**东北大学秦皇岛分校  
计算中心**

**《C++程序设计》**

**综合设计报告**

|  |  |
| --- | --- |
| 专 业： | 工商管理类 |
| 班 级： | 2303班 |
| 小组编号： | 5组 |
| 学 号： | 202311242 |
| 姓 名： | 湛传斌 |
| 指导教师： | 李飞 |

2024年5月11日

**说 明**

1. 不得改动模板的页面设置、封面及以给正文标题（黑体加粗的文字）的格式和内容。
2. 模板各标题下的给出红色正文内容为示例（示例仅供参考），请将示例删除后依据实际完成内容填写。
3. 正文使用小四字号，中文使用宋体，英文、数字及符号使用Times New Roman（新罗马）字体，1.3倍行距，段落首行缩进2个字符，段前、段后间距为0。
4. 图片在页面内居中对齐，图序号和标题（宋体5号字），居中放置。

**目 录**

**一、** **题目要求** 1

**二、** **设计思路** 1

2.1 概述（简要说明设计思路） 1

2.2 自定义的数据结构、结构体（存储结构）说明 1

2.3 程序的功能结构设计 1

2.4 程序执行过程的总流程描述 1

2.5 程序的组成文件 1

**三、** **小组任务分工** 1

**四、** **程序实现** 1

4.1 使用的集成开发环境 1

4.2 函数的设计与实现 2

**五、** **调试与测试** 2

5.1 自己设计的函数在调试时的测试函数或测试数据与测试结果 2

5.2 调试时遇到的问题及解决方法 2

**六、** **总结** 2

**航班订票管理系统**

1. **题目要求**
2. 信息类型与属性

航班信息：航班号、最大载客数、起飞地点、起飞时间、降落地点、降落时间、单价、剩余座位数量及座位号等。

订票信息：航班号、身份证号码、姓名、性别、出生年月、座位号等。

1. 功能描述
2. 航班信息的录入、修改、删除和查询。
3. 航班订票：录入订票信息，自动更新航班信息。
4. 航班改签：修改订票信息，自动更新航班信息。
5. 航班退票：删除订票信息，自动更新航班信息。
6. 退出系统
7. **设计思路**
   1. 概述（简要说明设计思路）

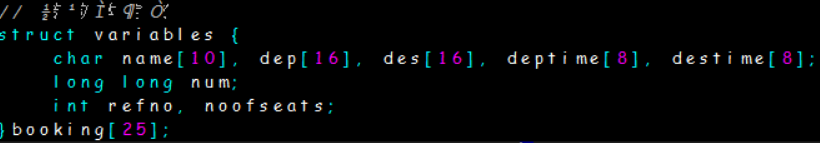
我们商量后决定做一个页面简洁，操作易上手的航班订票管理系统。

我们将其命名NEUQ航班管理系统

首先，我们设计了登录界面并添加了输入错误时的红灯报错。

然后，来到我们订票系统的主界面。我们准备分为订票、已定票务列表、搜索票务信息、修改票务信息、取消预定票务和退出系统几大模块.

* 1. 自定义的数据结构、结构体（存储结构）说明



我们定义了一个名为 `struct variables` 的结构体，用于存储与票务相关的数据。这个结构体包含以下字段：

- `name`: 旅客的姓名，长度为16个字符。

- `dep`: 出发地，长度为16个字符。

- `des`: 目的地，长度为16个字符。

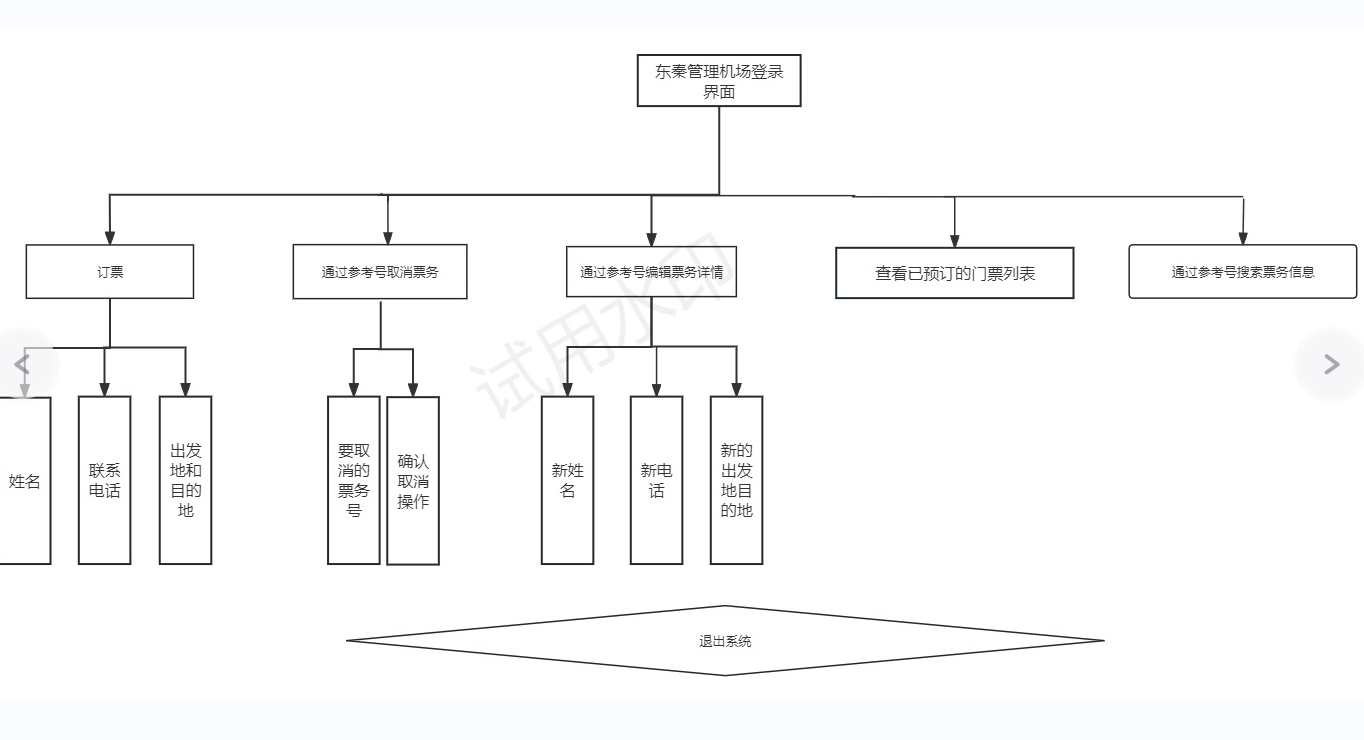
- `deptime`: 出发时间，长度为8个字符。

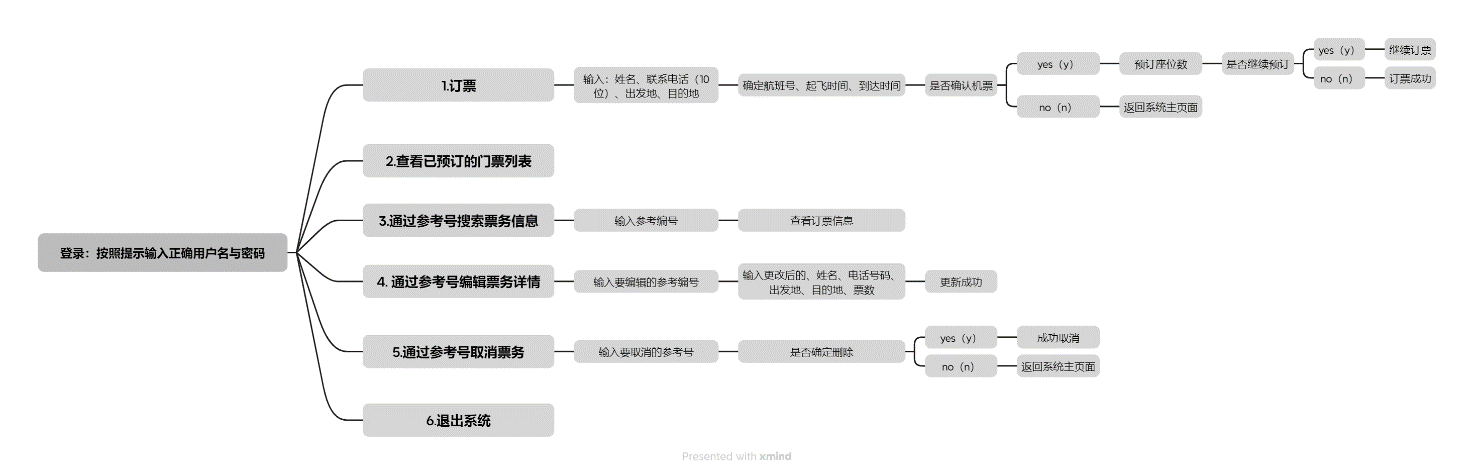
- `destime`: 到达时间，长度为8个字符。

- `num`: 旅客的联系电话，长度为10个数字。

- `refno`: 票务参考号。

- `noofseats`: 预订的座位数。

* 1. 程序的功能结构设计
* 程序的功能结构图
* // 函数声明、函数功能、参数返回值说明
* extern void authentication();// 用于用户认证的函数//湛
* extern void loader(char value[10]);// 显示加载动画的函数//湛
* extern void options();// 显示程序选项的函数//湛
* extern void bookticket();// 用于预订票务的函数 //凌
* extern int count\_digits\_in\_num(long long number);// 计算数字中的位数的函数//凌
* extern void flight\_time(char dep[3], char des[3]);// 根据出发地和目的地提供航班时间的函数//柴
* extern void list\_booked\_tickets();// 显示所有预订的票务的函数//凌
* extern void search\_ticket();// 通过参考号搜索票务的函数//湛
* extern void edit\_ticket();// 通过参考号编辑票务的函数//凌
* extern void cancel\_ticket();// 通过参考号取消票务的函数//柴
* void exit\_terminal();// 退出程序的函数//柴
  1. 程序执行过程的总流程描述



* 1. 程序的组成文件

程序共由1个文件组成，是东秦机场.c。

其中，东秦机场.c包含main函数和以下函数

extern void authentication();// 用于用户认证的函数//湛

extern void loader(char value[10]);// 显示加载动画的函数//湛

extern void options();// 显示程序选项的函数//湛

extern void bookticket();// 用于预订票务的函数 //凌

extern int count\_digits\_in\_num(long long number);// 计算数字中的位数的函数//凌

extern void flight\_time(char dep[3], char des[3]);// 根据出发地和目的地提供航班时间的函数//柴

extern void list\_booked\_tickets();// 显示所有预订的票务的函数//凌

extern void search\_ticket();// 通过参考号搜索票务的函数//湛

extern void edit\_ticket();// 通过参考号编辑票务的函数//凌

extern void cancel\_ticket();// 通过参考号取消票务的函数//柴

void exit\_terminal();// 退出程序的函数//柴

1. **小组任务分工**

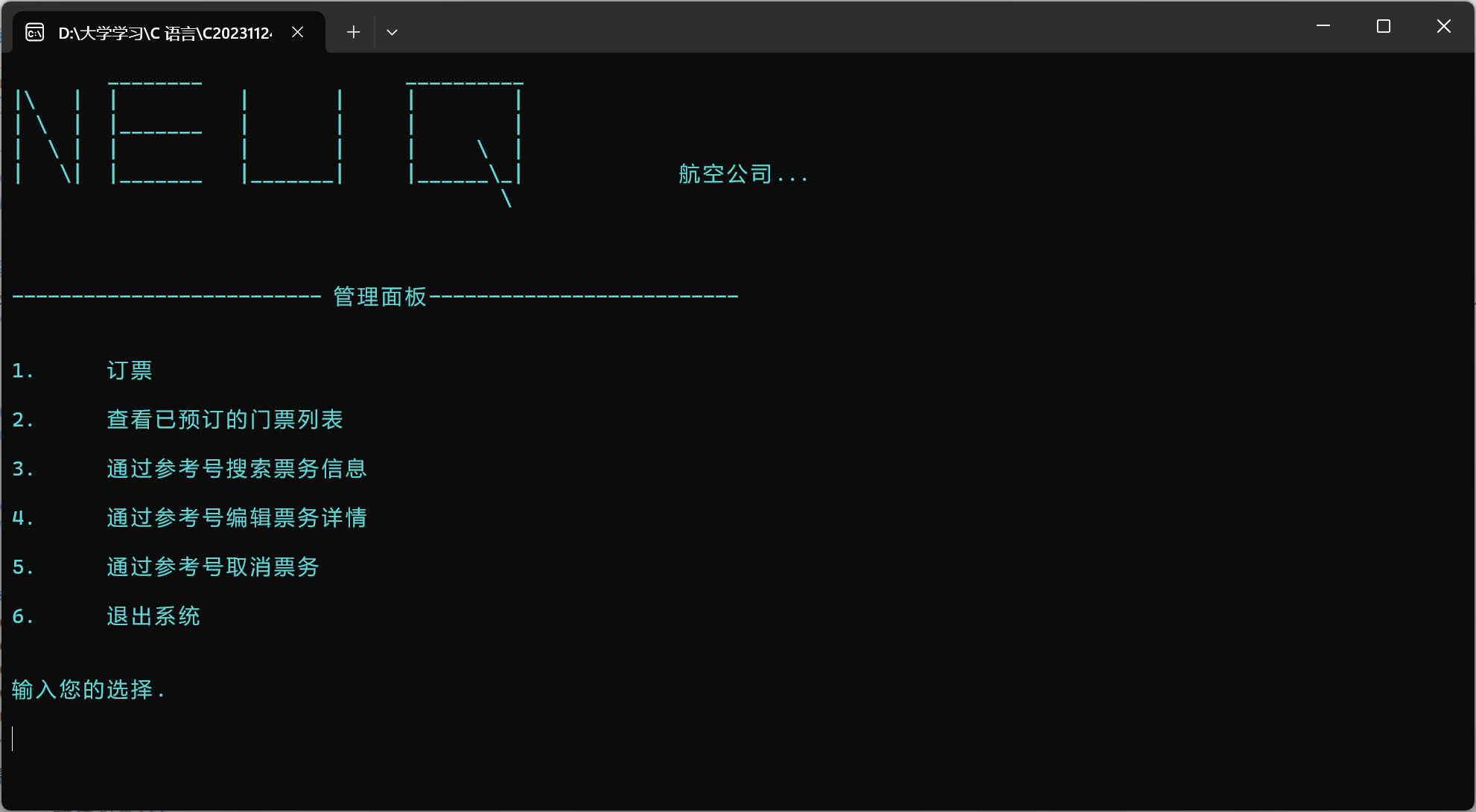
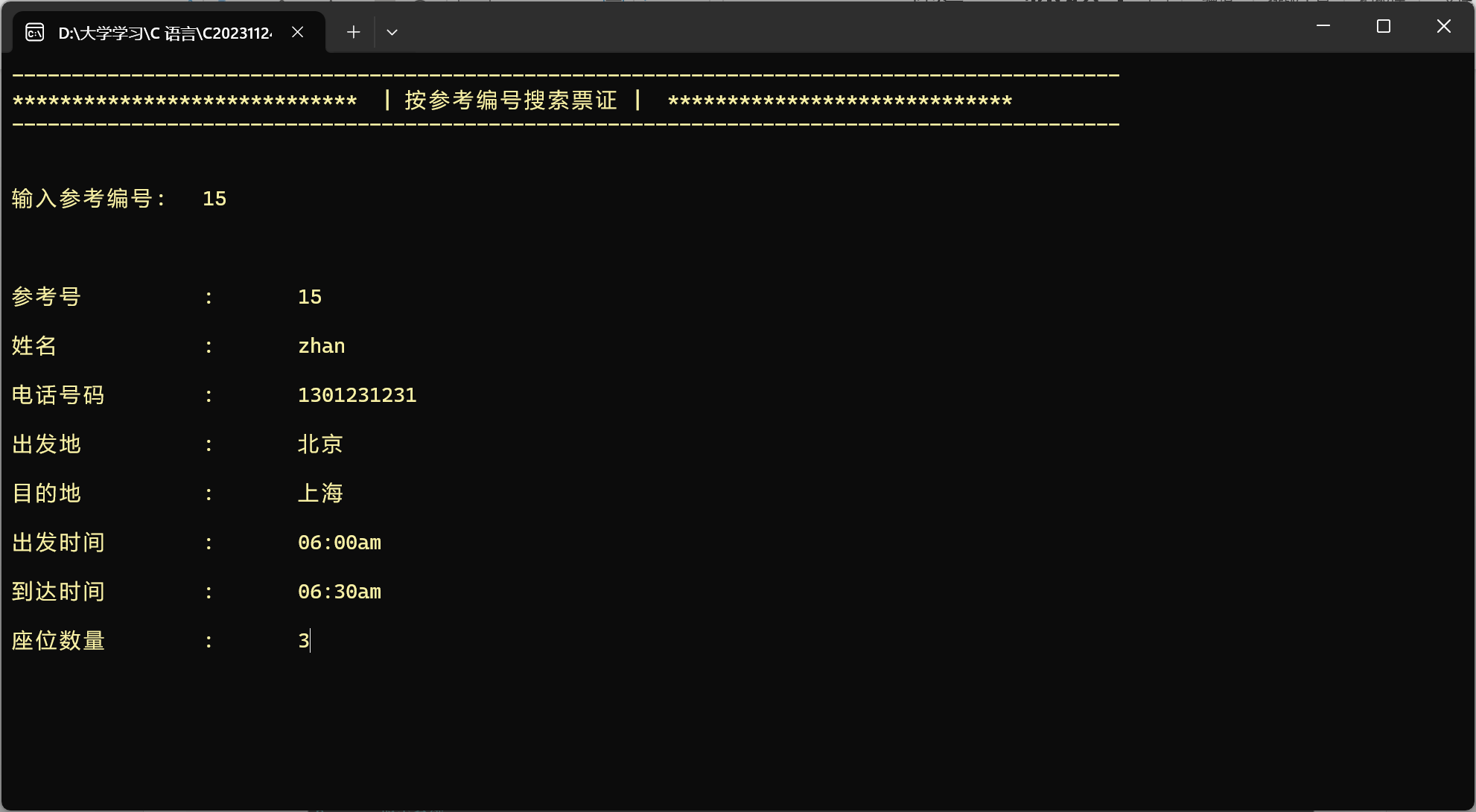
* 湛传斌：完成以下函数设计：
  + extern void authentication();// 用于用户认证的函数//湛
  + extern void loader(char value[10]);// 显示加载动画的函数//湛
  + extern void options();// 显示程序选项的函数//湛
  + extern void search\_ticket();// 通过参考号搜索票务的函数//湛
* 凌勇：完成以下函数设计：
  + extern void bookticket();// 用于预订票务的函数 //凌
  + extern void list\_booked\_tickets();// 显示所有预订的票务的函数//凌
  + extern void edit\_ticket();// 通过参考号编辑票务的函数//凌
  + extern int count\_digits\_in\_num(long long number);// 计算数字中的位数的函数//凌
* 柴东博：完成以下函数设计：
  + extern void flight\_time(char dep[3], char des[3]);// 根据出发地和目的地提供航班时间的函数//柴
  + extern void cancel\_ticket();// 通过参考号取消票务的函数//柴
  + void exit\_terminal();// 退出程序的函数//柴

1. **程序实现**
   1. 使用的集成开发环境

Dev-C++ 5.11

* 1. 函数的设计与实现
* // 用于用户认证的函数
* extern void authentication()
  + /\* `authentication`这个函数的主要功能是验证用户输入的用户名和密码是否正确。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，这通常用于美化用户界面。
  + 2. 使用 `system("cls")` 命令来清除屏幕，以便为用户认证界面提供一个干净的起始点。
  + 3. 接下来，程序打印出登录界面，提示用户输入用户名和密码。
  + 4. 使用 `scanf("%s",&username)` 命令来获取用户名，并使用 `getch()` 命令来获取密码输入，以避免密码在屏幕上显示。
  + 5. 如果用户名和密码都是 "admin" 和 "captcha"，则用户成功登录。此时，重置密码尝试次数 `pw\_tries`，并调用 `loader("loading")` 函数显示加载动画，然后调用 `options()` 函数显示程序选项。
  + 6. 如果密码尝试次数达到3次，则显示错误信息，告诉用户已经三次输入错误，然后退出程序。
  + 7. 如果用户名或密码错误，增加密码尝试次数 `pw\_tries`，并显示错误信息，告诉用户输入的用户名或密码不正确。然后等待用户按键，如果用户按下 Enter 键，则返回主函数 `main()`，重新开始程序。
  + \*/
* // 显示加载动画的函数
* void loader(char value[10])
  + /\* `loader` 函数，该函数用于显示一个简单的加载动画。以下是这个函数的详细解释：
  + 1. 函数接受一个字符串参数 `value`，这个参数可以包含不同的值，以指示加载动画的上下文。例如，如果 `value` 包含 "loading"，则显示 "Loading..."；如果包含 "updating"，则显示 "Updating..."，依此类推。
  + 2. 如果 `value` 包含 "loading"，则打印 "Loading..."。
  + 3. 如果 `value` 包含 "updating"，则打印 "Updating..."。
  + 4. 如果 `value` 包含 "deleting"，则打印 "Deleting..."。
  + 5. 如果 `value` 包含 "loggingout"，则打印 "Logging out..."。
  + 6. 使用一个循环，该循环从1到25，每次循环打印一个 "." 字符，然后暂停80毫秒。这个循环模拟了加载过程，通过延迟来创建动画效果。
  + \*/
* // 显示程序选项的函数
* void options()
  + /\*
  + `options` 函数，该函数用于显示程序的选项菜单。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为选项菜单提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0B")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 接下来，程序打印出程序选项的菜单，包括预订票务、查看已预订的票务列表、通过参考号搜索票务信息、编辑票务详情、取消票务和退出系统。
  + 4. 使用 `getch()` 命令获取用户的输入选择，然后使用 `switch` 语句来处理用户的输入。
  + 5. 根据用户的输入，程序会调用相应的函数来执行相应的操作，如 `bookticket()` 用于预订票务，`list\_booked\_tickets()` 用于查看已预订的票务列表，`search\_ticket()` 用于通过参考号搜索票务信息，`edit\_ticket()` 用于编辑票务详情，`cancel\_ticket()` 用于取消票务，`exit\_terminal()` 用于退出系统。
  + 6. 如果用户输入了无效的选项，程序会显示错误信息，告诉用户只能选择提到的选项，并让用户重新输入选择。如果用户再次输入无效选项，程序会重复显示错误信息，直到用户输入有效的选项为止。
  + \*/
* // 通过参考号搜索票务的函数
* void search\_ticket()
  + `search\_ticket` 函数，该函数用于通过参考号搜索票务信息。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为搜索票务界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 打开 `ticket\_details.txt` 文件以读取模式。如果文件为空，显示一条信息并返回选项菜单。
  + 4. 提示用户输入要搜索的票务参考号。
  + 5. 循环读取文件内容，将票务记录存储在 `booking` 数组中。
  + 6. 关闭文件。
  + 7. 循环检查输入的参考号，如果找到匹配的记录，则打印出票务信息。
  + 8. 返回选项菜单。
  + 这个函数通过允许用户输入参考号来搜索票务信息，提供了一个方便的方式来查找特定的票务记录。如果您对这个函数有任何疑问或需要进一步的帮助，请告诉我！
  + \*/
* //凌勇
* // 用于预订票务的函数
* void bookticket()
  + /\*
  + `bookticket` 函数，该函数用于预订票务。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为预订票务界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 打开文件 `ticket\_details.txt` 以读取现有票务记录，并打开文件以追加新记录。
  + 4. 读取 `ticket\_details.txt` 文件中的票务记录，并将它们存储在 `booking` 数组中。如果文件结束，则跳出循环。
  + 5. 关闭文件。
  + 6. 提示用户输入预订票务的详细信息，包括姓名、联系方式、出发地和目的地。
  + 7. 检查输入的联系方式是否为10位数，如果不是，则提示用户重新输入。
  + 8. 生成新的参考号，并提示用户输入出发地和目的地。
  + 9. 检查出发地和目的地是否相同，如果是，则提示用户重新输入。
  + 10. 检查出发地和目的地是否有效，如果不是，则提示用户重新输入。
  + 11. 询问用户是否确认预订，如果确认，则将新预订的信息追加到 `ticket\_details.txt` 文件中，并关闭文件。
  + 12. 询问用户是否需要继续预订，如果需要，则返回 `bookticket` 函数，重新开始预订过程。
  + \*/
* // 计算数字中的位数的函数
  + int count\_digits\_in\_num(long long number)
  + //这个函数的工作原理是通过不断地将输入的数字除以10，然后递减，直到数字变为0。每次除以10都相当于移除数字的最后一位。因此，递减的次数就是数字的位数。
* // 显示所有预订的票务的函数
* void list\_booked\_tickets()
  + /\*
  + `list\_booked\_tickets` 函数，该函数用于显示所有已预订的票务信息。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为显示票务列表界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 打开 `ticket\_details.txt` 文件以读取模式。如果文件为空，显示一条信息并返回选项菜单。
  + 4. 打印出票务列表的标题。
  + 5. 循环读取文件内容，将票务记录存储在 `booking` 数组中，并打印出每条记录的信息。
  + 6. 关闭文件。
  + 7. 等待用户按键。
  + 8. 返回选项菜单。
  + \*/
* // 通过参考号编辑票务的函数
* void edit\_ticket()
  + //这个函数首先清屏并改变背景颜色，然后打开票务信息文件。如果文件为空，它将显示一条信息并返回选项菜单。如果文件中有票务信息，它将允许用户输入一个参考号来编辑相应的票。
  + /\*
  + `edit\_ticket` 函数，该函数用于通过参考号编辑票务信息。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为编辑票务界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 打开 `ticket\_details.txt` 文件以读取和写入模式。如果文件为空，显示一条信息并返回选项菜单。
  + 4. 提示用户输入要编辑的票务参考号。
  + 5. 将文件指针重新定位到文件开始，并读取票务记录。
  + 6. 循环检查输入的参考号，如果找到匹配的记录，则提示用户输入新的票务信息，如姓名、联系方式、出发地和目的地等。
  + 7. 验证输入的联系方式是否为10位数，如果不是，则提示用户重新输入。
  + 8. 检查出发地和目的地是否相同，如果是，则提示用户重新输入。
  + 9. 检查出发地和目的地是否有效，如果不是，则提示用户重新输入。
  + 10. 询问用户是否确认更新票务信息，如果确认，则将新的票务信息写入 `ticket\_details.txt` 文件中，并关闭文件。
  + 11. 显示更新成功的信息，并显示加载动画。
  + 12. 返回选项菜单。
  + \*/
* //柴东博
* // 根据出发地和目的地提供航班时间的函数
* void flight\_time(char dep[3], char des[3])
  + /\*
  + `flight\_time` 函数，该函数用于根据出发地和目的地提供航班时间。以下是这个函数的详细解释：
  + 1. 函数接受两个字符串参数 `dep` 和 `des`，分别表示出发地和目的地。
  + 2. 使用 `strcmp` 函数比较 `dep` 和 `des` 参数，以确定航班的出发地和目的地。
  + 3. 根据不同的出发地和目的地组合，函数提供相应的航班时间。例如，如果从 北京 出发到 上海，则航班时间为 06:00am 到 06:30am。
  + 4. 如果出发地或目的地无效（即不在预定义的列表中），函数将打印一条错误信息，告诉用户输入的出发地或目的地无效。
  + 这个函数通过提供航班时间信息，为用户提供了一个方便的方式来查询航班的出发和到达时间。如果您对这个函数有任何疑问或需要进一步的帮助，请告诉我！
  + \*/
* // 通过参考号取消票务的函数
* void cancel\_ticket()
  + /\*
  + `cancel\_ticket` 函数，该函数用于通过参考号取消票务。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为取消票务界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 打开 `ticket\_details.txt` 文件以读取和写入模式。如果文件为空，显示一条信息并返回选项菜单。
  + 4. 提示用户输入要取消的票务参考号。
  + 5. 将文件指针重新定位到文件开始，并读取票务记录。
  + 6. 循环检查输入的参考号，如果找到匹配的记录，则显示一条信息表示票务已成功取消，并更新文件中的票务信息。
  + 7. 返回选项菜单。
  + \*/
* // 退出程序的函数
* void exit\_terminal()
  + /\*
  + `exit\_terminal` 函数，该函数用于退出程序。以下是这个函数的详细解释：
  + 1. 函数开始时，使用 `system("cls")` 命令来清除屏幕，以便为退出程序界面提供一个干净的起始点。
  + 2. 使用 `system("COLOR 0E")` 命令来改变控制台背景和字体的颜色，通常用于美化用户界面。
  + 3. 显示一条感谢信息，告诉用户程序即将退出。
  + 4. 等待用户按键。
  + 5. 关闭所有打开的文件，包括文件指针 `fp`。
  + 6. 使用 `exit(0)` 命令退出程序。
  + \*/

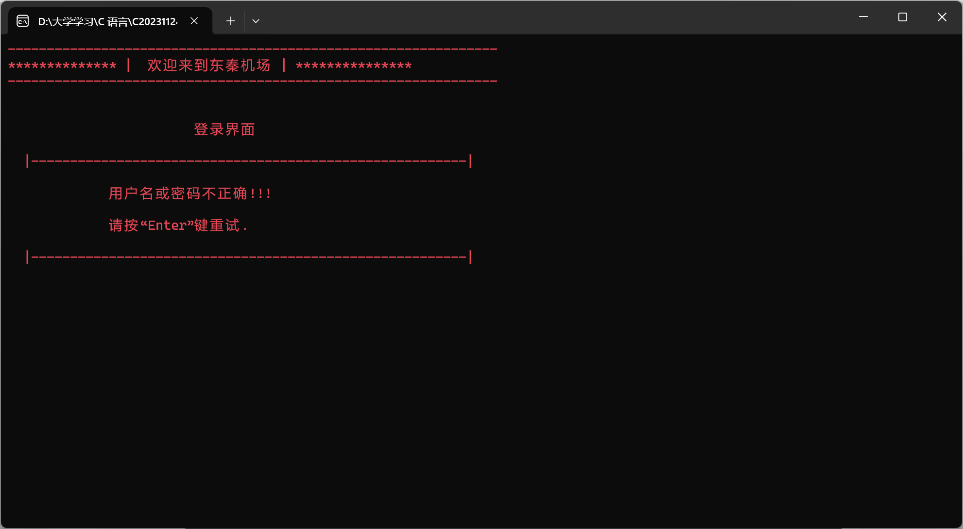
1. **调试与测试**
   1. 自己设计的函数在调试时的测试函数或测试数据与测试结果

* 1. 调试时遇到的问题及解决方法

认证函数代码测试

给出测试结果截图



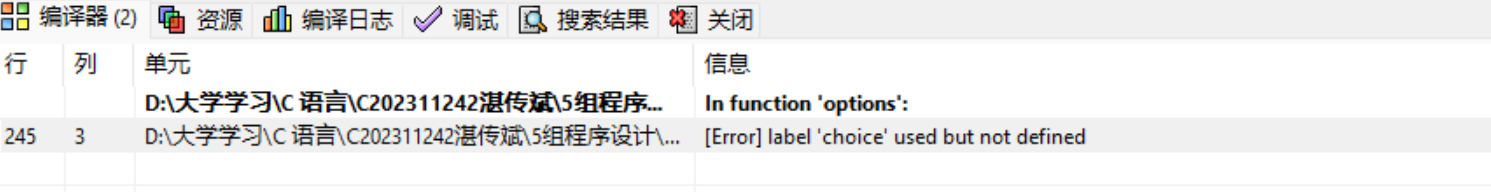
出错原因

由于char username[10]定义为10个字节，而用户名大于10个字节，导致用户名输入时并没有全部存入username中，导致用户名判断时，一直显示错误，无法进入系统。

解决方法

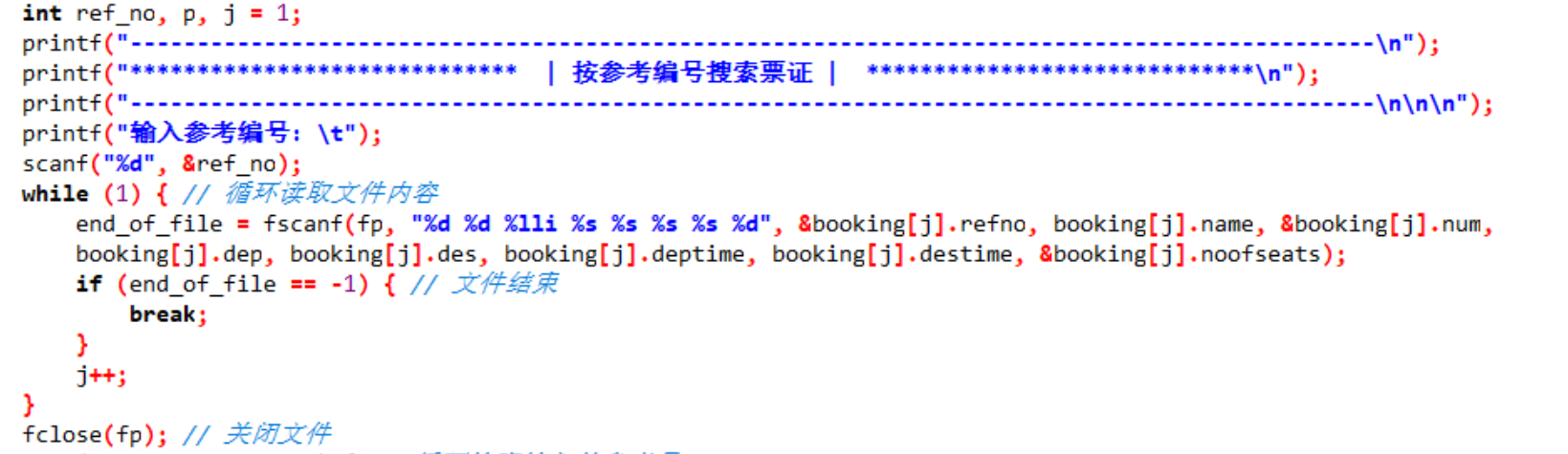
将数组char username[10]改为char username[50]，使输入用户名正常存储。

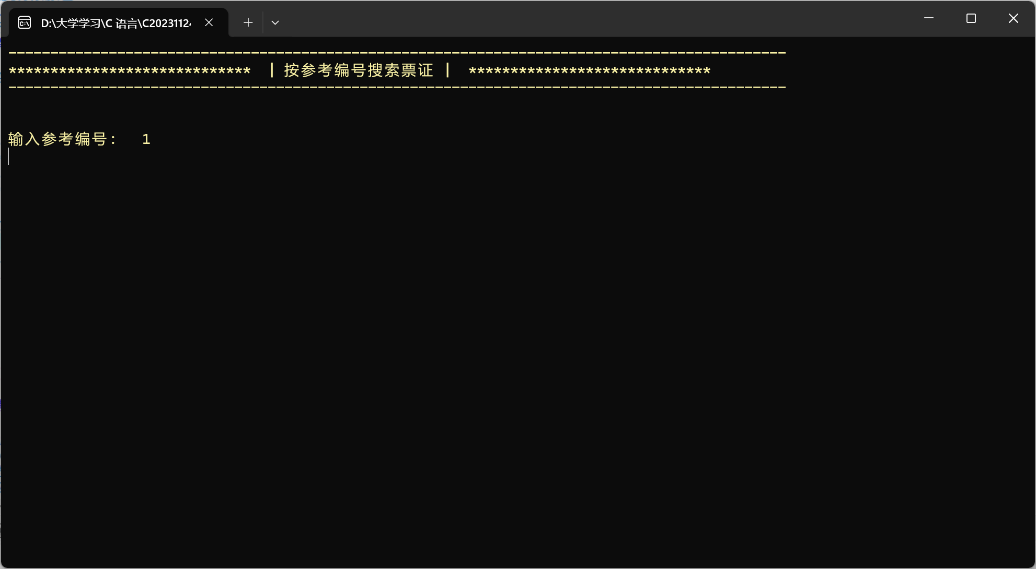
* 编译时提示错误：



* 原因为choice没有事先声明就使用。
* 解决方法在前面程序加入 choice:声明后再使用

通过参考号搜索票务的函数代码测试

问题：输入参考编号后，不输出票证信息，程序没有反应

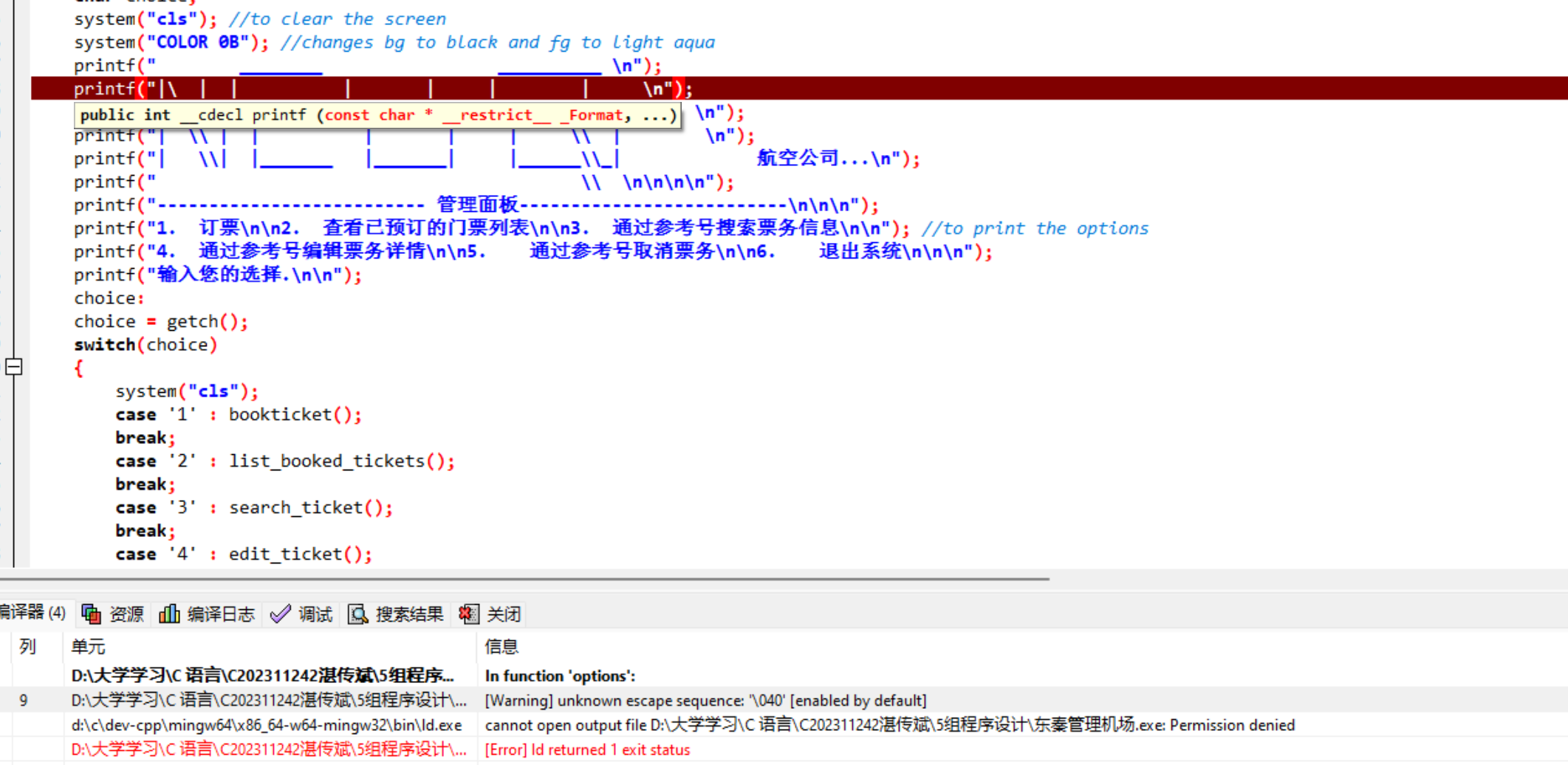


原因：文件读取数据时booking[j].name应为%s，但程序中为%d，数据类型出错，导致无法正常读取数据输出，因此程序卡死。

解决方法：将数据类型改正确

在设计过程中有多次数据类型出错的情况，这里不一一列举，但数据类型要使用正确很重要

问题：符号使用错误



在打出NEUQ时，需要使用\,但是要显示斜杠需要使用两个斜杠才行。

还有一些函数格式也需要注意。

1. **总结**

在设计过程中有多次数据类型出错的情况，程序编译过程不会报错，但是程序运行不输出正确结果，导致很难发现程序出错原因，所以数据类型要使用正确很重要

其次，在使用中英文输入法时要注意切换，不小心打了中文符号，就会报错。

在使用相关符号时，要注意使用的格式正确，比如：在打出NEUQ时，需要使用\,但是要显示斜杠需要使用两个斜杠才行。还有一些函数格式也需要注意。

另外，定义变量时要注意需要的字节数，以免存储大小不够。还要注意变量的作用范围，是全局变量还是函数内变量。

指针的使用也要注意指针所在位置。