**西 安 邮 电 大 学**

**（计算机学院）**

数据结构课程设计报告

# 题 目： 校园导游系统

**专业名称： 计算机科学与技术**

**班 级： 计科**1701

**学生姓名： 雷洁萍**

**学号（8位）： 04171018**

**指导教师：** 王春梅

**设计起止时间：** 2018年12月31日—2019年1月4日

## 一. 设计目的

校园导游系统是针对用户和管理员使用的。对于管理人员，要有权限对信息根据实际情况进行一定的修改，如增加新的景点、增添两景点间的路线和删除已经不存在的景点、对不再有的路线进行删除，以及得到图中所有地点的最佳布网方案，需要对信息的变更及时修改。另一方面对于用户来说，通过校园导游系统，可以查询校园的俯瞰图，了解校园景点的大致位置分布情况，要可以对校园景点的信息进行查询，可以查询两地点之间的所有简单路径、距离最短路径、中转次数最少的路径等，并且可以留下对校园景点的评价以供其他人参考。

通过对代码的实现，更深层次的理解了图的相关算法，并且巩固了文件方面的相关知识。在该过程中提高了编程水平，了解到一个完善的系统要能应对实际中的各种问题，代码要有健壮性。

## 二. 设计内容

1.登录模块：实现了管理员和用户登录的功能。只有当管理员输入正确的账号和密码时，才有权限进行其他操作，否则系统会提示账号或密码输入错误，提示管理员重新登录。而用户也需要输入相应正确的信息才能进入系统，否则系统会提示输入信息有误，需要重新登录。登录成功之后会在界面上显示一个主菜单，管理员或者用户可以根据主菜单上的提示信息，选择所需的功能，进而进行下一步操作。

2.查询模块：实现了多种查询功能。首先可以查询校园的全景图，对校园景点有一个大致的了解；还可以对相应的景点查询，可以查询某个具体景点的详细介绍，也可以查询之前用户对各个景点的好评指数；其次可以查询两景点之间的路线，可以选择查看两地之间的所有路径、中转次数最少的路径以及距离最短的路径，用户从而根据喜好选择地点和路径。对于管理员而言，还可以查看校园所有景点的最佳布网方案。

3.插入模块：当以管理员的身份进入系统时，可以实现根据实际情况添加新景点和增加两地之间新路线的功能，以便系统实时更新。

4.删除功能：当以管理员的身份进入系统时，可以根据实际情况撤销已经不存在的地点，删除两点之间已不再有的旧路线，以便系统及时更新。

6.评价模块：用户可根据个人体验，对不同的景点进行评价，以供之后的用户作以参考。

## 三．概要设计

首先通过登录模块，区分登录者的身份，是管理员还是用户。登录成功之后显示主界面，提示用户下一步可进行的操作，用户可根据主菜单选择想要实现的功能，如查询校园全景图、校园景点信息、两景点之间的路径和评价景点等。通过该系统，管理员可以实现对景点的添加和删除、对两地点之间新路径的增添和旧路径的撤销，以便根据实际情况及时对系统进行更新，并且还可以查看校园所有景点的最佳布网路线等。

### 1．功能模块图；

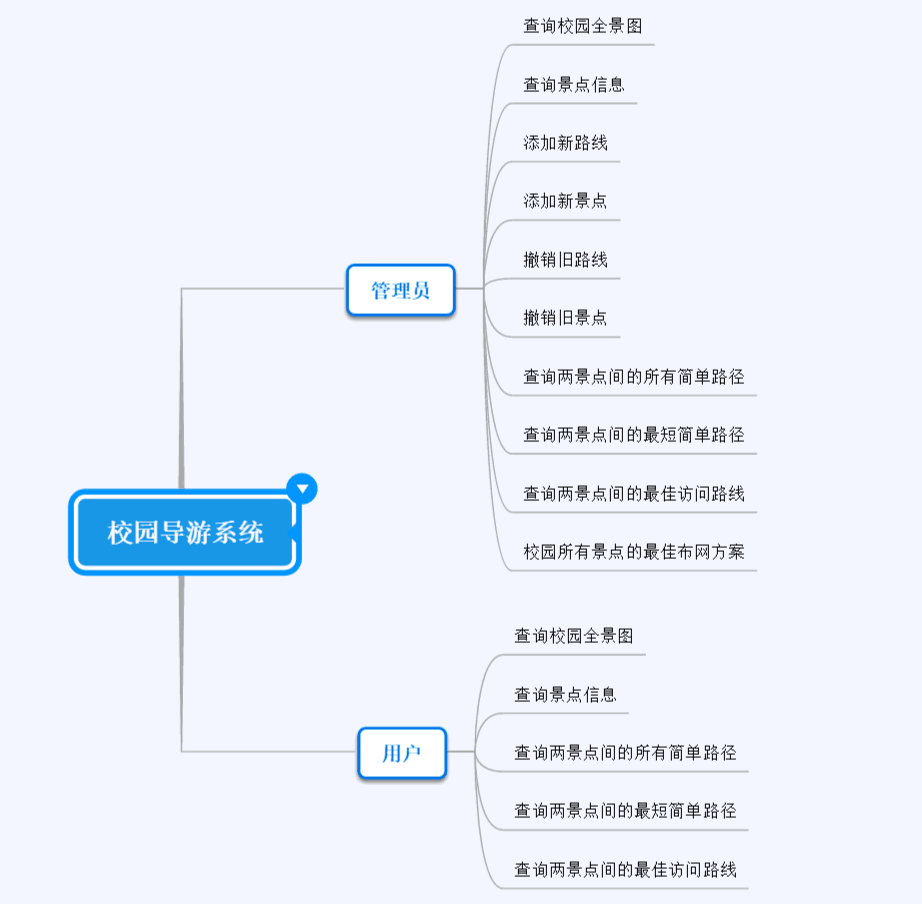


图1

### 2．各个模块详细的功能描述。

（1）Boundary\_layer(); 函数用于用户和管理员登录和显示主界面，只有当输入正确的信息时，才可以进行其他操作，否则系统会提示信息输入错误，要求重新登录。登录成功之后会在界面上显示一个主菜单，用户可以根据主菜单上的提示信息，选择所需的功能，进而进行下一步操作。

（2）Map(); 函数的功能是显示校园地图。

（3）Search(); 函数的功能是查询景点信息，可以选择查询景点星级排名情况，也可以选择查询景点详情介绍。

（4）AddNewRoute(); 函数的功能是添加新路线，以便管理员根据实际情况增添两景点之间的路线。

（5）AddNewSence(); 函数的功能添加新景点，以便管理员根据校园景点的变动而添加新的景点。

（6）DeleteOldRoute(); 函数的功能是删除旧路线，以便管理员根据实际情况删除校园中已经不再有的路线。

（7）DeleteOldSence(); 函数的功能是删除旧景点，以便管理员删除校园中已经不存在的景点。

（8）SearchAllRoute(); 函数的功能是查询所有简单路径，用户可以查询任意两个景点间的所有简单路径。

（9）Minsimpleroute(); 函数的功能是查询最短简单路径，用户可以查询任意两个景点之间中转次数最少的路径。

（10）Shortcut(); 函数的功能是查询最佳访问路线，用户可以查询任意两景点之间距离最短的路径。

（11）BestRoute(); 函数用于生成校园景点的最佳布网方案，当以管理员的身份进入系统时，可以得到一个校园景点的最佳布网方案。

（12）Satisfied(); 函数的功能是调查用户满意度，用户可以留下自己对某个景点的评价，以供之后的用户做一个参考。

（13）Welcome(); 函数的功能是显示一个进入系统的界面。

（14）Menuout(); 函数的功能是显示一个退出系统的界面。

## 四．详细设计

### 1．功能函数的调用关系图

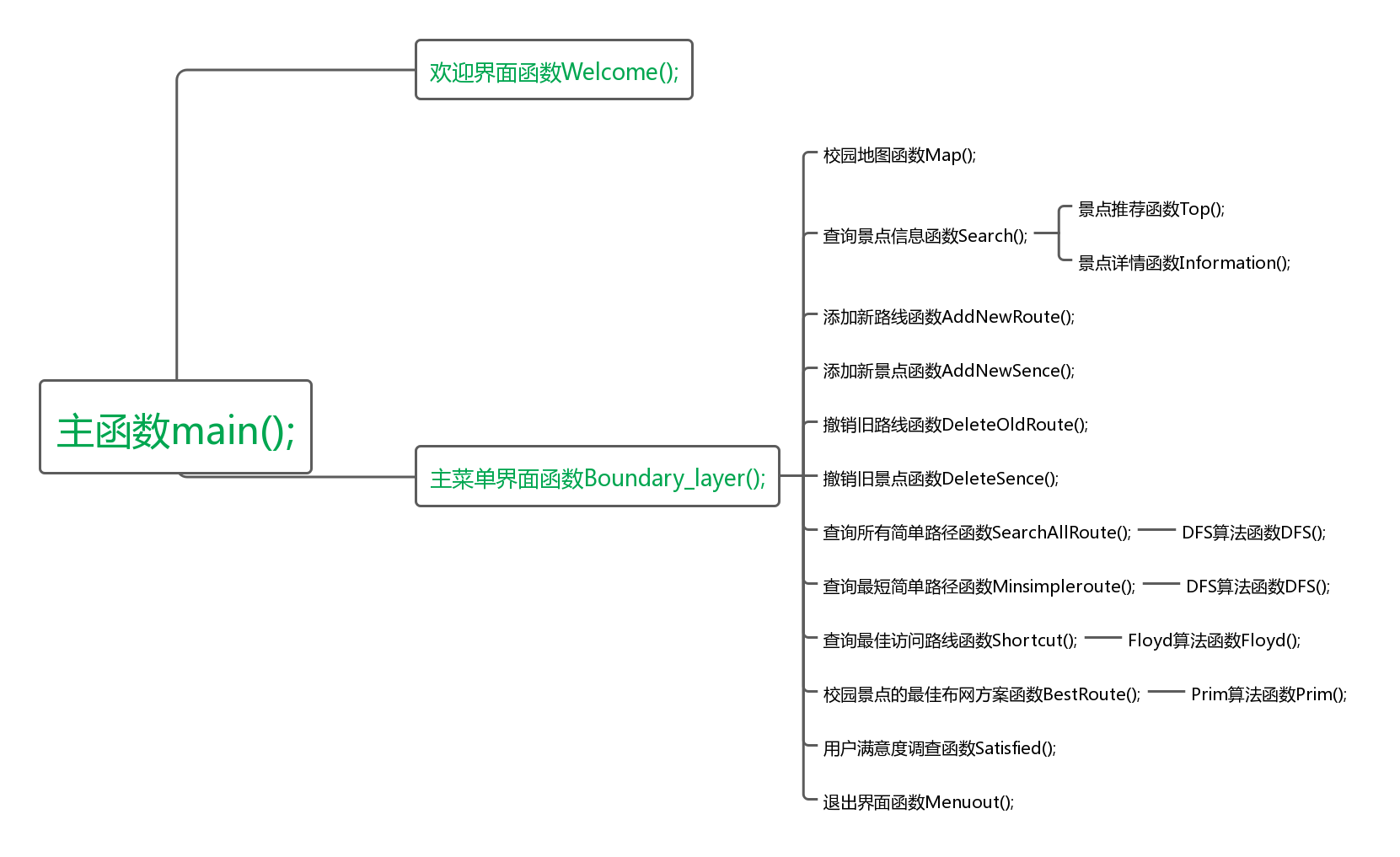


图2

### 2．各功能函数的数据流程图

（1）总系统流程图

开始

登录（输入账号和密码）

否

账号、密码是否正确

是

否

是否是管理员

是

选择相应的功能

1、查询校园全景图

2、查询景点信息

3、添加新路线

4、添加新景点

5、撤销旧路线

6、撤销旧景点

7、查询两地间所有简单路径

8、查询两地间最短简单路径

9、查询两地间最佳访问路线

10、所有景点的最佳布网方案

0、退出系统

请选择（0-10）

用户

选择相应的功能

1、查询校园全景图

2、查询景点信息

3、查询两地间所有简单路径

4、查询两地间最短简单路径

5、查询两地间最佳访问路线

6、填写景点评价

0、退出系统

请选择（0-6）

否

进行相应功能的实现

否

进行相应功能的实现

是否输入0

是否输入0

是

是

结束

图3

（2）登录系统函数流程图

开始

登录(输入账号和密码)

否

账号、密码正确

是

登陆成功

结束

图4

（3）查询景点信息函数流程图

开始

选择查询方式

是

是否查询景点推荐信息

是

否

否

是否查询某个景点详情

输入错误信息

输入查询信息

输出查询结果

结束

图5

（4）查询路径信息函数流程图

开始

选择查询方式

查询两地间所有简单路径

是

图6

结束

是

输出查询结果

输入查询信息

否

查询两地间最短简单路径

否

是

查询两地间距离最短路径

否

是

查询所有景点最佳布网方案

否

输入信息错误

（5）添加新路线函数流程图

开 始

否

是否是管理员

是

输入路径起点、终点信息

图的边数加一

保存至文件

结束

图7

（6）添加新景点函数流程图

开 始

否

是否是管理员

是

输入新添景点名称等信息

图的顶点数加一

保存至文件

结束

图8

（7）撤销旧路线函数流程图

开 始

否

是否是管理员

是

输入路径起点、终点信息

图的边数减一

保存至文件

结束

图9

（8）撤销旧景点函数流程图

开 始

否

是否是管理员

是

输入景点信息

图的顶点数减一，相应边数减少

保存至文件

结束

图10

（9）用户满意度调查函数流程图

开 始

输入景点信息

进行评价

否

评价星级比系统中高

是

更改系统中星级指数

保存至文件

结束

图11

### 3．重点设计及编码

(1)DFS深度遍历查找路径DFS(int start,int ending,int stack[],int visited[]);

用递归的方法，从某个顶点出发，访问此顶点，然后依次从该顶点的各个未被访问的邻接点出发深度优先搜索遍历图；将走过的路径存入数组中，当找到终点时输出路径。该函数主要用于查找两景点之间的所有简单路径以及两景点之间中转次数最少的路径。

代码：

int m=1,min=INT\_MAX;

int path[20]={0};

int tag;

void DFS(int start,int ending,int stack[],int visited[])

{

Readfile();

int i, j;

for(i=1; i<=G->vexnum; i++)

{

if(G->arcs[start][i] != Infinity && visited[i]==0)

{

if(i == ending){ //如果深搜到了终点，就输出刚才经过的路径；

if(tag==1){ // tag用于判断找最优路径还是所有路径,tag=1时找所有路径；

printf("\n\t\t\t\t");

for(j=0; j<m; j++)

printf("%s-->", G->vex[stack[j]].name);

printf("%s\n", G->vex[ending].name);

getch();

}

else //找最优路径时记录次数最少的路径；

if(m<=min){

min=m;

for(j=0;j<m;j++)

path[j]=stack[j];

}

}

else{ //如果该点不是终点；

visited[i]=1;

stack[m] = i;

m++;

DFS(i,ending,stack,visited);

visited[i]=0;

m--;

}

}

}

}

1. Floyd算法求得任意两顶点之间的最短路径Floyd(int start,int end);

代码：

void Floyd(int start,int end)

{

int F[Max][Max];

int path[Max][Max];

int i,j,k,m,temp;

for(i=1;i<=G->vexnum;i++)

for(j=1;j<=G->vexnum;j++)

F[i][j]=Infinity;

for(i=1;i<=G->vexnum;i++)

for(j=1;j<=G->vexnum;j++)

{

if(G->arcs[i][j]!=Infinity)

{

F[i][j]=G->arcs[i][j];

}

if(G->arcs[j][i]!=Infinity)

{

F[i][j]=G->arcs[j][i];

}

if(i!=j)

path[i][j]=j;

else path[i][j]=0;

}

for(k=1;k<=G->vexnum;k++)

{

for(j=1;j<=i;j++)

{

for(i=1;i<=G->vexnum;i++)

if(F[i][j]>F[i][k]+F[k][j])

{

F[i][j]=F[i][k]+F[k][j];

F[j][i]=F[i][k]+F[k][j];

path[i][j]=path[i][k];

}

}

}

printf("\n\t\t\t\t从%s到%s的最短路径为：\n\n\t\t\t\t从%s",G->vex[start].name,G->vex[end].name,G->vex[start].name);

temp=path[start][end];

while(1){

printf("----->%s",G->vex[temp].name);

if(temp == end) break;

temp=path[temp][end];

}

printf("\n\t\t\t\t距离%dkm",F[start][end]);

getch();

}

（3）prim算法求得最小生成树；

从图中的某一顶点v0出发，选择与它关联的具有最小权值的边，将其顶点加入到生成树的顶点集合中，以后的每一步从一个顶点在生成树中，而另一个顶点不在生成树中的各条边中选择权值最小的边，再把它的顶点加入到生成树顶点集合中，直到所有顶点都加入到生成树集合中。

void Prim(int start)

{

Readfile();

int i,e,k,m,min,j=1,sum=0;

int s[Max];

s[0]=start;

closedge[start].lowcost=0;

for(i=1;i<=G->vexnum;i++)

if(i!=start)

{

closedge[i].adjvex=start;

closedge[i].lowcost=G->arcs[start][i];

}

for(e=1;e<=G->vexnum-1;e++)

{

min=Infinity;

for(k=1;k<=G->vexnum;k++)

{

if(closedge[k].lowcost!=0 && closedge[k].lowcost<min)

{

m=k;

min=closedge[k].lowcost;

}

}

sum+=closedge[m].lowcost;

closedge[m].lowcost=0;

s[j++]=m;

for(i=1;i<=G->vexnum;i++)

if(i!=m && G->arcs[m][i]<closedge[i].lowcost)

{

closedge[i].lowcost=G->arcs[m][i];

closedge[i].adjvex=m;

}

}

printf("路径长度：%dkm\n\n",sum);

for(i=0;i<G->vexnum-1;i++)

{

printf("%s----->",G->vex[s[i]].name);

}

printf("%s",G->vex[s[i]].name);

getch();

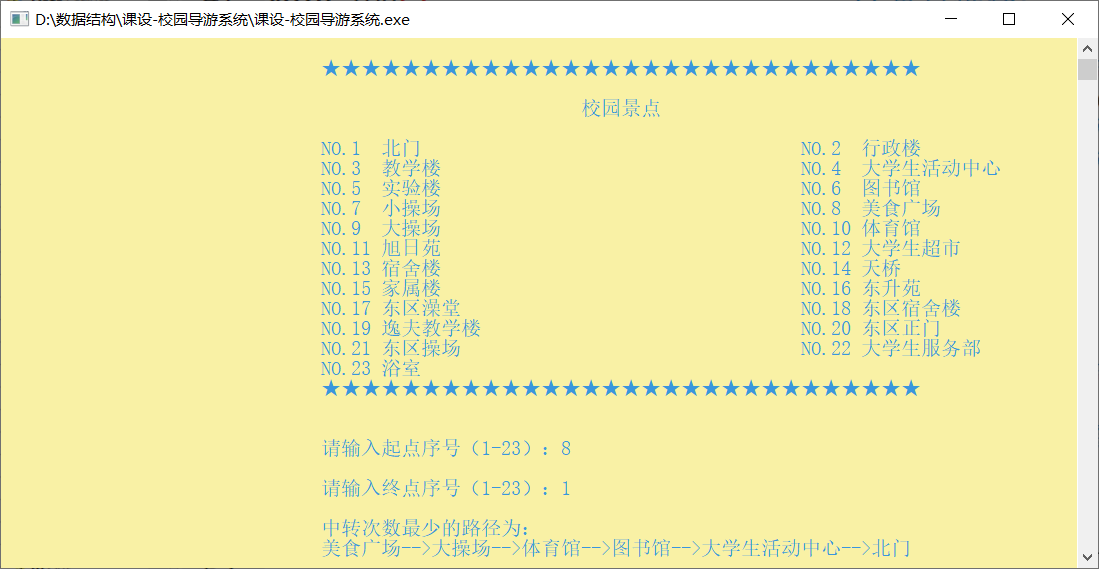
}

图12

## 五．测试数据及运行结果

### 1．正常测试数据和运行结果

（1）查询模块：登录成功之后，先输入4进入到两景点之间最短简单路径查询模块，再输入起点序号8，终点序号1进行查询；



（2）添加模块：管理员登录成功之后，先输入4进行添加新景点的功能，在输入景点的详细信息；

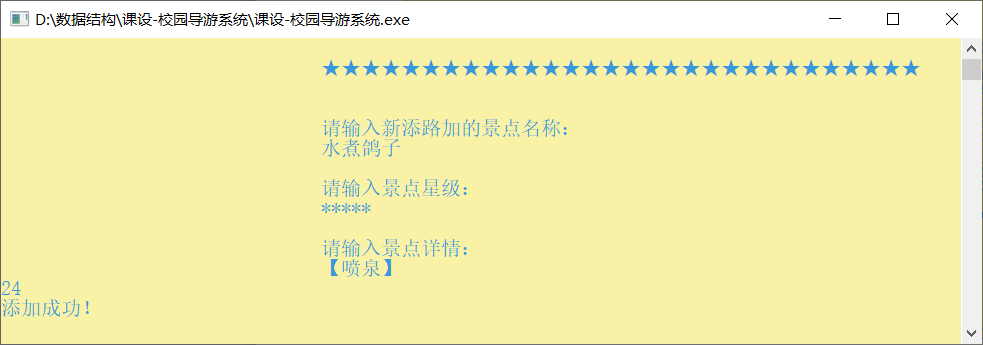


图13

1. 删除模块：管理员登录成功之后，先输入5进入删除旧路线的模块，再输入要删除路线的起点和终点序号；



图14

### 2．异常测试数据及运行结果

（1）当登录时，如果不按要求输入账号信息，系统会提示输入有误，要求重新输入；

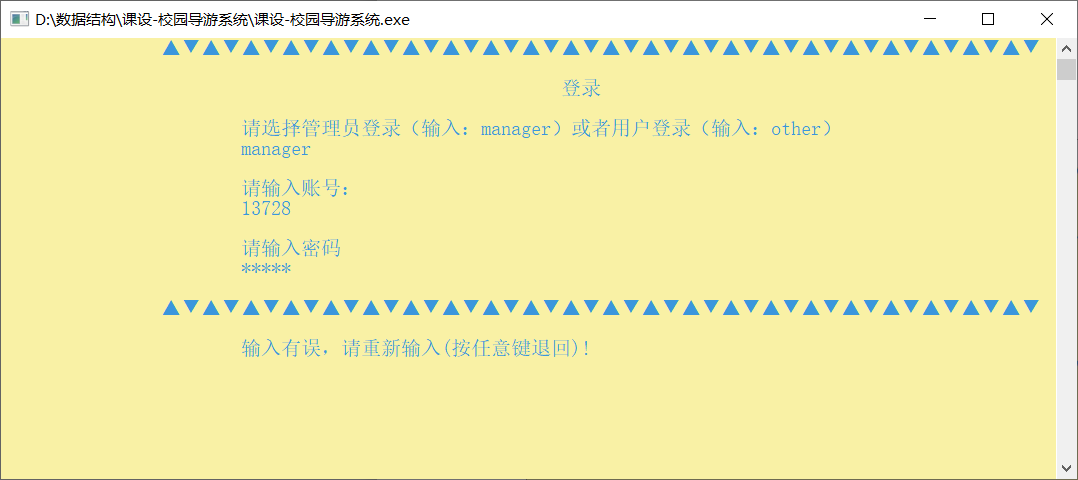


图15

（2）在查询景点信息时，若不按要求输入，系统会提示输入信息错误；

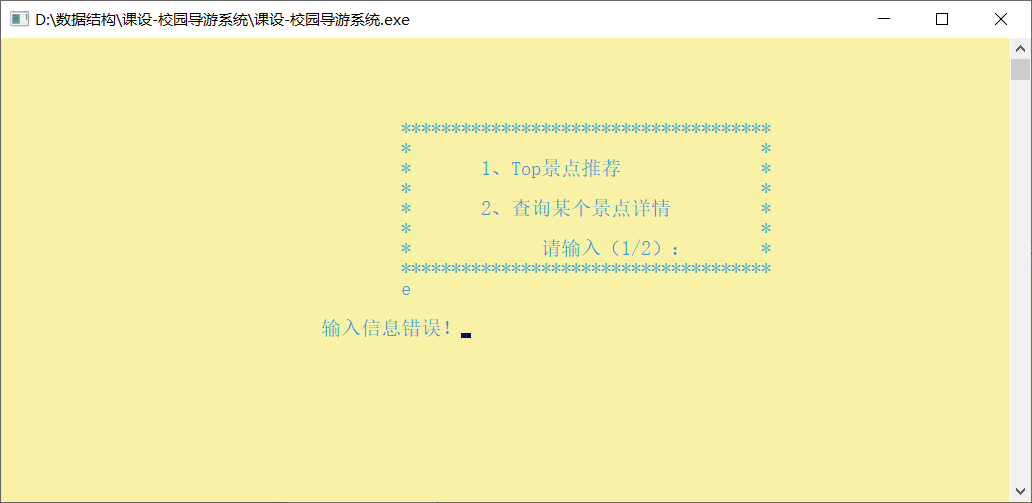


图16

## 六．调试情况，设计技巧及体会

### 1．改进方案

我所做的校园导游系统具有较好的人机交互界面，每一步操作都有相应的提示，用户可根据提示进行每一步操作。在进入系统之前，有一个登录操作，只有输入正确的信息才能进入校园导游系统进行功能选择，否则系统会提醒信息输入错误，要求重新输入。以不同的身份登录时所具有的权限有所差别，例如管理员有权限进行景点、路线的增删，查看所有景点的最佳布网方案等，而用户只能查询景点信息、路线信息、进行景点评价等。在输入不符合要求的信息时，系统会给出相应的提示，提醒用户输入信息错误，请用户重新输入。

不足之处在于没有用邻接表实现图的各种功能，导致在进行景点删除等操作时，时间复杂度相比较高。

### 2．体会

写代码过程中应该考虑到哪些过程是重复的，可以把这些过程写成一个函数，在写其他函数的时候可以直接调用，不用每一次都写一遍，会造成程序过于繁琐，应及时为代码写必要的注释，不仅有利于他人的阅读，也有利于自己以后的查看，并且要注意如何使修改的信息与文件中保持一致。还要考虑到输入错误信息的情况，不应该直接退出系统，而应该退到上一层中去。

调试的时候，尽量选择部分进行调试，不要整体调试代码。

## 七．参考文献

[1]《C语言程序设计教程》

[2] 百度文库

[3]《数据结构与算法》

1. **备注（修改后实现功能）**
2. 评价改进：用Satisfied();函数实现了用户评价某个景点星级的功能，该函数将当前用户评价的星级指数与之前系统里该景点的星级指数相比较，保留星级高的指数存入到文件中。
3. 密码用Backspace键可以进行回删，在Boundary\_layer();函数中实现了该功能。
4. 删除旧景点：用DeleteOldSence();函数实现了撤销旧景点的功能，同时也撤销旧景点的相关路线。
5. 校园中所有景点的最佳布网方案路径打印：用prim();函数实现了计算路径距离以及路径打印的功能。
6. 错误异常处理：在登录时如若输入信息有误，系统会提示输入信息错误，并需要重新输入。并且大部分函数实现了对错误异常的处理，在输入错误信息时提示输入信息有误。