        }
        if (rcd!=NULL){
            free(rcd);
            rcd=NULL;
        }
    }

    if(j==0){
        if (fclose(frecord)!= 0){
            fprintf(log,"\nerr:close record fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:get record trkey set by pakey %d fail:no pakey in record!",pakey);

        fclose(log);
        return -1;
    }
```

```c

    if (fclose(frecord)!= 0){
        fprintf(log,"\nerr:close record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:get record trkey set by pakey %d success:",pakey);
    for(i=0;i<j;i++)fprintf(log," %d",desttrkeyset[i]);
    fprintf(log,"!");

    fclose(log);
    return j;
}

//>0 pakey个数  0 错误 -1 没有pakey
int getrecordbytrkey(int trkey,int destpakeyset[50]){
    FILE* frecord;
    FILE* log;
    struct record* rcd;
    int i;
    int n;
    int j=0;

    log=fopen(".\\LOGGER","a+");

    if((frecord=fopen(".\\DB\\RECORD", "rb+" ))==NULL){
        fprintf(log,"\nerr:open record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                          //???
    }

    fseek(frecord,0,SEEK_END);
    n=ftell(frecord)/sizeof(RECORD);


    for(i=0;i<n;i++){
        if ((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL){
            fprintf(log,"\nerr:no space for record!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(frecord,i*sizeof(RECORD),SEEK_SET);
        fread(rcd,sizeof(RECORD),1,frecord);

        if(rcd->trkey==trkey){
            destpakeyset[j]=rcd->pakey;
            j++;

            if (rcd!=NULL){
                free(rcd);
                rcd=NULL;

            }
        }
        if (rcd!=NULL){
            free(rcd);
            rcd=NULL;
        }
    }

    if(j==0){
        if (fclose(frecord)!= 0){
            fprintf(log,"\nerr:close record fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:get record pakey set by trkey %d fail:no pakey in record!",trkey);

        fclose(log);
        return -1;
    }

    if (fclose(frecord)!= 0){
        fprintf(log,"\nerr:close record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:get record pakey set by trkey %d success:",trkey);
    for(i=0;i<j;i++)fprintf(log," %d",destpakeyset[i]);
    fprintf(log,"!");

    fclose(log);
    return j;
}

int gettrackbypakey(int pakey,char trackset[10][10]){
    FILE* frecord;
    FILE* log;
    struct record* rcd;
    int trackcount=0;
    int i;
    int n;

    log=fopen(".\\LOGGER","a+");

    if((frecord=fopen(".\\DB\\RECORD", "rb+" ))==NULL){
        fprintf(log,"\nerr:open record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;                                                   //???
    }

    fseek(frecord,0,SEEK_END);
    n=ftell(frecord)/sizeof(RECORD);

    for(i=0;i<n;i++){
        if ((rcd=(RECORD*)malloc(sizeof(RECORD)))==NULL){
            fprintf(log,"\nerr:no space for record!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fseek(frecord,i*sizeof(RECORD),SEEK_SET);
        fread(rcd,sizeof(RECORD),1,frecord);

        if(rcd->pakey==pakey){
            strcpy(trackset[trackcount],rcd->track);
            trackcount++;

            if (rcd!=NULL){
                free(rcd);
                rcd=NULL;
            }
        }
        if (rcd!=NULL){
            free(rcd);
            rcd=NULL;
        }
    }

    if(trackcount==0){
        if (fclose(frecord)!= 0){
            fprintf(log,"\nerr:close record fail!");
            delay(3000);
            exit(1);

            fclose(log);
            return 0;
        }
        fprintf(log,"\ninfo:get record track set by pakey %d fail:no track in record!",pakey);

        fclose(log);
        return -1;
    }

    if (fclose(frecord)!= 0){
        fprintf(log,"\nerr:close record fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fprintf(log,"\ninfo:get track set by pakey %d success:",pakey);
    for(i=0;i<trackcount;i++)fprintf(log," %s",trackset[i]);
    fprintf(log,"!");

    fclose(log);
    return trackcount;
}
```

7. filetxt.c

C

```
#include"public.h"
```

```c
#include"public.h"

int getcnamebyID(char *ID,char *cname){
    FILE *fcnametxt;
    FILE *log;
    char tempID[20];

    log=fopen(".\\LOGGER","a+");

    if((fcnametxt=fopen(".\\DB\\cname.txt", "r+" ))==NULL){
        fprintf(log,"\nerr:open cname.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    while(!feof(fcnametxt)){
        memset(tempID,'\0',sizeof(ID));
        memset(cname,'\0',sizeof(cname));
        fscanf(fcnametxt,"%s",tempID);
        fgetc(fcnametxt);
        fscanf(fcnametxt,"%s",cname);
        fgetc(fcnametxt);
        if(strcmp(tempID,ID)==0){
            if (fclose(fcnametxt)!=0){
                fprintf(log,"\nerr:close cname.txt fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

        fclose(log);
        return 1;
        }
    }

    if (fclose(fcnametxt)!=0){
        fprintf(log,"\nerr:close cname.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return 0;
}

int getstartbytrainname(char *trainname,char *start){
    FILE *fstarttxt;
    FILE *log;
    char temptrainname[10];

    log=fopen(".\\LOGGER","a+");

    if((fstarttxt=fopen(".\\DB\\start.txt", "r+" ))==NULL){
        fprintf(log,"\nerr:open start.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    while(!feof(fstarttxt)){
        memset(start,'\0',sizeof(start));
        memset(temptrainname,'\0',sizeof(temptrainname));
        fscanf(fstarttxt,"%s",temptrainname);
        fgetc(fstarttxt);
        fscanf(fstarttxt,"%s",start);
        fgetc(fstarttxt);
        if(strcmp(temptrainname,trainname)==0){
            if (fclose(fstarttxt)!=0){
                fprintf(log,"\nerr:close start.txt fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

        fclose(log);
        return 1;
        }
    }

    if (fclose(fstarttxt)!=0){
        fprintf(log,"\nerr:close start.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return -1;
}

int getendbytrainname(char *trainname,char *end){
    FILE *fendtxt;
    FILE *log;
    char temptrainname[10];

    log=fopen(".\\LOGGER","a+");

    if((fendtxt=fopen(".\\DB\\end.txt", "r+" ))==NULL){
        fprintf(log,"\nerr:open end.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    while(!feof(fendtxt)){
        memset(end,'\0',sizeof(end));
        memset(temptrainname,'\0',sizeof(temptrainname));
        fscanf(fendtxt,"%s",temptrainname);
        fgetc(fendtxt);
        fscanf(fendtxt,"%s",end);
        fgetc(fendtxt);
        if(strcmp(temptrainname,trainname)==0){
            if (fclose(fendtxt)!=0){
                fprintf(log,"\nerr:close end.txt fail!");
                delay(3000);
                exit(1);

                fclose(log);
                return 0;
            }

        fclose(log);
        return 1;
        }
    }

    if (fclose(fendtxt)!=0){
        fprintf(log,"\nerr:close end.txt fail!");
        delay(3000);
        exit(1);

        fclose(log);
        return 0;
    }

    fclose(log);
    return -1;
}
```

8. graphpro.c

```C
#include <graphics.h>
#include "hz.h"
#include"public.h"
#include<stdio.h>
#include<stdlib.h>

void drawtop(void){
    setfillstyle(1,BLUE);
    bar(0,0,640,40);
    puthz(160,4,"公共交通行程管理系统",32,32,LIGHTGRAY);

    setcolor(RED);
    setfillstyle(1,RED);
    pieslice(560,18,315,360,13);
    pieslice(560,18,0,225,13);
    bar(557,3,563,8);
    setcolor(BLUE);
    setfillstyle(1,BLUE);
    pieslice(560,18,315,360,10);
    pieslice(560,18,0,225,10);
    setcolor(RED);
    setfillstyle(1,RED);
    bar(553,16,567,20);
    bar(559,20,561,30);
    bar(548,30,572,32);

    return;
}

void barword16(int left,int top,int barcolor,int width,int n,char *s,int wordc
olor){
```

```c
    int right=width*16+4+left;
    int bottom=20+top;
    float m=n/2.0;
    float middle=(left+right)/2.0;

    if(left==0){
        setfillstyle(1,barcolor);
        bar(320-8*width-2,top,320+8*width+2,bottom);
        puthz(320-16*m,top+2,s,16,16,wordcolor);
        return;
    }//位置居中

    setfillstyle(1,barcolor);
    bar(left,top,right,bottom);
    puthz(middle-16*m,top+2,s,16,16,wordcolor);
    return;
}
void barword24(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*24+6+left;
    int bottom=30+top;
    float m=n/2.0;
    float middle=(left+right)/2.0;

    if(left==0){
        setfillstyle(1,barcolor);
        bar(320-12*width-3,top,320+12*width+3,bottom);
        puthz(320-12*m,top+3,s,24,24,wordcolor);
        return;
    }//位置居中

    setfillstyle(1,barcolor);
    bar(left,top,right,bottom);
    puthz(middle-24*m,top+3,s,24,24,wordcolor);
    return;
}
void barword32(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*32+8+left;
    int bottom=40+top;
    float m=n/2.0;
    float middle=(left+right)/2.0;

    if(left==0){
        setfillstyle(1,barcolor);
        bar(320-16*width-4,top,320+16*width+4,bottom);
        puthz(320-32*m,top+4,s,32,32,wordcolor);
        return;
    }//位置居中

    setfillstyle(1,barcolor);
    bar(left,top,right,bottom);
    puthz(middle-32*m,top+4,s,32,32,wordcolor);
    return;
}
void barword48(int left,int top,int barcolor,int width,int n,char *s,int wordcolor){
    int right=width*48+6+left;
    int bottom=60+top;
    float m=n/2.0;
    float middle=(left+right)/2.0;

    if(left==0){
        setfillstyle(1,barcolor);
        bar(320-24*width-6,top,320+24*width+6,bottom);
        puthz(320-48*m,top+6,s,48,48,wordcolor);
        return;
    }//位置居中

    setfillstyle(1,barcolor);
    bar(left,top,right,bottom);
    puthz(middle-48*m,top+6,s,48,48,wordcolor);
    return;
}

void barwordframe(int left,int top,int barcolor,int width,char *s,int wordcolor,int framecolor){
    int right=width*32+8+left;
    int bottom=40+top;
    float m=width/2.0;
    int framewidth=1;

    setfillstyle(1,framecolor);
    bar(left,top,right,bottom);

    setfillstyle(1,barcolor);
    bar(left+framewidth,top+framewidth,right-framewidth,bottom-framewidth);

    puthz(left+4,top+4,s,32,32,wordcolor);
    return;
}

void returnbutton(int color){//返回按钮
    int x1=10,x2=30,x3=30,y1=20,y2=10,y3=30;
    setcolor(color);

    setlinestyle(SOLID_LINE,0,3);
    line(x1,y1,x2,y2);
    line(x1,y1,x3,y3);

    return;
}
void searchbutton(int color){
    setcolor(color);
    setfillstyle(1,BLUE);
    setlinestyle(0,0,3);
    bar(510,100,550,140);
    circle(535,115,10);
    line(515,132,527,120);
}

void loginok(void){
    barword32(0,200,WHITE,5,4,"登录成功",GREEN);
    return;
}
void loginfail(void){
    barword32(0,200,WHITE,5,4,"登录失败",RED);
    return;
}
void loginpasswordwrong(void){
    barword32(0,200,WHITE,5,4,"密码错误",RED);
}
void loginnoregister(void){
    barword32(0,200,WHITE,7,6,"该账号未注册",RED);
}

void registerok(void){
    barword32(0,225,BLUE,4,4,"注册成功",GREEN);
    return;
}
void registerrepeat(void){
    barword32(0,225,BLUE,5,5,"账号已注册",RED);
    return;
}
void registerpasswrong(void){
    barword32(0,225,BLUE,6,6,"两次密码不同",RED);
    return;
}
void registerpasszero(void){
    barword32(0,225,BLUE,6,6,"密码不得为空",RED);
    return;
}
void registernamezero(void){
    barword32(0,225,BLUE,6,6,"账号不得为空",RED);
    return;
}
void registerpassshort(void){
    barword32(0,225,BLUE,10,10,"密码长度不得少于六位",RED);
    return;
}

void bindok(void){
    barword32(0,225,BLUE,4,4,"绑定成功",GREEN);
    return;
}
void bindIDrepeat(void){
    barword32(0,225,BLUE,7,7,"此身份证已绑定",RED);
    return;
}
void bindIDwrong(void){
    barword32(0,225,BLUE,7,7,"身份证号不合法",RED);
    return;
}
void bindtelwrong(void){
    barword32(0,225,BLUE,7,7,"电话号码不合法",RED);
    return;
}

void labelok(void){
    barword32(0,225,BLUE,4,4,"录入成功",GREEN);
    return;
}
void labelIDzero(void){
    barword32(0,225,BLUE,8,8,"后台无该身份证号",RED);
    return;
}

void statusabnormal(void){
    barword32(0,225,BLUE,6,6,"健康状态异常",RED);
    return;
}

void recordzero(void){
    barword32(0,225,BLUE,4,4,"无该车次",RED);
    return;
}

void trackrecordrepeat(void){
    barword32(0,225,BLUE,4,4,"重复录入",RED);
    return;
```

```
}

void trackrecordok(void){
    barword32(0,225,BLUE,4,4,"录入成功",RED);
    return;
}

void palabelu(struct passenger *pa){
    int x1=60;
    int y1=130;
    int x2=x1+520;
    int y2=80+y1;
    int x;
    char cname[5];
    getcnamebyID(pa->ID,cname);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"姓名",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"状态",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"电话",16,16,LIGHTGRAY);
    puthz(x1+180,y1+50,"身份证号",16,16,LIGHTGRAY);

    puthz(x1+80,y1+10,cname,16,16,LIGHTGRAY);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+260,y1+6,pa->tel);
    outtextxy(x1+260,y1+44,pa->ID);

    x=pa->status;
    if(x==0)puthz(x1+80,y1+50,"健康",16,16,GREEN);
    else if(x==1)puthz(x1+80,y1+50,"新冠肺炎患者",16,16,RED);
    else if(x==2)puthz(x1+80,y1+50,"无症状感染者",16,16,RED);
    else if(x==3)puthz(x1+80,y1+50,"密切接触者",16,16,YELLOW);
}
void palabelm(struct passenger *pa){
    int x1=60;
    int y1=230;
    int x2=x1+520;
    int y2=80+y1;
    int x;
    char cname[5];
    getcnamebyID(pa->ID,cname);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"姓名",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"状态",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"电话",16,16,LIGHTGRAY);
    puthz(x1+180,y1+50,"身份证号",16,16,LIGHTGRAY);

    puthz(x1+80,y1+10,cname,16,16,LIGHTGRAY);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+260,y1+6,pa->tel);
    outtextxy(x1+260,y1+44,pa->ID);
    x=pa->status;

    if(x==0)puthz(x1+80,y1+50,"健康",16,16,GREEN);
    else if(x==1)puthz(x1+80,y1+50,"新冠肺炎患者",16,16,RED);
    else if(x==2)puthz(x1+80,y1+50,"无症状感染者",16,16,RED);
    else if(x==3)puthz(x1+80,y1+50,"密切接触者",16,16,YELLOW);
}
void palabell(struct passenger *pa){
    int x1=60;
    int y1=330;
    int x2=x1+520;
    int y2=80+y1;
    int x;
    char cname[5];
    getcnamebyID(pa->ID,cname);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"姓名",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"状态",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"电话",16,16,LIGHTGRAY);
    puthz(x1+180,y1+50,"身份证号",16,16,LIGHTGRAY);

    puthz(x1+80,y1+10,cname,16,16,LIGHTGRAY);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+260,y1+6,pa->tel);
    outtextxy(x1+260,y1+44,pa->ID);

    x=pa->status;
    if(x==0)puthz(x1+80,y1+50,"健康",16,16,GREEN);
    else if(x==1)puthz(x1+80,y1+50,"新冠肺炎患者",16,16,RED);
    else if(x==2)puthz(x1+80,y1+50,"无症状感染者",16,16,RED);
    else if(x==3)puthz(x1+80,y1+50,"密切接触者",16,16,YELLOW);
}

void trlabelu(struct pagetrain *trn){
    int x1=60;
    int y1=130;
    int x2=x1+520;
```

```c
    int x2=x1+520;
    int y2=80+y1;
    int yy,mm,dd;
    char start[5];
    char end[5];
    getstartbytrainname(trn->trainname,start);
    getendbytrainname(trn->trainname,end);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"车次",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"日期",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"始发",16,16,LIGHTGRAY);
    puthz(x1+330,y1+10,"终点",16,16,LIGHTGRAY);
    puthz(x1+330,y1+50,"状态",16,16,LIGHTGRAY);

    getdate(trn->date,&yy,&mm,&dd);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+100,y1+6,trn->trainname);
    puthz(x1+220,y1+10,start,16,16,LIGHTGRAY);
    puthz(x1+370,y1+10,end,16,16,LIGHTGRAY);
    outtextxy(x1+100,y1+46,datestring(trn->date));
    if(trn->status==0){
        puthz(x1+370,y1+50,"正常",16,16,GREEN);
    }
    else if(trn->status==1){
        puthz(x1+370,y1+50,"异常",16,16,RED);
    }
}
void trlabelm(struct pagetrain *trn){
    int x1=60;
    int y1=230;
    int x2=x1+520;
    int y2=80+y1;
    int yy,mm,dd;
    char start[5];
    char end[5];
    getstartbytrainname(trn->trainname,start);
    getendbytrainname(trn->trainname,end);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"车次",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"日期",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"始发",16,16,LIGHTGRAY);
    puthz(x1+330,y1+10,"终点",16,16,LIGHTGRAY);
    puthz(x1+330,y1+50,"状态",16,16,LIGHTGRAY);

    getdate(trn->date,&yy,&mm,&dd);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+100,y1+6,trn->trainname);
    puthz(x1+220,y1+10,start,16,16,LIGHTGRAY);
    puthz(x1+370,y1+10,end,16,16,LIGHTGRAY);
    outtextxy(x1+100,y1+46,datestring(trn->date));
    if(trn->status==0){
        puthz(x1+370,y1+50,"正常",16,16,GREEN);
    }
    else if(trn->status==1){
        puthz(x1+370,y1+50,"异常",16,16,RED);
    }
}
void trlabell(struct pagetrain *trn){
    int x1=60;
    int y1=330;
    int x2=x1+520;
    int y2=80+y1;
    int yy,mm,dd;
    char start[5];
    char end[5];
    getstartbytrainname(trn->trainname,start);
    getendbytrainname(trn->trainname,end);

    setfillstyle(1,BLUE);
    bar(x1,y1,x2,y2);
    puthz(x1+20,y1+10,"车次",16,16,LIGHTGRAY);
    puthz(x1+20,y1+50,"日期",16,16,LIGHTGRAY);
    puthz(x1+180,y1+10,"始发",16,16,LIGHTGRAY);
    puthz(x1+330,y1+10,"终点",16,16,LIGHTGRAY);
    puthz(x1+330,y1+50,"状态",16,16,LIGHTGRAY);

    getdate(trn->date,&yy,&mm,&dd);
    setcolor(LIGHTGRAY);
    settextstyle(1,0,2);
    outtextxy(x1+100,y1+6,trn->trainname);
    puthz(x1+220,y1+10,start,16,16,LIGHTGRAY);
    puthz(x1+370,y1+10,end,16,16,LIGHTGRAY);
    outtextxy(x1+100,y1+46,datestring(trn->date));
    if(trn->status==0){
        puthz(x1+370,y1+50,"正常",16,16,GREEN);
    }
    else if(trn->status==1){
        puthz(x1+370,y1+50,"异常",16,16,RED);
    }
}
```

```c

void leftarrow(int color){
    setcolor(color);
    line(270,450,270,430);
    line(270,430,250,440);
    line(270,450,250,440);
}
void rightarrow(int color){
    setcolor(color);
    line(370,430,370,450);
    line(370,430,390,440);
    line(370,450,390,440);
}

void pagenumber(int currentpage,int countpage){
    char currentstr[3];
    char countstr[3];

    currentstr[0]=currentpage+'0';
    currentstr[1]='\0';
    countstr[0]=countpage+'0';
    countstr[1]='\0';

    settextstyle(1,0,3);
    setcolor(BLUE);
    outtextxy(320-22,425,currentstr);
    outtextxy(320+14,425,countstr);
    setlinestyle(SOLID_LINE,0,3);
    line(325,430,320-5,450);
}

void putnum(int x,int y,int num){
    char *numstring;
    itoa(num,numstring,10);
    outtextxy(x,y,numstring);
    free(numstring);
    return;
}

void todaydate(void){
    int yy;
    int mm;
    int dd;
    char Y[5]={'\0'};
    char M[5]={'\0'};
    char D[5]={'\0'};
    puthz(350,370,"乘车日期为",16,16,RED);
    puthz(500,370,"年",16,16,RED);
    puthz(500,370,"年",16,16,RED);
    puthz(543,370,"月",16,16,RED);
    puthz(586,370,"日",16,16,RED);
    gettodaydate(&yy,&mm,&dd);
    setcolor(RED);
    settextstyle(1, 0, 1);

    itoa(yy,Y,10);
    itoa(mm,M,10);
    itoa(dd,D,10);
    outtextxy(443,365,Y);
    outtextxy(525,365,M);
    outtextxy(560,365,D);

    return;
}

void position1(int x,int y,int *s){              //名字在上角
    setcolor(GREEN);
    setfillstyle(1,GREEN);
    pieslice(x,y,0,360,3);
    setcolor(GREEN);
    line(x,y,x+3,y);
    puthz(x,y-25,s,16,16,RED);
}

void position2(int x,int y,int *s){               //名字在右下角
    setcolor(GREEN);
    setfillstyle(1,GREEN);
    pieslice(x,y,0,360,3);
    setcolor(GREEN);
    line(x,y,x+3,y);
    puthz(x,y+15,s,16,16,RED);
}

void drawmap(void){
    position1(425,120,"北京");
    position1(325,165,"太原");
    position1(155,210,"西安");
    position1(55,270,"成都");
    position1(95,310,"重庆");
    position1(305,270,"武汉");
    position1(465,240,"南京");
    position2(485,280,"杭州");
    position1(505,250,"苏州");
    position2(565,252,"上海");
    position1(209,420,"广州");
    position1(229,440,"深圳");
```

```c
    position1(385,330,"南昌");
    position1(200,350,"长沙");
}

void mapline(char *s,int color){
    setlinestyle(0,1,3);
    setcolor(color);
    do {
        line(xpos(*s),ypos(*s),xpos(*(s+1)),ypos(*(s+1)));
        s++;
    }while(*(s+1)!='\0');
    return;
}

void track1(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(110,190,BLUE,6,6,"北京》》太原",color);
    if(strcmp(trainname,"G567")==0)barword24(110,190,BLUE,6,6,"上海》》杭州",color);
    if(strcmp(trainname,"G751")==0)barword24(110,190,BLUE,6,6,"南昌》》武汉",color);
    if(strcmp(trainname,"G768")==0)barword24(110,190,BLUE,6,6,"广州》》重庆",color);
    if(strcmp(trainname,"G267")==0)barword24(110,190,BLUE,6,6,"深圳》》广州",color);
    if(strcmp(trainname,"G186")==0)barword24(110,190,BLUE,6,6,"南京》》武汉",color);
    if(strcmp(trainname,"G379")==0)barword24(110,190,BLUE,6,6,"杭州》》南昌",color);
    if(strcmp(trainname,"G467")==0)barword24(110,190,BLUE,6,6,"西安》》武汉",color);
    if(strcmp(trainname,"G685")==0)barword24(110,190,BLUE,6,6,"苏州》》南昌",color);
    if(strcmp(trainname,"G335")==0)barword24(110,190,BLUE,6,6,"北京》》武汉",color);
}
void track2(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(110,246,BLUE,6,6,"北京》》南京",color);
    if(strcmp(trainname,"G567")==0)barword24(110,246,BLUE,6,6,"上海》》南昌",color);
    if(strcmp(trainname,"G751")==0)barword24(110,246,BLUE,6,6,"南昌》》南京",color);
    if(strcmp(trainname,"G768")==0)barword24(110,246,BLUE,6,6,"广州》》成都",color);
    if(strcmp(trainname,"G267")==0)barword24(110,246,BLUE,6,6,"深圳》》长沙",color);
    if(strcmp(trainname,"G186")==0)barword24(110,246,BLUE,6,6,"南京》》长沙",col
or);
    if(strcmp(trainname,"G379")==0)barword24(110,246,BLUE,6,6,"杭州》》武汉",color);
    if(strcmp(trainname,"G467")==0)barword24(110,246,BLUE,6,6,"西安》》南昌",color);
    if(strcmp(trainname,"G685")==0)barword24(110,246,BLUE,6,6,"苏州》》北京",color);
    if(strcmp(trainname,"G335")==0)barword24(110,246,BLUE,6,6,"北京》》长沙",color);
}
void track3(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(110,302,BLUE,6,6,"北京》》苏州",color);
    if(strcmp(trainname,"G567")==0)barword24(110,302,BLUE,6,6,"上海》》深圳",color);
    if(strcmp(trainname,"G751")==0)barword24(110,302,BLUE,6,6,"南昌》》太原",color);
    if(strcmp(trainname,"G768")==0)barword24(110,302,BLUE,6,6,"广州》》西安",color);
    if(strcmp(trainname,"G267")==0)barword24(110,302,BLUE,6,6,"深圳》》武汉",color);
    if(strcmp(trainname,"G186")==0)barword24(110,302,BLUE,6,6,"南京》》重庆",color);
    if(strcmp(trainname,"G379")==0)barword24(110,302,BLUE,6,6,"杭州》》西安",color);
    if(strcmp(trainname,"G467")==0)barword24(110,302,BLUE,6,6,"西安》》杭州",color);
    if(strcmp(trainname,"G685")==0)barword24(110,302,BLUE,6,6,"苏州》》太原",color);
    if(strcmp(trainname,"G335")==0)barword24(110,302,BLUE,6,6,"北京》》广州",color);
}
void track4(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(110,358,BLUE,6,6,"北京》》上海",color);
    if(strcmp(trainname,"G567")==0)barword24(110,358,BLUE,6,6,"上海》》广州",color);
    if(strcmp(trainname,"G751")==0)barword24(110,358,BLUE,6,6,"南昌》》北京",color);
    if(strcmp(trainname,"G768")==0)barword24(110,358,BLUE,6,6,"广州》》太原",color);
    if(strcmp(trainname,"G267")==0)barword24(110,358,BLUE,6,6,"深圳》》北京",color);
    if(strcmp(trainname,"G186")==0)barword24(110,358,BLUE,6,6,"南京》》成都",color);
    if(strcmp(trainname,"G379")==0)barword24(110,358,BLUE,6,6,"杭州》》成都",color);
    if(strcmp(trainname,"G467")==0)barword24(110,358,BLUE,6,6,"西安》》上海",color);
    if(strcmp(trainname,"G685")==0)barword24(110,358,BLUE,6,6,"苏州》》西安",color);
    if(strcmp(trainname,"G335")==0)barword24(110,358,BLUE,6,6,"北京》》深圳",color);
}
void track5(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(110,414,BLUE,6,6,"太原》》南京",color);
    if(strcmp(trainname,"G567")==0)barword24(110,414,BLUE,6,6,"杭州》》南昌",color);
    if(strcmp(trainname,"G751")==0)barword24(110,414,BLUE,6,6,"武汉》》南京",color);
    if(strcmp(trainname,"G768")==0)barword24(110,414,BLUE,6,6,"重庆》》成都",color);
    if(strcmp(trainname,"G267")==0)barword24(110,414,BLUE,6,6,"广州》》长沙",color);
    if(strcmp(trainname,"G186")==0)barword24(110,414,BLUE,6,6,"武汉》》长沙",color);
    if(strcmp(trainname,"G379")==0)barword24(110,414,BLUE,6,6,"南昌》》武汉",color);
    if(strcmp(trainname,"G467")==0)barword24(110,414,BLUE,6,6,"武汉》》南昌",color);
    if(strcmp(trainname,"G685")==0)barword24(110,414,BLUE,6,6,"南昌》》北京",color);
    if(strcmp(trainname,"G335")==0)barword24(110,414,BLUE,6,6,"武汉》》长沙",color);
}
void track6(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(380,190,BLUE,6,6,"太原》》苏州",color);
    if(strcmp(trainname,"G567")==0)barword24(380,190,BLUE,6,6,"杭州》》深圳",color);
    if(strcmp(trainname,"G751")==0)barword24(380,190,BLUE,6,6,"武汉》》太原",color);
    if(strcmp(trainname,"G768")==0)barword24(380,190,BLUE,6,6,"重庆》》西安",color);
    if(strcmp(trainname,"G267")==0)barword24(380,190,BLUE,6,6,"广州》》武汉",color);
    if(strcmp(trainname,"G186")==0)barword24(380,190,BLUE,6,6,"武汉》》重庆",color);
    if(strcmp(trainname,"G379")==0)barword24(380,190,BLUE,6,6,"南昌》》西安",color);
    if(strcmp(trainname,"G467")==0)barword24(380,190,BLUE,6,6,"武汉》》杭州",color);
    if(strcmp(trainname,"G685")==0)barword24(380,190,BLUE,6,6,"南昌》》太原",color);
    if(strcmp(trainname,"G335")==0)barword24(380,190,BLUE,6,6,"武汉》》广州",color);
}
void track7(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(380,246,BLUE,6,6,"太原》》上海",color);
    if(strcmp(trainname,"G567")==0)barword24(380,246,BLUE,6,6,"杭州》》广西",color);
    if(strcmp(trainname,"G751")==0)barword24(380,246,BLUE,6,6,"武汉》》北京",color);
    if(strcmp(trainname,"G768")==0)barword24(380,246,BLUE,6,6,"重庆》》太原",color);
    if(strcmp(trainname,"G267")==0)barword24(380,246,BLUE,6,6,"广州》》北京",color);
    if(strcmp(trainname,"G186")==0)barword24(380,246,BLUE,6,6,"武汉》》成都",color);
    if(strcmp(trainname,"G379")==0)barword24(380,246,BLUE,6,6,"南昌》》成都",color);
    if(strcmp(trainname,"G467")==0)barword24(380,246,BLUE,6,6,"武汉》》上海",color);
    if(strcmp(trainname,"G685")==0)barword24(380,246,BLUE,6,6,"南昌》》西安",color);
    if(strcmp(trainname,"G335")==0)barword24(380,246,BLUE,6,6,"武汉》》深圳",color);

}
void track8(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(380,302,BLUE,6,6,"南京》》苏州",color);
    if(strcmp(trainname,"G567")==0)barword24(380,302,BLUE,6,6,"南昌》》深圳",color);
    if(strcmp(trainname,"G751")==0)barword24(380,302,BLUE,6,6,"南京》》太原",color);
    if(strcmp(trainname,"G768")==0)barword24(380,302,BLUE,6,6,"成都》》西安",color);
    if(strcmp(trainname,"G267")==0)barword24(380,302,BLUE,6,6,"长沙》》武汉",color);
    if(strcmp(trainname,"G186")==0)barword24(380,302,BLUE,6,6,"长沙》》重庆",color);
    if(strcmp(trainname,"G379")==0)barword24(380,302,BLUE,6,6,"武汉》》西安",color);
    if(strcmp(trainname,"G467")==0)barword24(380,302,BLUE,6,6,"南昌》》杭州",color);
    if(strcmp(trainname,"G685")==0)barword24(380,302,BLUE,6,6,"北京》》太原",color);
    if(strcmp(trainname,"G335")==0)barword24(380,302,BLUE,6,6,"长沙》》广州",color);

```

```
}
void track9(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(380,358,BLUE,6,6,"南京》》上海",color);
    if(strcmp(trainname,"G567")==0)barword24(380,358,BLUE,6,6,"南昌》》广州",color);
    if(strcmp(trainname,"G751")==0)barword24(380,358,BLUE,6,6,"南京》》北京",color);
    if(strcmp(trainname,"G768")==0)barword24(380,358,BLUE,6,6,"成都》》太原",color);
    if(strcmp(trainname,"G267")==0)barword24(380,358,BLUE,6,6,"长沙》》北京",color);
    if(strcmp(trainname,"G186")==0)barword24(380,358,BLUE,6,6,"长沙》》成都",color);
    if(strcmp(trainname,"G379")==0)barword24(380,358,BLUE,6,6,"武汉》》成都",color);
    if(strcmp(trainname,"G467")==0)barword24(380,358,BLUE,6,6,"南昌》》上海",color);
    if(strcmp(trainname,"G685")==0)barword24(380,358,BLUE,6,6,"北京》》西安",color);
    if(strcmp(trainname,"G335")==0)barword24(380,358,BLUE,6,6,"长沙》》深圳",color);
}
void track10(char *trainname,int color){
    if(strcmp(trainname,"G562")==0)barword24(380,414,BLUE,6,6,"苏州》》上海",color);
    if(strcmp(trainname,"G567")==0)barword24(380,414,BLUE,6,6,"深圳》》广州",color);
    if(strcmp(trainname,"G751")==0)barword24(380,414,BLUE,6,6,"太原》》北京",color);
    if(strcmp(trainname,"G768")==0)barword24(380,414,BLUE,6,6,"西安》》太原",color);
    if(strcmp(trainname,"G267")==0)barword24(380,414,BLUE,6,6,"武汉》》北京",color);
    if(strcmp(trainname,"G186")==0)barword24(380,414,BLUE,6,6,"重庆》》成都",color);
    if(strcmp(trainname,"G379")==0)barword24(380,414,BLUE,6,6,"西安》》成都",color);
    if(strcmp(trainname,"G467")==0)barword24(380,414,BLUE,6,6,"杭州》》上海",color);
    if(strcmp(trainname,"G685")==0)barword24(380,414,BLUE,6,6,"太原》》西安",color);
    if(strcmp(trainname,"G335")==0)barword24(380,414,BLUE,6,6,"广州》》深圳",color);
}
```

9. page2.c

C

```c
#include<public.h>

void pagestart(int *page){
    int flag=0;                 //1为点亮乘客登录，2为点亮管理员登录
    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawstart();
    while (1)
    {

        newmouse(&MouseX,&MouseY,&press);
        if (inbarword32(0,200,BLUE,5,4,"乘客登录",LIGHTGRAY)){
            if(pressbarword32(0,200,BLUE,5,4,"乘客登录",LIGHTGRAY)==2){
                MouseS=1;
                if(flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,200,BLUE,5,4,"乘客登录",RED);
                    flag=1;
                }
                continue;
            }
            if(pressbarword32(0,200,BLUE,5,4,"乘客登录",LIGHTGRAY)==1){
                barword32(0,200,BLUE,5,4,"乘客登录",RED);
                MouseS=0;
                *page=1;
                break;
            }
        }

        if (inbarword32(0,300,BLUE,5,5,"管理员登录",LIGHTGRAY)){
            if(pressbarword32(0,300,BLUE,5,5,"管理员登录",LIGHTGRAY)==2){
                MouseS=1;
                if (flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,300,BLUE,5,5,"管理员登录",RED);
                    flag=2;
                }
```

```
                    continue;
                }
                if(pressbarword32(0,300,BLUE,5,5,"管理员登录",LIGHTGRAY)==1)
                {
                    barword32(0,300,BLUE,5,5,"管理员登录",RED);
                    MouseS=0;
                    *page=2;
                    break;
                }
            }
            if(flag){
                if(flag==1){
                    clrmous(MouseX,MouseY);
                    barword32(0,200,BLUE,5,4,"乘客登录",LIGHTGRAY);
                }
                if(flag==2){
                    clrmous(MouseX,MouseY);
                    barword32(0,300,BLUE,5,5,"管理员登录",LIGHTGRAY);
                }
                flag=0;
            }
            if(MouseS!=0){
                MouseS=0;
            }
        }
        return;
}

void pagepalogin(int *page,int *pakey){
    int pos=0;
    int flag=0;
    char name[20]={'\0'};
    char password[20]={'\0'};
    struct passenger para;

    *pakey=0;

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpalogin();

    while(1){
        newmouse(&MouseX,&MouseY,&press);

        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=0;
                break;
            }
        }
        if(inbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag=1;
                }
                continue;
            }
            else if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                name[0]='\0';
                barwordframe(180,170,WHITE,11,"",DARKGRAY,RED);
                Input_Vis(name,180,170,18,WHITE,BLACK);         //可视输入
                if(strlen(name)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag==1;
                }
                continue;
            }
            else if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                MouseS=0;
                password[0]='\0';
                barwordframe(180,250,WHITE,11,"",DARKGRAY,RED);
                Input_Invis(password,180,250,18,WHITE,BLACK);   //不可视输入
                if(strlen(password)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)){
            if(pressbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)==2){
                MouseS=1;
                if (flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,350,BLUE,3,2,"登录",RED);
                    flag=1;
                }
                continue;
            }
            if(pressbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)==1){
                MouseS=0;
                if((*pakey=loginuser(name,password))>0){
                    loginok();
                    delay(300);
                    if(getpassengerbypakey(*pakey,&para)==-1){
                        *page=4;
                        break;
                    }
                    *page=5;
                    break;
                }
                else if(*pakey==-1){
                    loginpasswordwrong();
                    delay(300);
                    *page=1;
                    break;
                }
                else if(*pakey==-2){
                    loginnoregister();
                    delay(300);
                    *page=1;
                    break;
                }
```

```c
                else{
                    exit(1);             //todo
                }
            }
        }
        if(inbarword16(450,420,BLUE,5,5,"新用户注册",LIGHTGRAY)){
            if(pressbarword16(450,420,BLUE,5,5,"新用户注册",LIGHTGRAY)==2){
                MouseS=1;
                if (flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword16(450,420,BLUE,5,5,"新用户注册",RED);
                    flag=5;
                }
                continue;
            }
            if(pressbarword16(450,420,BLUE,5,5,"新用户注册",LIGHTGRAY)==1){
                MouseS=0;
                delay(100);
                *page=3;
                break;
            }
        }
        if(flag){
            if(flag==1){
                clrmous(MouseX,MouseY);
                barword32(0,350,BLUE,3,2,"登录",LIGHTGRAY);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==5){
                clrmous(MouseX,MouseY);
                barword16(450,420,BLUE,5,5,"新用户注册",LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }

    return;
}

```

```c
void pageadlogin(int *page){
    int pos=0;
    int flag=0;
    int loginjudge;
    char name[20]={'\0'};
    char password[20]={'\0'};
    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadlogin();

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=0;
                break;
            }
        }

        if(inbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag=1;
                }
                continue;
            }
            else if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                name[0]='\0';
                barwordframe(180,170,WHITE,11,"",DARKGRAY,RED);
                Input_Vis(name,180,170,18,WHITE,BLACK);         //可视输入
                if(strlen(name)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }

        if(inbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag==1;
                }
                continue;
            }
            else if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                password[0]='\0';
                barwordframe(180,250,WHITE,11,"",DARKGRAY,RED);
                Input_Invis(password,180,250,18,WHITE,BLACK);   //不可视输入
                if(strlen(password)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }

        if(inbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)){
            if(pressbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)==2){
                MouseS=1;
                if (flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,350,BLUE,3,2,"登录",RED);
                    flag=1;
                }
                continue;
            }
            else if (pressbarword32(0,350,BLUE,3,2,"登录",LIGHTGRAY)==1)
            {
                barword32(0,350,BLUE,3,2,"登录",RED);
                MouseS=0;
                loginjudge=loginadmin(name,password);
                if(loginjudge==1){
                    loginok();
                    delay(300);
                    *page=21;
                    break;
                }
                else if(loginjudge==-1){
                    loginpasswordwrong();
                    delay(300);
                    *page=2;
                    break;
                }
                else if(loginjudge==-2){
                    loginnoregister();
                    delay(300);
                    *page=2;
                    break;
                }
                else{
                    exit(1);            //todo
                }
            }
        }

        if(flag){
            if(flag==1){
                clrmous(MouseX,MouseY);
                barword32(0,350,BLUE,3,2,"登录",LIGHTGRAY);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
    return;
}
```

10. pagead.c

```
C
```

```c
#include"public.h"

void pageadmanager(int *page,char searchID[20]){
    int pos=0;
    int flag=0;
    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadmanager();

    while(1){
        newmouse(&MouseX,&MouseY,&press);

        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=2;
                break;
            }
        }

        if(inbarwordframe(100,100,WHITE,11,"",WHITE,RED)==1){
            if(pressbarwordframe(100,100,WHITE,11,"",WHITE,RED)==2){
                MouseS=2;
                continue;
            }
            else if(pressbarwordframe(100,100,WHITE,11,"",WHITE,RED)==1){
                MouseS=0;
                Input_Vis(searchID,100,100,18,WHITE,BLACK);
                if(strlen(searchID)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }

        if(insearchbuttuon()==1){
            if(presssearchbutton()==2){
                MouseS=1;
                if (flag!=822){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    searchbutton(RED);
                    flag=822;
                }
                continue;
            }
            else if(presssearchbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=215;
                break;
            }
        }

        if (inbarword32(100,200,CYAN,6,6,"标记乘客状况",WHITE)){
            if(pressbarword32(100,200,CYAN,6,6,"标记乘客状况",WHITE)==2){
                MouseS=1;
                if(flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(100,200,CYAN,6,6,"标记乘客状况",RED);
                    flag=1;
                }
                continue;
            }
            if(pressbarword32(100,200,CYAN,6,6,"标记乘客状况",WHITE)==1){
                barword32(100,200,CYAN,6,6,"标记乘客状况",RED);
                MouseS=0;
                *page=211;
                break;
            }
        }

        if (inbarword32(350,200,DARKGRAY,6,6,"查询阳性乘客",WHITE)){
            if(pressbarword32(350,200,DARKGRAY,6,6,"查询阳性乘客",WHITE)==2){
                MouseS=1;
                if(flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
```

```c
                barword32(350,200,DARKGRAY,6,6,"查询阳性乘客",RED);
                flag=2;
            }
            continue;
        }
        if(pressbarword32(350,200,DARKGRAY,6,6,"查询阳性乘客",WHITE)==1){
            barword32(350,200,DARKGRAY,6,6,"查询阳性乘客",RED);
            MouseS=0;
            *page=212;
            break;
        }
    }

    if (inbarword32(350,300,BLUE,6,6,"查询密接乘客",WHITE)){
        if(pressbarword32(350,300,BLUE,6,6,"查询密接乘客",WHITE)==2){
            MouseS=1;
            if(flag==0){
                clrmous(MouseX,MouseY);
                delay(10);
                barword32(350,300,BLUE,6,6,"查询密接乘客",RED);
                flag=3;
            }
            continue;
        }
        if(pressbarword32(350,300,BLUE,6,6,"查询密接乘客",WHITE)==1){
            barword32(350,300,BLUE,6,6,"查询密接乘客",RED);
            MouseS=0;
            *page=214;
            break;
        }
    }

    if (inbarword32(100,300,BROWN,6,6,"查询可疑车次",WHITE)){
        if(pressbarword32(100,300,BROWN,6,6,"查询可疑车次",WHITE)==2){
            MouseS=1;
            if(flag==0){
                clrmous(MouseX,MouseY);
                delay(10);
                barword32(100,300,BROWN,6,6,"查询可疑车次",RED);
                flag=4;
            }
            continue;
        }
        if(pressbarword32(100,300,BROWN,6,6,"查询可疑车次",RED)==1){
            barword32(100,300,BROWN,6,6,"查询可疑车次",RED);
            MouseS=0;
            *page=213;
```

```c
                    break;
                }
            }

            if(flag){
                if(flag==1){
                    clrmous(MouseX,MouseY);
                    barword32(100,200,CYAN,6,6,"标记乘客状况",WHITE);
                }
                if(flag==2){
                    clrmous(MouseX,MouseY);
                    barword32(350,200,DARKGRAY,6,6,"查询阳性乘客",WHITE);
                }
                if(flag==3){
                    clrmous(MouseX,MouseY);
                    barword32(350,300,BLUE,6,6,"查询密接乘客",WHITE);
                }
                if(flag==4){
                    clrmous(MouseX,MouseY);
                    barword32(100,300,BROWN,6,6,"查询可疑车次",WHITE);
                }
                if(flag==113){
                    clrmous(MouseX,MouseY);
                    returnbutton(LIGHTGRAY);
                }
                if(flag==822){
                    clrmous(MouseX,MouseY);
                    searchbutton(WHITE);
                }
                flag=0;
            }
            if(MouseS!=0){
                MouseS=0;
            }
        }
    return ;
}

void pageadlabel(int *page){
    int asymptomatic=0;
    int symptomatic=0;
    char ID[20]={'\0'};
    int trkeyset[50];
    int pakeyset[50];
    int counttrkey;
    int countpakey;
    int pos=0;
    int flag=0;
    int pakey;
    int i,j;

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadlabel();

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=21;
                break;
            }
        }

        if(inbarwordframe(230,150,WHITE,11,"",0,RED)){
            if(pressbarwordframe(230,150,WHITE,11,"",0,RED)==2){
                MouseS=2;
                continue;
            }
            else if(pressbarwordframe(230,150,WHITE,11,"",0,RED)==1){
                MouseS=0;
                Input_Vis(ID,230,150,18,WHITE,BLACK);
                if(strlen(ID)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(incircle(380,252,10)){
            if (presscircle(380,252,10)==2)
            {
                MouseS=1;
                continue;                                           //todo:预点击?
            }
            if (presscircle(380,252,10)==1)
            {
                clrmous(MouseX,MouseY);
                setfillstyle(1,BLUE);
                setcolor(BLUE);

                circle(380,252,7);
                floodfill(380,252,BLUE);
                delay(10);
                if(symptomatic==1){
                    clrmous(MouseX,MouseY);
                    setfillstyle(1,LIGHTGRAY);
                    setcolor(LIGHTGRAY);

                    circle(480,252,7);
                    floodfill(480,252,LIGHTGRAY);
                    symptomatic=0;
                }
                asymptomatic=1;
                continue;
            }
        }

        if(incircle(480,252,10)){
            if (presscircle(480,252,10)==2)
            {
                MouseS=1;
                continue;                                           //todo:预点击?
            }
            if (presscircle(480,252,10)==1)
            {
                clrmous(MouseX,MouseY);
                setfillstyle(1,BLUE);
                setcolor(BLUE);

                circle(480,252,7);
                floodfill(480,252,BLUE);
                delay(10);
                if(asymptomatic==1){
                    clrmous(MouseX,MouseY);
                    setfillstyle(1,LIGHTGRAY);
                    setcolor(LIGHTGRAY);

                    circle(380,252,7);
                    floodfill(380,252,LIGHTGRAY);
                    asymptomatic=0;
                }
                symptomatic=1;
                continue;
            }
        }

        if(inbarword32(0,320,GREEN,4,4,"确认录入",WHITE)){
            if(pressbarword32(0,320,GREEN,4,4,"确认录入",WHITE)==2){
                MouseS=1;
                if(flag==0){
                    clrmous(MouseX,MouseY);
//todo:???
                    delay(10);
                    barword32(0,320,GREEN,4,4,"确认录入",BLUE);
                    flag=1;
                }
                continue;
            }
            if(pressbarword32(0,320,GREEN,4,4,"确认录入",WHITE)==1){
                barword32(0,320,GREEN,4,4,"确认录入",BLUE);
                MouseS=0;
                if(getpakeybyID(ID,&pakey)!=1){
                    labelIDzero();
                    delay(300);
                    *page=211;
                    break;;
                }
                delay(100);
                if(symptomatic){
                    updatestatusbypakey(pakey,1);
                    counttrkey=getrecordbypakey(pakey,trkeyset);
                    for(i=0;i<counttrkey;i++){
                        updatestatusbytrkey(trkeyset[i],1);
                        countpakey=getrecordbytrkey(trkeyset[i],pakeyset);
                        for(j=0;j<countpakey;j++){
                            updatestatusbypakey(pakeyset[j],3);
                        }
                    }
                }
                if(asymptomatic){
                    updatestatusbypakey(pakey,2);
                    counttrkey=getrecordbypakey(pakey,trkeyset);
                    for(i=0;i<counttrkey;i++){
```

```c
                        updatestatusbytrkey(trkeyset[i],1);
                        countpakey=getrecordbytrkey(trkeyset[i],pakeyset);
                        for(j=0;j<countpakey;j++){
                            updatestatusbypakey(pakeyset[j],3);
                        }
                    }
                }
                labelok();
                delay(300);
                *page=211;
                break;
            }
        }

        if(flag){
            if(flag==1){
                clrmous(MouseX,MouseY);
                barword32(0,320,GREEN,4,4,"确认录入",WHITE);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageadgetpospa(int *page){
    int flag;
    struct passenger target[20];
    struct passenger *pau;
    struct passenger *pam;
    struct pagetrain *pal;
    int i;
    int u=0,m=0,l=0;

    int currentpage=1;
    int countlabel=getpassengerbystatus(1,target);
    int countpage=(countlabel-1)/3+1;

    FILE *log;

    log=fopen(".\\LOGGER","a+");
    log=fopen(".\\LOGGER","a+");

    for(i=0;i<countlabel;i++) fprintf(log,"\n %d  %s",target[i].pakey,target[i].ID);

    fclose(log);

    u=0;
    m=1;
    l=2;
    pau=&target[u];
    pam=&target[m];
    pal=&target[l];


    if(u>=countlabel){
        pau=NULL;
    }
    if(m>=countlabel){
        pam=NULL;
    }
    if(l>=countlabel){
        pal=NULL;
    }

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadgetpospa(pau,pam,pal,currentpage,countpage);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=21;
```

```
                    break;
                }
            }
            if(inleftarrow()==1&&currentpage!=1){
                if(pressleftarrow()==2){
                    MouseS=1;
                    if (flag!=1){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        leftarrow(RED);
                        flag=1;
                    }
                    continue;
                }
                else if(pressleftarrow()==1){
                    MouseS=0;
                    u-=3;
                    m-=3;
                    l-=3;
                    pau=&target[u];
                    pam=&target[m];
                    pal=&target[l];
                    currentpage--;
                    drawadgetpospa(pau,pam,pal,currentpage,countpage);
                    continue;
                }
            }
            if(inrightarrow()==1&&currentpage!=countpage){
                if(pressrightarrow()==2){
                    MouseS=1;
                    if (flag!=2){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        rightarrow(RED);
                        flag=2;
                    }
                    continue;
                }
                else if(pressrightarrow()==1){
                    MouseS=0;
                    u+=3;
                    m+=3;
                    l+=3;
                    if(u>=countlabel){
                        pau=NULL;
                    }
                    else{
                        pau=&target[u];
                    }
                    if(m>=countlabel){
                        pam=NULL;
                    }
                    else{
                        pam=&target[m];
                    }
                    if(l>=countlabel){
                        pal=NULL;
                    }
                    else{
                        pal=&target[u];
                    }
                    currentpage++;
                    drawadgetpospa(pau,pam,pal,currentpage,countpage);
                    continue;
                }
            }
            if(flag){
                if(flag==113){
                    clrmous(MouseX,MouseY);
                    returnbutton(LIGHTGRAY);
                    flag=0;
                }
                if(flag==1){
                    clrmous(MouseX,MouseY);
                    leftarrow(BLUE);
                    flag=0;
                }
                if(flag==2){
                    clrmous(MouseX,MouseY);
                    rightarrow(BLUE);
                    flag=0;
                }
            }
            if(MouseS!=0){
                MouseS=0;
            }
        }
    }

    void pageadgetcttpa(int *page){
        int flag;
        struct passenger target[20];
        struct passenger *pau;
        struct passenger *pam;
        struct pagetrain *pal;
        int i;
        int u=0,m=0,l=0;

        int currentpage=1;
        int countlabel=getpassengerbystatus(3,target);
        int countpage=(countlabel-1)/3+1;

        u=0;
        m=1;
        l=2;
        pau=&target[u];
        pam=&target[m];
        pal=&target[l];

        if(u>=countlabel){
            pau=NULL;
        }
        if(m>=countlabel){
            pam=NULL;
        }
        if(l>=countlabel){
            pal=NULL;
        }

        clrmous(MouseX,MouseY);
        delay(100);
        save_bk_mou(MouseX,MouseY);

        drawadgetcttpa(pau,pam,pal,currentpage,countpage);

        while(1){
            newmouse(&MouseX,&MouseY,&press);
            if(inreturnbutton()==1){
                if(pressreturnbutton()==2){
                    MouseS=1;
                    if (flag!=113){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        returnbutton(RED);
                        flag=113;
                    }
                    continue;
                }
                else if(pressreturnbutton()==1){
                    returnbutton(RED);
                    MouseS=0;
                    *page=21;
                    break;
                }
            }
            if(inleftarrow()==1&&currentpage!=1){
                if(pressleftarrow()==2){
                    MouseS=1;
                    if (flag!=1){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        leftarrow(RED);
                        flag=1;
                    }
                    continue;
                }
                else if(pressleftarrow()==1){
                    MouseS=0;
                    u-=3;
                    m-=3;
                    l-=3;
                    pau=&target[u];
                    pam=&target[m];
                    pal=&target[l];
                    currentpage--;
                    drawadgetcttpa(pau,pam,pal,currentpage,countpage);
                    continue;
                }
            }
            if(inrightarrow()==1&&currentpage!=countpage){
                if(pressrightarrow()==2){
                    MouseS=1;
                    if (flag!=2){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        rightarrow(RED);
                        flag=2;
                    }
                    continue;
                }
                else if(pressrightarrow()==1){
                    MouseS=0;
                    u+=3;
                    m+=3;
                    l+=3;
                    if(u>=countlabel){
                        pau=NULL;
                    }
                    else{
                        pau=&target[u];
                    }
```

```c
                if(m>=countlabel){
                    pam=NULL;
                }
                else{
                    pam=&target[m];
                }
                if(l>=countlabel){
                    pal=NULL;
                }
                else{
                    pal=&target[u];
                }
                currentpage++;
                drawadgetcttpa(pau,pam,pal,currentpage,countpage);
                continue;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==1){
                clrmous(MouseX,MouseY);
                leftarrow(BLUE);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                rightarrow(BLUE);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageadgetpostrn(int *page){
    int flag;
    struct pagetrain target[20];
    struct pagetrain *ptrnu;
    struct pagetrain *ptrnm;
    struct pagetrain *ptrnl;
    int i;
    //todo u的值
    int u=0,m=0,l=0;

    int currentpage=1;
    int countlabel=gettrainbystatus(1,target);
    int countpage=(countlabel-1)/3+1;

    u=0;
    m=1;
    l=2;
    ptrnu=&target[u];
    ptrnm=&target[m];
    ptrnl=&target[l];

    if(u>=countlabel){
        ptrnu=NULL;
    }
    if(m>=countlabel){
        ptrnm=NULL;
    }
    if(l>=countlabel){
        ptrnl=NULL;
    }

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadgetpostr(ptrnu,ptrnm,ptrnl,currentpage,countpage);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=21;
                break;
            }
        }
        if(inleftarrow()==1&&currentpage!=1){
            if(pressleftarrow()==2){
                MouseS=1;
                if (flag!=1){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    leftarrow(RED);
                    flag=1;
                }
                continue;
            }
            else if(pressleftarrow()==1){
                MouseS=0;
                u-=3;
                m-=3;
                l-=3;
                ptrnu=&target[u];
                ptrnm=&target[m];
                ptrnl=&target[u];
                currentpage--;
                drawadgetpostr(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(inrightarrow()==1&&currentpage!=countpage){
            if(pressrightarrow()==2){
                MouseS=1;
                if (flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    rightarrow(RED);
                    flag=2;
                }
                continue;
            }
            else if(pressrightarrow()==1){
                MouseS=0;
                u+=3;
                m+=3;
                l+=3;
                if(u>=countlabel){
                    ptrnu=NULL;
                }
                else{
                    ptrnu=&target[u];
                }
```

```c
                }
                if(m>=countlabel){
                    ptrnm=NULL;
                }
                else{
                    ptrnm=&target[m];
                }
                if(l>=countlabel){
                    ptrnl=NULL;
                }
                else{
                    ptrnl=&target[u];
                }
                currentpage++;
                drawadgetpostr(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==1){
                clrmous(MouseX,MouseY);
                leftarrow(BLUE);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                rightarrow(BLUE);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageadsearch(int *page,char searchID[20],int *pakey){
    int flag;
    struct passenger target[20];
    struct passenger *pau;
    struct passenger *pam;
    struct passenger *pal;
    int i;
    //todo u的值
    int u=0,m=0,l=0;

    int currentpage=1;
    int countlabel=getpassengerbysearch(searchID,target);
    int countpage=(countlabel-1)/3+1;

    memset(searchID,'\0',sizeof(searchID));

    u=0;
    m=1;
    l=2;
    pau=&target[u];
    pam=&target[m];
    pal=&target[l];

    if(u>=countlabel){
        pau=NULL;
    }
    if(m>=countlabel){
        pam=NULL;
    }
    if(l>=countlabel){
        pal=NULL;
    }

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadsearch(pau,pam,pal,currentpage,countpage);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
```

```
                    *page=21;
                    break;
                }
            }
            if(inleftarrow()==1&&currentpage!=1){
                if(pressleftarrow()==2){
                    MouseS=1;
                    if (flag!=1){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        leftarrow(RED);
                        flag=1;
                    }
                    continue;
                }
                else if(pressleftarrow()==1){
                    MouseS=0;
                    u-=3;
                    m-=3;
                    l-=3;
                    pau=&target[u];
                    pam=&target[m];
                    pal=&target[u];
                    currentpage--;
                    drawadsearch(pau,pam,pal,currentpage,countpage);
                    continue;
                }
            }
            if(inrightarrow()==1&&currentpage!=countpage){
                if(pressrightarrow()==2){
                    MouseS=1;
                    if (flag!=2){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        rightarrow(RED);
                        flag=2;
                    }
                    continue;
                }
                else if(pressrightarrow()==1){
                    MouseS=0;
                    u+=3;
                    m+=3;
                    l+=3;
                    if(u>=countlabel){
                        pau=NULL;
                    }
                    else{
```

```
                else{
                    pau=&target[u];
                }
                if(m>=countlabel){
                    pam=NULL;
                }
                else{
                    pam=&target[m];
                }
                if(l>=countlabel){
                    pal=NULL;
                }
                else{
                    pal=&target[u];
                }
                currentpage++;
                drawadsearch(pau,pam,pal,currentpage,countpage);
                continue;
            }
        }
        if(u<countlabel&&inpalabelu()==1){
            if(presspalabelu()==2){
                MouseS=1;
                continue;
            }
            if(presspalabelu()==1){
                *pakey=pau->pakey;
                delay(10);
                *page=221;
                break;
            }
        }
        if(m<countlabel&&inpalabelm()==1){
            if(presspalabelm()==2){
                MouseS=1;
                continue;
            }
            if(presspalabelm()==1){
                *pakey=pam->pakey;
                delay(10);
                *page=221;
                break;
            }
        }
        if(l<countlabel&&inpalabell()==1){
            if(presspalabell()==2){
                MouseS=1;
                continue;
            }
            if(presspalabell()==1){
                *pakey=pal->pakey;
                delay(10);
                *page=221;
                break;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==1){
                clrmous(MouseX,MouseY);
                leftarrow(BLUE);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                rightarrow(BLUE);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageadpamessage(int *page,int *pakey){
    int flag=0;
    int pos=0;
    int judgepakey;
    struct passenger pa;

    //todo
    judgepakey=getpassengerbypakey(*pakey,&pa);

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawadpamessage(&pa);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
```

```c
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=21;
                break;
            }
        }
        if(inbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)){
            if(pressbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)==2){
                MouseS=1;
                if(flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(265,400,BLUE,4,4,"乘车记录",RED);
                    flag=2;
                }
                continue;
            }
            else if(pressbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)==1){
                MouseS=0;
                delay(100);
                *page=222;
                break;
            }
        }
        if(inbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)){
            if(pressbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)==2){
                MouseS=1;
                if(flag!=9){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(465,400,BLUE,4,4,"轨迹查询",RED);
                    flag=9;
                }
                continue;
            }
            else if(pressbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)==1){
```

```c
            else if(pressbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)==1){
                MouseS=0;
                delay(100);
                *page=223;
                break;
            }
        }
        if(flag){
            if(flag==2){
                clrmous(MouseX,MouseY);
                barword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY);
                flag=0;
            }
            if(flag==9){
                clrmous(MouseX,MouseY);
                barword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }

    }

}

void pageadpagetrecord(int *page,int *pakey){
    int flag;
    int trkeyset[20]={0};
    struct train *ptrnu;
    struct train *ptrnm;
    struct train *ptrnl;
    int i;

    int currentpage=1;
    int countlabel=getrecordbypakey(*pakey,trkeyset);
    int countpage=(countlabel-1)/3+1;

    int u=0,m=0,l=0;

    ptrnu=(struct train *)malloc(sizeof(TRAIN));
    ptrnm=(struct train *)malloc(sizeof(TRAIN));
    ptrnl=(struct train *)malloc(sizeof(TRAIN));

    u=0;
    m=1;
    l=2;
    gettrainbytrkey(trkeyset[u],ptrnu);
    gettrainbytrkey(trkeyset[m],ptrnm);
    gettrainbytrkey(trkeyset[l],ptrnl);

    if(u>=countlabel){
        ptrnu=NULL;
    }
    if(m>=countlabel){
        ptrnm=NULL;
    }
    if(l>=countlabel){
        ptrnl=NULL;
    }

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=221;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                break;
            }
        }
        if(inleftarrow()==1&&currentpage!=1){
            if(pressleftarrow()==2){
                MouseS=1;
                if (flag!=1){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    leftarrow(RED);
                    flag=1;
                }
                continue;
            }
            else if(pressleftarrow()==1){
                MouseS=0;
                u-=3;
                m-=3;
                l-=3;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                ptrnu=(struct train *)malloc(sizeof(TRAIN));
                ptrnm=(struct train *)malloc(sizeof(TRAIN));
                ptrnl=(struct train *)malloc(sizeof(TRAIN));
                gettrainbytrkey(trkeyset[u],ptrnu);
                gettrainbytrkey(trkeyset[m],ptrnm);
                gettrainbytrkey(trkeyset[l],ptrnl);
                currentpage--;
                drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(inrightarrow()==1&&currentpage!=countpage){
            if(pressrightarrow()==2){
                MouseS=1;
                if (flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    rightarrow(RED);
                    flag=2;
                }
                continue;
            }
            else if(pressrightarrow()==1){
                MouseS=0;
                u+=3;
                m+=3;
                l+=3;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                ptrnu=(struct train *)malloc(sizeof(TRAIN));
                ptrnm=(struct train *)malloc(sizeof(TRAIN));
                ptrnl=(struct train *)malloc(sizeof(TRAIN));
                gettrainbytrkey(trkeyset[u],ptrnu);
                gettrainbytrkey(trkeyset[m],ptrnm);
                gettrainbytrkey(trkeyset[l],ptrnl);
                if(u>=countlabel){
                     ptrnu=NULL;
                }
                if(m>=countlabel){
                     ptrnm=NULL;
                }
                if(l>=countlabel){
                    ptrnl=NULL;
                }
                currentpage++;
                drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==1){
                clrmous(MouseX,MouseY);
                leftarrow(BLUE);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                rightarrow(BLUE);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageadpagettrack(int *page,int *pakey){
    int flag;
    char trackset[10][10];
    int i;
    int trackcount=gettrackbypakey(*pakey,trackset);

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpagettrack();

    for(i=0;i<trackcount;i++){
        mapline(trackset[i],RED);
    }

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=221;
                break;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}
```

11. pagepa.c

JavaScript

```
#include "public.h"

void pageparegister(int *page){
    int pos=0;
    int flag=0;
    char name[20]={'\0'};
    char password1[20]={'\0'};
    char password2[20]={'\0'};

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawparegister();

    while(1){
        newmouse(&MouseX,&MouseY,&press);

        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=0;
                break;
            }
        }
        if(inbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag=1;
                }
                continue;
            }
            else if(pressbarwordframe(180,170,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                name[0]='\0';
                barwordframe(180,170,WHITE,11,"",DARKGRAY,RED);
                Input_Vis(name,180,170,18,WHITE,BLACK);         //可视输入
                if(strlen(name)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag==1;
                }
                continue;
            }
            else if(pressbarwordframe(180,250,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                password1[0]='\0';
                barwordframe(180,250,WHITE,11,"",DARKGRAY,RED);
                Input_Invis(password1,180,250,18,WHITE,BLACK);  //不可视输入
                if(strlen(password1)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos){
                    flag==1;
                }
                continue;
            }
            else if(pressbarwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                password2[0]='\0';
                delay(10);
                barwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED);
                barwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED);
                Input_Invis(password2,180,330,18,WHITE,BLACK);
                if(strlen(password2)==0){
                    barwordframe(180,330,WHITE,11,"",LIGHTGRAY,RED);
                    puthz(190,335,"再次输入密码",32,32,LIGHTGRAY);      //todo
                }
                if(strlen(password1)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarword32(0,400,BLUE,2,2,"注册",LIGHTGRAY)){
            if(pressbarword32(0,400,BLUE,2,2,"注册",LIGHTGRAY)==2){
                MouseS=1;
                if (flag==0){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,400,BLUE,2,2,"注册",RED);
                    flag=1;
                }
                continue;
            }
            if(pressbarword32(0,400,BLUE,2,2,"注册",LIGHTGRAY)==1){
                MouseS=0;
                if(strlen(name)==0){
                    registernamezero();
                    delay(1000);
                    *page=3;
                    break;
                }
                if(strlen(password1)==0){
                    registerpasszero();
                    delay(1000);
                    *page=3;
                    break;
                }
                if(strcmp(password1,password2)!=0){
                    registerpasswrong();
                    delay(1000);
                    *page=3;
                    break;
                }
                if(strlen(password1)<6){
                    registerpassshort();
                    delay(1000);
                    *page=3;
                    break;
                }
                if(existusername(name)==1){
                    registerrepeat();
                    delay(1000);
                    *page=3;
                    break;
                }
                else{
                    writeuser(name,password1);
                    registerok();
                    delay(1000);
                    *page=1;
                    break;
                }
            }
        }
        if(flag){
            if(flag==1){
                clrmous(MouseX,MouseY);
                barword32(0,400,BLUE,2,2,"注册",LIGHTGRAY);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
         }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pagepabind(int *page,int *pakey){
    int pos=0;
    int flag=0;
    char tel[20]={'\0'};
    char ID[20]={'\0'};

    clrmous(MouseX,MouseY);
    delay(100);
```

```c
    save_bk_mou(MouseX,MouseY);

    drawpabind();

    while(1){
        newmouse(&MouseX,&MouseY,&press);

        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=1;
                break;
            }
        }

        if(inbarwordframe(210,200,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(210,200,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag=1;
                }
                continue;
            }
            else if(pressbarwordframe(210,200,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                ID[0]='\0';
                barwordframe(210,200,WHITE,11,"",LIGHTGRAY,RED);
                Input_Vis(ID,210,200,18,WHITE,BLACK);
                if(strlen(ID)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarwordframe(210,300,WHITE,11,"",LIGHTGRAY,RED)){
            if(pressbarwordframe(210,300,WHITE,11,"",LIGHTGRAY,RED)==2){
                MouseS=2;
                if(flag==0&&pos==0){
                    flag=1;
                }
                continue;
            }
            else if(pressbarwordframe(210,300,WHITE,11,"",LIGHTGRAY,RED)==1){
                MouseS=0;
                tel[0]='\0';
                barwordframe(210,300,WHITE,11,"",LIGHTGRAY,RED);
                Input_Vis(tel,210,300,11,WHITE,BLACK);
                if(strlen(tel)!=0){
                    pos=1;
                }
                else{
                    pos=0;
                }
                continue;
            }
        }
        if(inbarword32(0,400,BLUE,2,2,"绑定",WHITE)){
            if(pressbarword32(0,400,BLUE,2,2,"绑定",WHITE)==2){
                MouseS=1;
                if (flag!=5){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(0,400,BLUE,2,2,"绑定",RED);
                    flag=5;
                }
                continue;
            }
            else if(pressbarword32(0,400,BLUE,2,2,"绑定",WHITE)==1){
                MouseS=0;
                if(judgeID(ID)!=1){
                    bindIDwrong();
                    delay(300);
                    *page=4;
                    break;
                }
                if(judgetel(tel)!=1){
                    bindtelwrong();
                    delay(300);
                    *page=4;
                    break;
                }
                if(existpassengerID(ID)==1){
```

```c
                if(existpassengerID(ID)==1){
                    bindIDrepeat();
                    delay(300);
                    *page=4;
                    break;
                }
                writepassenger(*pakey,ID,tel,getsexbyID(ID),getagebyID(ID),0);
                bindok();
                delay(1000);
                *page=5;
                break;
            }
        }
        if(flag){
            if(flag==5){
                clrmous(MouseX,MouseY);
                barword32(0,400,BLUE,2,2,"绑定",WHITE);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }

    }
}

void pagepamessage(int *page,int *pakey){
    int flag=0;
    int pos=0;
    int judgepakey;
    struct passenger pa;

    //todo
    judgepakey=getpassengerbypakey(*pakey,&pa);

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpamessage(&pa);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=0;
                break;
            }
        }
        if(inbarword32(65,400,BLUE,4,4,"登记乘车",LIGHTGRAY)){
            if(pressbarword32(65,400,BLUE,4,4,"登记乘车",LIGHTGRAY)==2){
                MouseS=1;
                if(flag!=1){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(65,400,BLUE,4,4,"登记乘车",RED);
                    flag=1;
                }
                continue;
            }
            else if(pressbarword32(65,400,BLUE,4,4,"登记乘车",LIGHTGRAY)==1){
                if(pa.status!=0){
                    statusabnormal();
                    delay(300);
                    *page=5;
                    break;;
                }
                MouseS=0;
                delay(100);
                *page=6;
                break;
            }
        }
        if(inbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)){
            if(pressbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)==2){
                MouseS=1;
                if(flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(265,400,BLUE,4,4,"乘车记录",RED);
                    flag=2;
                }
                continue;
            }
            else if(pressbarword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY)==1){
                MouseS=0;
                delay(100);
                *page=7;
                break;
            }
        }
        if(inbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)){
            if(pressbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)==2){
                MouseS=1;
                if(flag!=9){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(465,400,BLUE,4,4,"轨迹查询",RED);
                    flag=9;
                }
                continue;
            }
            else if(pressbarword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY)==1){
                MouseS=0;
                delay(100);
                *page=9;
                break;
            }
        }
        if(flag){
            if(flag==1){
                clrmous(MouseX,MouseY);
                barword32(65,400,BLUE,4,4,"登记乘车",LIGHTGRAY);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                barword32(265,400,BLUE,4,4,"乘车记录",LIGHTGRAY);
                flag=0;
            }
            if(flag==9){
                clrmous(MouseX,MouseY);
                barword32(465,400,BLUE,4,4,"轨迹查询",LIGHTGRAY);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }

    }

}

void pagepapostrecord(int *page,int *pakey,int *trkey){
    int flag=0;
    char trainname[5]={'\0'};
    int yy;
    int mm;
    int dd;
    int date;

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpapostrecord();

    while(1){
        newmouse(&MouseX,&MouseY,&press);

        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=5;
                break;
            }
        }

        if(inbarwordframe(280,185,WHITE,3,"",0,RED)){
            if(pressbarwordframe(280,185,WHITE,3,"",0,RED)==2){
                MouseS=2;
                if(flag!=1){
                    flag=1;
                }
                continue;
            }
            if(pressbarwordframe(280,185,WHITE,3,"",0,RED)==1){
                MouseS=0;
                memset(trainname,'\0',sizeof(trainname));
                barwordframe(280,185,WHITE,3,"",0,RED);
                Input_Vis(trainname+1,280,185,3,WHITE,BLACK);
                continue;
            }
        }

        if(inbarword32(430,185,BLUE,2,2,"确认",WHITE)){
            if(pressbarword32(430,185,BLUE,2,2,"确认",WHITE)==2){
                MouseS=1;
                if(flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    barword32(430,185,BLUE,2,2,"确认",RED);
                    flag=2;
                }
                continue;
            }
            else if(pressbarword32(430,185,BLUE,2,2,"确认",WHITE)==1){
                MouseS=0;
                barword32(430,185,BLUE,2,2,"确认",RED);
                gettodaydate(&yy,&mm,&dd);
                postdate(yy,mm,dd,&date);
                trainname[0]='G';
                if(getpostedtrkey(date,trainname,trkey)!=1){
                    recordzero();
                    delay(300);
                    *page=6;
                    break;
                }
                delay(10);
                MouseS=0;
                *page=8;
                break;;
            }
        }
        if(flag){
            if(flag==2){
                clrmous(MouseX,MouseY);
                barword32(430,185,BLUE,2,2,"确认",WHITE);
                flag=0;
            }
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pagepagetrecord(int *page,int *pakey){
    int flag;
    int trkeyset[20]={0};
    struct train *ptrnu;
    struct train *ptrnm;
    struct train *ptrnl;
    int i;

    int currentpage=1;
    int countlabel=getrecordbypakey(*pakey,trkeyset);
    int countpage=(countlabel-1)/3+1;

    int u=0,m=0,l=0;

    ptrnu=(struct train *)malloc(sizeof(TRAIN));
    ptrnm=(struct train *)malloc(sizeof(TRAIN));
    ptrnl=(struct train *)malloc(sizeof(TRAIN));

    u=0;
    m=1;
    l=2;
    gettrainbytrkey(trkeyset[u],ptrnu);
    gettrainbytrkey(trkeyset[m],ptrnm);
    gettrainbytrkey(trkeyset[l],ptrnl);

    if(u>=countlabel){
        ptrnu=NULL;
    }
    if(m>=countlabel){
        ptrnm=NULL;
    }
    if(l>=countlabel){
        ptrnl=NULL;
    }

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);

    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=5;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                break;
            }
        }
        if(inleftarrow()==1&&currentpage!=1){
            if(pressleftarrow()==2){
                MouseS=1;
                if (flag!=1){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    leftarrow(RED);
                    flag=1;
                }
                continue;
            }
            else if(pressleftarrow()==1){
                MouseS=0;
                u-=3;
                m-=3;
                l-=3;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                ptrnu=(struct train *)malloc(sizeof(TRAIN));
                ptrnm=(struct train *)malloc(sizeof(TRAIN));
                ptrnl=(struct train *)malloc(sizeof(TRAIN));
                //question
                gettrainbytrkey(trkeyset[u],ptrnu);
                gettrainbytrkey(trkeyset[m],ptrnm);
                gettrainbytrkey(trkeyset[l],ptrnl);
                currentpage--;
                drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(inrightarrow()==1&&currentpage!=countpage){
            if(pressrightarrow()==2){
                MouseS=1;
                if (flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    rightarrow(RED);
                    flag=2;
                }
                continue;
            }
            else if(pressrightarrow()==1){
                MouseS=0;
                u+=3;
                m+=3;
                l+=3;
                free(ptrnu);
                free(ptrnm);
                free(ptrnl);
                ptrnu=(struct train *)malloc(sizeof(TRAIN));
                ptrnm=(struct train *)malloc(sizeof(TRAIN));
                ptrnl=(struct train *)malloc(sizeof(TRAIN));
                gettrainbytrkey(trkeyset[u],ptrnu);
                gettrainbytrkey(trkeyset[m],ptrnm);
                gettrainbytrkey(trkeyset[l],ptrnl);
                if(u>=countlabel){
                     ptrnu=NULL;
                }
                if(m>=countlabel){
```

```
                if(m>=countlabel){
                     ptrnm=NULL;
                }
                if(l>=countlabel){
                    ptrnl=NULL;
                }
                currentpage++;
                drawpagetrecord(ptrnu,ptrnm,ptrnl,currentpage,countpage);
                continue;
            }
        }
        if(flag){
            if(flag==113){
                clrmous(MouseX,MouseY);
                returnbutton(LIGHTGRAY);
                flag=0;
            }
            if(flag==1){
                clrmous(MouseX,MouseY);
                leftarrow(BLUE);
                flag=0;
            }
            if(flag==2){
                clrmous(MouseX,MouseY);
                rightarrow(BLUE);
                flag=0;
            }
        }
        if(MouseS!=0){
            MouseS=0;
        }
    }
}

void pageposttrack(int *page,int *pakey,int *trkey){
    int flag=0;
    char track[6];
    struct train trn;

    gettrainbytrkey(*trkey,&trn);

    clrmous(MouseX,MouseY);
    delay(100);
    save_bk_mou(MouseX,MouseY);

    drawstationselect(trn.trainname);
    while(1){
        newmouse(&MouseX,&MouseY,&press);
        if(inreturnbutton()==1){
            if(pressreturnbutton()==2){
                MouseS=1;
                if (flag!=113){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    returnbutton(RED);
                    flag=113;
                }
                continue;
            }
            else if(pressreturnbutton()==1){
                returnbutton(RED);
                MouseS=0;
                *page=6;
                break;
            }
        }
        if(intrack1()){
            if(presstrack1()==2){
                MouseS=1;
                if(flag!=1){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track1(trn.trainname,RED);
                    flag=1;
                }
                continue;
            }
            if(presstrack1()==1){
                MouseS=0;
                track1(trn.trainname,RED);
                gettrackstring(trn.trainname,1,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack2()){
            if(presstrack2()==2){
                MouseS=1;
                if(flag!=2){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track2(trn.trainname,RED);
                    flag=2;
                }
                continue;
            }
            if(presstrack2()==1){
                MouseS=0;
                track2(trn.trainname,RED);
                gettrackstring(trn.trainname,2,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack3()){
            if(presstrack3()==2){
                MouseS=1;
                if(flag!=3){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track3(trn.trainname,RED);
                    flag=3;
                }
                continue;
            }
            if(presstrack3()==1){
                MouseS=0;
                track3(trn.trainname,RED);
                gettrackstring(trn.trainname,3,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
```

```c
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack4()){
            if(presstrack4()==2){
                MouseS=1;
                if(flag!=4){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track4(trn.trainname,RED);
                    flag=4;
                }
                continue;
            }
            if(presstrack4()==1){
                MouseS=0;
                track4(trn.trainname,RED);
                gettrackstring(trn.trainname,4,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack5()){
            if(presstrack5()==2){
                MouseS=1;
                if(flag!=5){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track5(trn.trainname,RED);
                    flag=5;
                }
                continue;
            }
            if(presstrack5()==1){
                MouseS=0;
                track5(trn.trainname,RED);
                gettrackstring(trn.trainname,5,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack6()){
            if(presstrack6()==2){
                MouseS=1;
                if(flag!=6){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track6(trn.trainname,RED);
                    flag=6;
                }
                continue;
            }
            if(presstrack6()==1){
                MouseS=0;
                track6(trn.trainname,RED);
                gettrackstring(trn.trainname,6,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack7()){
            if(presstrack7()==2){
                MouseS=1;
                if(flag!=7){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track7(trn.trainname,RED);
                    flag=7;
                }
                continue;
            }
            if(presstrack7()==1){
                MouseS=0;
                track7(trn.trainname,RED);
                gettrackstring(trn.trainname,7,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack8()){
            if(presstrack8()==2){
                MouseS=1;
                if(flag!=8){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track8(trn.trainname,RED);
                    flag=8;
                }
                continue;
            }
            if(presstrack8()==1){
                MouseS=0;
                track8(trn.trainname,RED);
                gettrackstring(trn.trainname,8,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack9()){
            if(presstrack9()==2){
                MouseS=1;
                if(flag!=9){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track9(trn.trainname,RED);
                    flag=9;
                }
                continue;
            }
            if(presstrack9()==1){
                MouseS=0;
                track9(trn.trainname,RED);
                gettrackstring(trn.trainname,9,track);
                if(writerecordv1(*pakey,*trkey,track)==-1){
                    trackrecordrepeat();
                    delay(300);
                    *page=8;
                    break;
                }
                else{
                    trackrecordok();
                    delay(300);
                    *page=5;
                    break;
                }
            }
        }
        if(intrack10()){
            if(presstrack10()==2){
                MouseS=1;
                if(flag!=10){
                    clrmous(MouseX,MouseY);
                    delay(10);
                    track10(trn.trainname,RED);
```

```
                        flag=10;
                    }
                    continue;
                }
                if(presstrack10()==1){
                    MouseS=0;
                    track10(trn.trainname,RED);
                    gettrackstring(trn.trainname,10,track);
                    if(writerecordv1(*pakey,*trkey,track)==-1){
                        trackrecordrepeat();
                        delay(300);
                        *page=8;
                        break;
                    }
                    else{
                        trackrecordok();
                        delay(300);
                        *page=5;
                        break;
                    }
                }
            }

            if(flag){
                if(flag==1){
                    clrmous(MouseX,MouseY);
                    track1(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==113){
                    clrmous(MouseX,MouseY);
                    returnbutton(LIGHTGRAY);
                    flag=0;
                }
                if(flag==2){
                    clrmous(MouseX,MouseY);
                    track2(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==3){
                    clrmous(MouseX,MouseY);
                    track3(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==4){
                    clrmous(MouseX,MouseY);
                    track4(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==5){
                    clrmous(MouseX,MouseY);
                    track5(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==6){
                    clrmous(MouseX,MouseY);
                    track6(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==7){
                    clrmous(MouseX,MouseY);
                    track7(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==8){
                    clrmous(MouseX,MouseY);
                    track8(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==9){
                    clrmous(MouseX,MouseY);
                    track9(trn.trainname,WHITE);
                    flag=0;
                }
                if(flag==10){
                    clrmous(MouseX,MouseY);
                    track10(trn.trainname,WHITE);
                    flag=0;
                }
            }
            if(MouseS!=0){
                MouseS=0;
            }
        }

    }

    void pagepagettrack(int *page,int *pakey){
        int flag;
        char trackset[10][10];
        int i;
        int trackcount=gettrackbypakey(*pakey,trackset);

        clrmous(MouseX,MouseY);
        delay(100);
        save_bk_mou(MouseX,MouseY);

        drawpagettrack();

        for(i=0;i<trackcount;i++){
            mapline(trackset[i],RED);
        }

        while(1){
            newmouse(&MouseX,&MouseY,&press);
            if(inreturnbutton()==1){
                if(pressreturnbutton()==2){
                    MouseS=1;
                    if (flag!=113){
                        clrmous(MouseX,MouseY);
                        delay(10);
                        returnbutton(RED);
                        flag=113;
                    }
                    continue;
                }
                else if(pressreturnbutton()==1){
                    returnbutton(RED);
                    MouseS=0;
                    *page=5;
                    break;
                }
            }
            if(flag){
                if(flag==113){
                    clrmous(MouseX,MouseY);
                    returnbutton(LIGHTGRAY);
                    flag=0;
                }
            }
            if(MouseS!=0){
                MouseS=0;
            }
        }
    }
```

十二、时间安排与分工

表格视图 1

	多行文本	多行文本 1	单选
1			
2			
3			

3 条记录