****

**J I A N G S U U N I V E R S I T Y**

**数据结构与算法课程设计**

所属学院：计算机科学与通信工程学院

专业班级：

姓 名：

学 号：

指导教师：

2021年6月

目录

[1.问题分析和任务定义 1](#_Toc31879)

[1.1 任务分析 1](#_Toc3645)

[1.2 任务要求 1](#_Toc1707)

[2．数据结构的选择和概要设计 1](#_Toc18336)

[3.系统的详细设计 2](#_Toc11571)

[3.1初始化系统 2](#_Toc19866)

[3.2 公交信息维护 3](#_Toc22963)

[3.2.1 线路的新增 4](#_Toc2923)

[3.2.2 线路的删除 6](#_Toc23143)

[3.2.3 修改原有路线 7](#_Toc20085)

[3.3 菜单的设置以及各个菜单之间的链接 13](#_Toc26357)

[3.4 查询系统 15](#_Toc14440)

[3.4.1所有公交信息的显示 15](#_Toc18533)

[3.4.2特定条公交信息的显示 16](#_Toc19424)

[3.4.3 换乘查询以及旅游路线规划 17](#_Toc9679)

[4. 小结 21](#_Toc11983)

[5. 参考文献 21](#_Toc17299)

[6.附录（源代码） 21](#_Toc22897)

# 1.问题分析和任务定义

## 1.1 任务分析

在日常生活中，公交换乘十分普遍，但也至关重要，公交换乘关系着居民出行时间的长短，出行的便利与否。而根据参考因素，例如换乘路线的路径最短、耗费时间最短、所需车资最少等因素，经过分析来达到可达目的站点换乘次数最少的乘车方案，具体可分为：

（1）零次换乘 起始站点和目的站点之间存在可直达的公交线路，即出行居民无需转乘就可以直接到达目的站点，这也是较为理想的方案。

（2）一次换乘 起始站点和目的站点之间没有公交车直接往返，即两站点之间不存在可直达的公交线路，则出行居民需要在途经的某个站点下车，然后转乘另一线路公交车才能达到目的站点。

（3）多次换乘 在起始站点和目的站点之间没有可直达的公交线路，出行居民需要经过一次以上的转乘才能达到目的站点，则得到多次换乘方案。

## 1.2 任务要求

1) 初始化系统：

基本信息初始化，包括公交线路编号、站点名称、站点之间票价、耗费时间、发车时间等；

基本要求：将公交线路的基本信息存于文件，每次运行程序时，通过文件对基本信息初始化；

2) 公交信息维护：

由于修路等城市规划变动的原因，公交线路可能会有微调，例如需要增加、删除或者修改部分站点的信息；修改之后的课程信息写回文件，便于下一次初始化；

3) 查询功能：

* 从键盘输入编号，给出具体行车路线、行车时间等；
* 从键盘输入始发站和终点站，完成公交线路换乘查询（零次，1次，多次）
* 旅游线路规划：金山、焦山、北固山、南山风景区，为用户定制最优各景点间游玩路线。

4）用户角色：

分为管理员和游客，管理员可以进行初始化、公交信息维护和查询功能；游客只可以使用查询功能。

5）菜单：

要求有菜单，菜单要根据自己的功能定义来设计，需要增加子菜单，要通过循环让线路显示，并让用户根据需求进行相应的操作，直到用户选择退出咨询系统位置。

6）其他：

因为本程序只需要体现大致思想，所以仅仅设置了初始的五条线路和25个站点用于体现设计思想和实现功能。

# 2．数据结构的选择和概要设计

代码中主要选用了结构体的数组用于存放公交车的信息，并且运用图来显示公交站点之间的距离和信息。并且根据后续增加线路等操作来编写。其具体结构为：

struct station //途径站点信息

{

char c[20]; //站点名称

};

typedef struct bus //公交车信息

{

char num[20]; //公交车编号

int n; //站台数目

struct station name[12]; //站台名称

char topen[20]; //起始时间

char tclose[20]; //终止时间

int money; //票价

}BUS;

BUS bus[MAX];

以上是公交车的具体信息比如起始时间，公交编号，票价，沿途站点等。下面是各个站点之间图的距离（此处距离纯粹是依据于公交线路而定，若两站点之间无公交线路经过，则认为两公交站点之间的距离为INFINITY。在程序开头宏定义INFINITY的值为99999999。具体定义为：

typedef struct

{

int name; //站点名称

int lines[6]; //途径路线

}Vertex;

typedef struct

{

Vertex vexs[MAX]; //站点

int arc[MAX][MAX]; //邻接矩阵

int numVertexes, numEdges; //顶点数和边数

}MGraph;

MGraph \*G;

公交车的信息通过上述的结构体数组修改，而查询功能要求的换乘以及所需最短路径，则用floyd算法实现。

# 3.系统的详细设计

## 3.1初始化系统

先在电脑中创建一个txt文本，名为“bus.txt”。将初始的线路信息输入该txt文本，该部分代码为：

void init() //初始化函数

{

FILE\* fp; //文件指针

int i;

if ((fp = fopen("bus.txt", "r+")) == NULL) //初次尝试打开"bus.txt"文本文件

{

printf("\n\t\t文件打开失败\n\n\t\t正在尝试创建新文件...\n");

fp = fopen("bus.txt", "w"); //创建"bus.txt"文本文件

if ((fp = fopen("bus.txt", "r+")) == NULL) //再次判断是否打开成功

{

printf("\t\t文件创建失败!!!\n");

return;

}

}

fp = fopen("bus.txt", "r+");

for (i = 0; i < MAX; i++) //将磁盘中的信息输出到内存中

if (fread(&bus[i], sizeof(struct bus), 1, fp) != 1)

break;

fclose(fp);

printf("\n\t\t初始化完成!!!\n\n");

return;

}

若成功打开，则会显示出如下图样，则说明初始化成功：



图 3.1.1

在“bus.txt”文本中的公交线路如下图3.1.1所示，将其存入代码的结构体数组中：

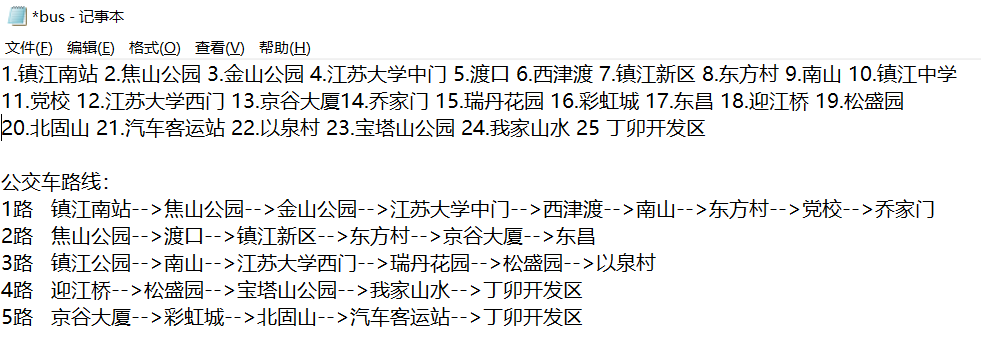


图3.1.2

则初始的公交线路信息就已经存入代码的结构体数组中。而图中的距离则用用一个初始化函数，将各个站点之间的距离输入即可。

## 3.2 公交信息维护

由于城市路线的变动，公交站点会发生变化，公交路线也会增加站点或者删除站点，公交车的名称，票价等等均有可能发生改变，所以身为系统线路的管理员，我们势必要写功能用于公交线路的修改。

其菜单如下：

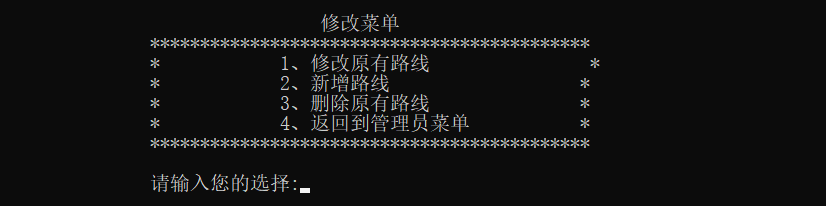


图 3.2.1

### 3.2.1 线路的新增

线路的新增用了一个自定义函数add（），通过输入新的公交线路的信息，把公交线路的信息存在最开始的结构体的数组内，具体代码如下：

void add()

{

FILE\* fp = NULL; //文件指针

int i, j;

char cc[20];

for (i = 0; i < MAX; i++)

{

if (\*bus[i].num != '\0')

continue;

else

{

printf("\n添加第%d辆公交车路线记录:\n", i + 1);

printf("\n请输入路线编码:\n"); //路线编码

scanf("%s", cc);

for (j = 0; j < MAX; j++) //检验是否重复

if (strcmp(bus[j].num, cc) == 0)

{

printf("\n与已有路线编码重复，按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

strcpy(bus[i].num, cc);

printf("\n请输入途径站台总数(>=2): "); //站台总数

scanf("%d", &bus[i].n);

printf("\n");

if (bus[i].n < 2 || bus[i].n>12)

{

while (bus[i].n < 2 || bus[i].n>12)

{

printf("\n站台总数应满足(2<=n<=25)，请重新输入: ");

scanf("%d", &bus[i].n);

printf("\n");

}

}

for (j = 0; j < bus[i].n; j++)

{

printf("请输入第%d个站台名称 ", j + 1); //站台名称

scanf("%s", bus[i].name[j].c);

}

printf("\n自动生成公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

{

printf(" ----> (%d) %s", j + 1, bus[i].name[j].c);

}

printf("\n\n请输入公交车的起始时间: "); //起始时间

scanf("%s", bus[i].topen);

printf("请输入公交车的终止时间: "); //终止时间

scanf("%s", bus[i].tclose);

printf("\n请输入公交车的票价: "); //票价

scanf("%d", &bus[i].money);

printf("\n第%d辆公交车路线记录创建成功!!!\n", i + 1);

save();

printf("\n\t按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

}

if (i == MAX)

printf("\n\n\n\t空间已满，不能录入!!!\n");

printf("\n\t按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

运行截图如下所示：

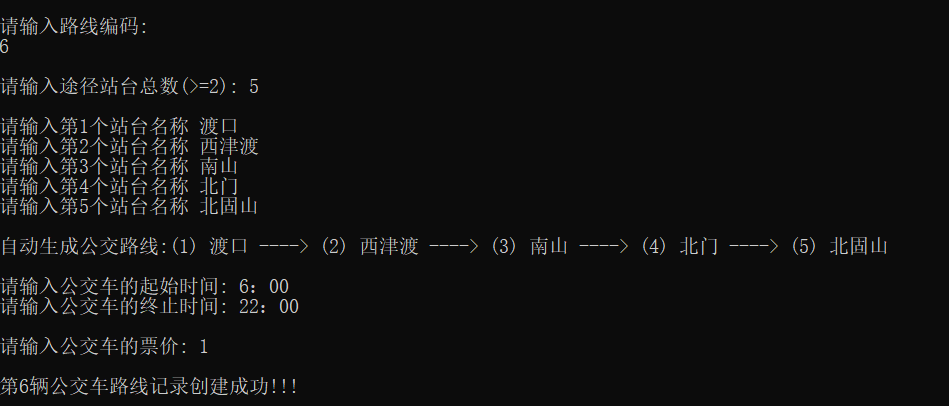


图 3.2.2

这样一条新的线路就增加好了。

### 3.2.2 线路的删除

线路的删除是运用memset函数直接清空，然后再将删除路线的下一条路线移动到删除的路线的数组中。是把后面的公交信息补上删去的公交车信息。在设计中，我们先输入要删除的公交线路的编号，比如1路车。然后该条公交线路的信息会显示出来，供管理员判断这条线路是否是要删除的线路。若确认该条线路将要删除，则按1，若不是则按除了1以外的其他字符。具体代码如下：

void delete() //信息删除函数

{

int i, j, t, m = 0;

char cc[20];

printf("\n请输入要删改信息的公交车路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++) //查找所输入的公交车

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n已找到:\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

printf("即将删除该公交车信息!!!\n确定请按1：");

scanf("%d", &t);

if (t == 1)

{

if (i + 1 == MAX)

memset(&bus[i], '\0', sizeof(struct bus)); //最后一辆直接进行清空

else

for (j = i; \*bus[j].num != '\0' && j + 1 < MAX; j++) //后面的公交车信息补上删去的公交车信息

bus[j] = bus[j + 1];

printf("编号为%s的公交车信息删除完毕!\n", cc);

save();

break;

}

}

}

if (m == 0)

printf("\n\n\t\t无此公交车信息!!!\n");

printf("\n\n\t输入回车键返回主菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

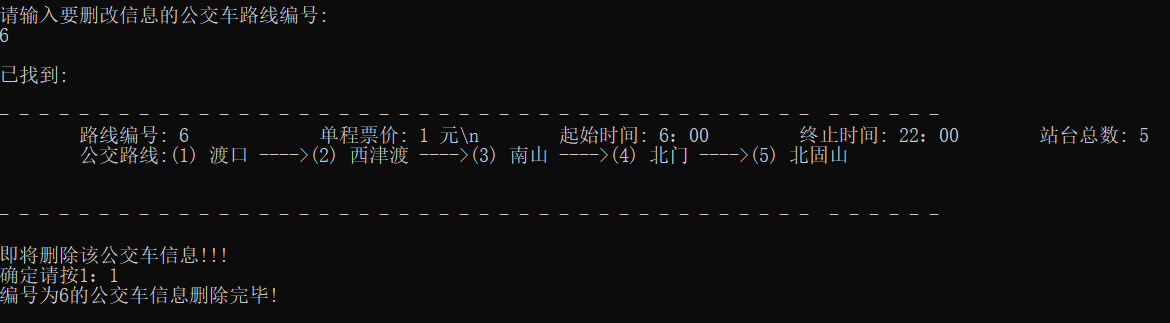
具体运行截图如下图所示：

图 3.2.3

可以看到，在删除后已经没有编号为6的公交线路的信息了。

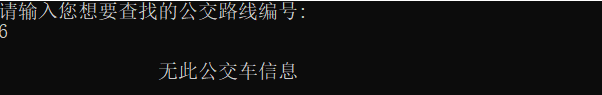


图 3.2.4

### 3.2.3 修改原有路线

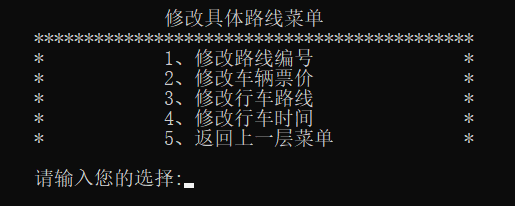
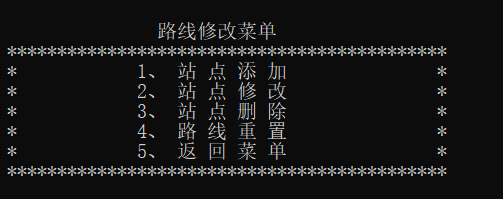
修改原有路线信息可以选择修改公交路线的编号，可以修改票价、时间等简单的信息，也要可以修改原经过的站点的功能。其修改的菜单如下，图3.2.6是图3.2.5的子菜单：

图 3.2.6

图 3.2.5

修改行车编号，票价，时间，线路等是比较简单的操作，只需找到对于路线的数组，进行修改即可。而其对应菜单的操作，只需要使用switch函数即可。具体代码如下图所示：

void mod()

{

int i, j, m = 0;

char t, cc[20], mod[20];

printf("请输入要修改信息的公交车路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++) //查找所输入的公交车

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n\n\t\t已找到!!!\n");

while (1)

{

system("cls");

printf("\n\n 正在修改的公交车信息\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

modload\_menu();

fflush(stdin); //清除键盘缓冲区

t = Test(getchar(), '1', '5'); //菜单检测输入函数

//system("cls");

switch (t)

{

case '1': //修改路线编号

{

int k;

printf("\n请输入新的公交车编号:");

scanf("%s", mod);

for (k = 0; \*bus[k].num != '\0' && k < MAX; k++)

{

if (strcmp(bus[k].num, mod) == 0)

{

printf("与已有编号重复,按回车键返回主菜单\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

}

strcpy(bus[i].num, mod);

save();

break;

}

case '2': //修改车辆信息

{

printf("\n请输入新的公交车票价:");

scanf("%d", &bus[i].money);

save();

break;

}

case '3': //修改行车路线

{

void modstation\_menu(); //路线修改菜单

int k;

char z;

while (1)

{

//system("cls");

modstation\_menu();

fflush(stdin); //清除键盘缓冲区

z = Test(getchar(), '1', '6'); //菜单检测输入函数

//system("cls");

switch (z)

{

case '1': //添加站点

{

if (bus[i].n + 1 > 12) //判断是否满足条件

{

printf("\n站台总数达到25个，无法添加新站点\n\n按回车键返回\n");

fflush(stdin); //清除键盘缓冲区

getchar();

system("cls");

return;

}

printf("\n请输入需要添加第几个站点：");

scanf("%d", &k);

while (bus[i].n + 1 < k)

{

printf("目前共%d个站点，无法添加第%d个站点\n请重新输入：", bus[i].n, k);

scanf("%d", &k);

printf("\n");

}

bus[i].n = bus[i].n + 1;

for (j = bus[i].n; j > k - 1; j--)

{

bus[i].name[j] = bus[i].name[j - 1];

}

printf("\n请输入新添加的站点名称：");

scanf("%s", bus[i].name[k - 1].c);

save();

break;

}

case '2': //修改站点

{

printf("\n请输入需要修改第几个站点：");

scanf("%d", &k);

printf("\n请输入新的站点名称：");

scanf("%s", bus[i].name[k - 1].c);

save();

break;

}

case '3': //删除站点

{

printf("\n请输入需要删除第几个站点：");

scanf("%d", &k);

for (j = k - 1; j < bus[i].n; j++)

{

bus[i].name[j] = bus[i].name[j + 1];

}

bus[i].n = bus[i].n - 1;

save();

break;

}

case '4': //重置路线

{

printf("\n请输入新的途径站台总数(2<=n<=25): ");

scanf("%d", &bus[i].n);

printf("\n");

while (bus[i].n < 2 || bus[i].n>12) //判断是否满足条件

{

printf("\n站台总数应满足(2<=n<=25)，请重新输入: ");

scanf("%d", &bus[i].n);

printf("\n");

}

for (j = 0; j < bus[i].n; j++)

{

printf("请输入新的第%d个站台名称: ", j + 1);

scanf("%s", bus[i].name[j].c);

}

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

save();

break;

}

case '5': //返回菜单

{

printf("\n\t\t按回车键返回菜单\n");

modbus\_menu();

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return; //返回主菜单

}

}

}

}

case '4': //修改行车时间

{

printf("\n\n请输入公交车的起始时间(格式为：时:分): ");

scanf("%s", bus[i].topen);

printf("请输入公交车的终止时间(格式为：时:分): ");

scanf("%s", bus[i].tclose);

save();

break;

}

case '5':

{

printf("\n\t\t按回车键返回主菜单\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return; //返回主菜单

}

}

}

}

}

if (m == 0)

printf("\n\n\t\t无此公交车信息!!!\n");

printf("\n\t\t按回车键返回\n");

fflush(stdin); //清除键盘缓冲区

getchar();

system("cls");

return;

}

其修改时间、票价、公交编号都位于修改的第一层菜单，而较为复杂的修改路线则位于第二层菜单。修改前前某一条线路的信息如下：

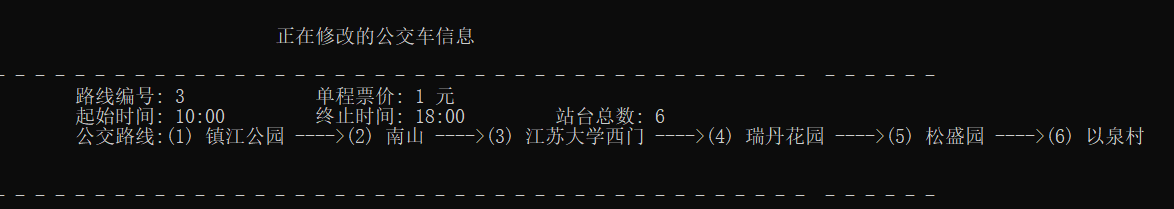


图 3.2.7

把其时间修改成7：00-20：00，票价修改为3元，公交线路名称修改为8路，修改后显示出的公交信息为：

可以看到已经修改完成。

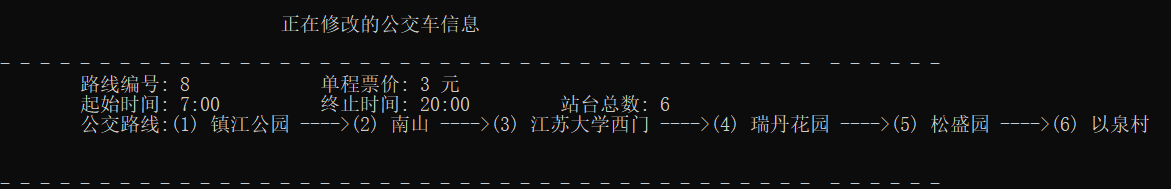


图 3.2.8

那么在路线修改时，管理员可以增加站点，也可以减少站点，在原有站点改名时，也可以修改原有站点的名字，若是由于线路施工导致原有站点大幅度调整，也可以选择充值站点。程序演示：

1. 在三号线路镇江南站和南山站之间新增一个站点：school（英文和中文区别度大，可以更加显著的看到区别）。
2. 删去原有瑞丹花园这个站。
3. 将以泉村改名为church。

以上操作的运行截图如下所示：

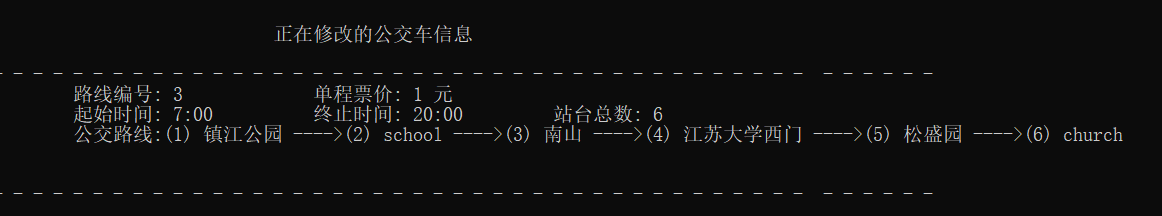


图 3.2.9

可以看到修改完成。

若将第三条路线重置，其运行截图为：

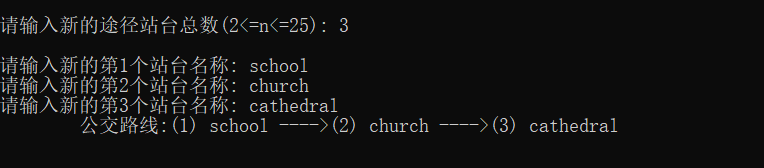


图 3.2.10

再次查询线路三的信息可得：

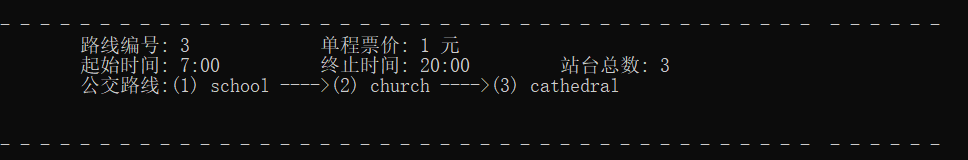


图 3.2.11

## 菜单的设置以及各个菜单之间的链接

菜单本身的设置没有什么太大的难度，只要用一些printf函数将其输出即可，下面时几个菜单的设置（以运行界面给出）：

主菜单分为管理员系统和乘客系统，管理员系统可以进行路线的修改和查询功能，而乘客系统仅仅只有查询系统，其中管理员系统需要输入密码。而查询界面分为查询所有线路路线，也可以查询某一条线路路线，可以进行换乘，最短路径（依据站点）的查询，也可以进行旅游路线的查询，界面如下面四图所示（路线修改菜单已经在前文给出）：

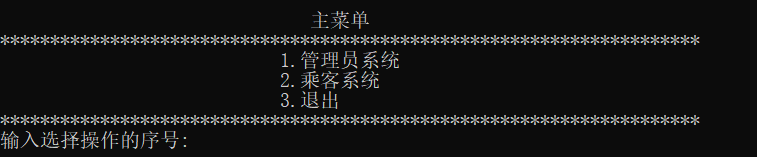


图 3.3.1

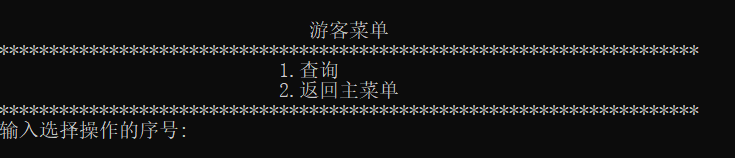


图 3.3.2

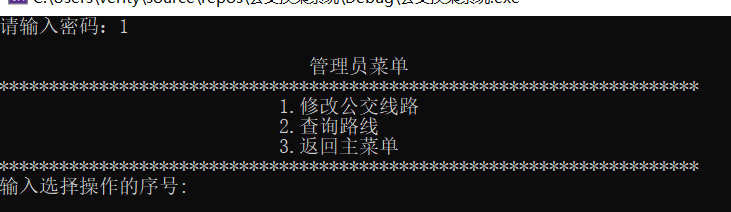


图 3.3.3

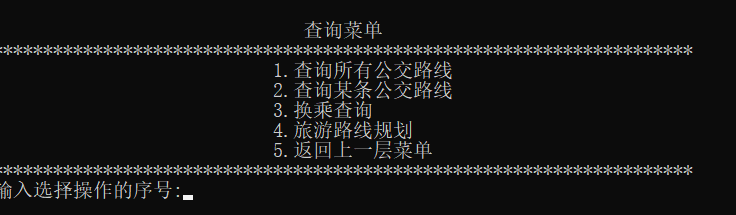


图 3.3.4

而其进行连接主要使用switch函数即可，其中修改菜单的连接代码如下所示，其他连接类似于如下连接：

void modbus() {

char a;

while (1)

{

//system("cls");

modbus\_menu();

a = Test(getchar(), '1', '4'); //菜单检测输入函数

switch (a)

{

case '1': system("cls"); mod(); break;

case '2': system("cls"); add(); break;

case '3': system("cls"); delete(); break;

case '4': system("cls"); administrator(); break;

}

}

return 0;

}

## 查询系统

### 3.4.1所有公交信息的显示

使用数组for循环的方式把所有公交信息输出，直至数组内为NULL。具体代码如下所示：

void view()

{

int i, j, min;

show();

printf("\n\n 公交车信息库");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

}

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n\t\t公交车信息显示完毕!!!\n");

printf("\n\t\t输入回车键返回主菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

其运行截图为：

### 3.4.2特定条公交信息的显示

图 3.4.1

输入公交编号，在结构体对应的存储公交编号的数组中寻找该条路线的信息，若没找到则输出没找到。此处寻找使用的是strcmp这个函数。具体代码如下所示：

void findnum()

{

int i, j, m = 0;

char cc[20];

printf("请输入您想要查找的公交路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n已找到:\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

break;

}

}

if (m == 0)

printf("\n\t\t无此公交车信息\n");

printf("\n\t\t输入回车键返回查询菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

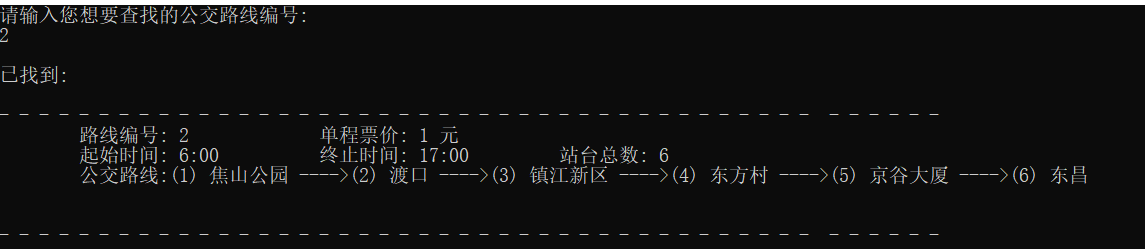
运行截图如下所示：

图 3.4.2

### 3.4.3 换乘查询以及旅游路线规划

先寻找直达，一次换乘，多次换乘等情况，由于站点路线较少以及我的技术问题，最多的情况也仅仅止步于一次换乘，没有多次换乘的情况出现，我在此加上了距离，并且有图的初始化来完成，可以当作图的权值，此处初始化图的代码不给出。主要使用Floyd算法实现：

void Floyd(MGraph G)

{

int v, w, k;

for (v = 0; v < G.numVertexes; ++v) /\* 初始化D与P \*/

{

for (w = 0; w < G.numVertexes; ++w)

{

D[v][w] = G.arc[v][w]; /\* D[v][w]值即为对应点间的权值 \*/

P[v][w] = w; /\* 初始化P \*/

}

}

for (k = 0; k < G.numVertexes; ++k)

{

for (v = 0; v < G.numVertexes; ++v)

{

for (w = 0; w < G.numVertexes; ++w)

{

if (D[v][w] > D[v][k] + D[k][w])/\* 如果经过下标为k顶点路径比原两点间路径更短 \*/

{

D[v][w] = D[v][k] + D[k][w];/\* 将当前两点间权值设为更小的一个 \*/

P[v][w] = P[v][k];/\* 路径设置为经过下标为k的顶点 \*/

}

}

}

}

}

再将两点之间的公交路线输出，代码如下图所示：

void Transfer(MGraph\* G, int L[MAXVEX])

{

int i = 0, j = 0, k = 0;

for (i = 0; i < 25; i++)

{

for (j = 1; j < 6; j++)

{

if (G->vexs[L[i]].lines[j] == G->vexs[L[i + 1]].lines[j])

{

if (G->vexs[L[i]].lines[j] != 0)

{

printf(" 站点%d %d路)", L[k], G->vexs[L[i]].lines[j]);

k++;

break;

}

}

}

}

printf(" 站点 %d", L[k]);

}

void Dispath(MGraph\* G)

{

int v, w, k;

int i = 0;

int L[MAXVEX] = { 0 }; //存放两站之间的最短路径

printf(" 输入出发站：站点");

scanf("%d", &v);

printf(" 输入终点站：站点");

scanf("%d", &w);

printf("站点%d --> 站点 %d 距离：%d 千米", v, w, D[v][w]);

k = P[v][w]; /\* 获得第一个路径顶点下标 \*/

printf(" 路线: %d", v); /\* 打印源点 \*/

L[i] = v;

while (k != w) /\* 如果路径顶点下标不是终点 \*/

{

printf(" -> %d", k); /\* 打印路径顶点 \*/

i++;

L[i] = k;

k = P[k][w]; /\* 获得下一个路径顶点下标 \*/

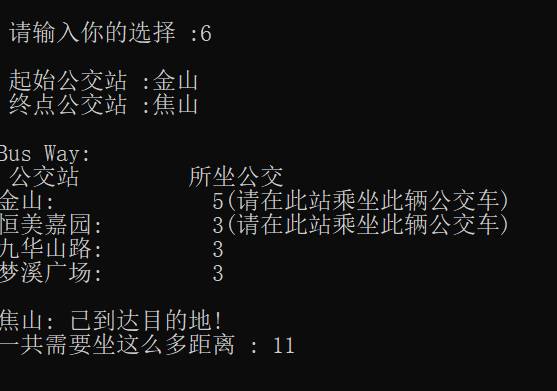
 }

图 3.4.3

L[i + 1] = w;

printf(" -> %d 共 %d 站", w, i + 1);

Transfer(G, L);

}

运行截图如右图所示：

当然 可以显示出完整的两路线路，代码如下所示：

void searchnavi() //车辆换乘查询函数

{

int i, j, k, m = 0;

int t1, t2, n = 0;

char c1[20], c2[20];

printf("请输入您目前所在的站点:\n\t\t");

scanf("%s", c1);

printf("请输入您想要前往的站点:\n\t\t");

scanf("%s", c2);

printf("\n\n您可选择的公交车信息如下\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

for (k = j + 1; k < bus[i].n; k++)

if (strcmp(bus[i].name[j].c, c1) == 0 && strcmp(bus[i].name[k].c, c2) == 0)

{

m = 1;

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s",bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

}

}

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

printf("\t\t\t\t显示完毕\n");

if (m == 0)

{

printf("\n\n\t\t\t\t抱歉,无途经此路线的公交车\n");

for (i = 0; i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

if (strcmp(bus[i].name[j].c, c1) == 0)

t1 = i;

}

for (i = 0; i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

if (strcmp(bus[i].name[j].c, c2) == 0)

t2 = i;

}

for (i = 0; i <bus[t1].n; i++)

{

for (j = 0; j < bus[t2].n; j++)

{

if (strcmp(bus[t1].name[i].c,bus[t2].name[j].c) == 0)

{

n = 1;

}

}

}

if (n == 1)

{

printf("你可以选择换乘，具体路线信息如下：\n");

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d \n", bus[t1].num, bus[t1].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[t1].topen, bus[t1].tclose, bus[t1].n);

printf("\t公交路线:(1) %s", bus[t1].name[0].c);

for (j = 1; j < bus[t1].n; j++)

printf(" -->(%d) %s", j + 1, bus[t1].name[j].c);

printf("\n\n");

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[t2].num, bus[t2].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[t2].topen, bus[t2].tclose, bus[t2].n);

printf("\t公交路线:(1) %s", bus[t2].name[0].c);

for (j = 1; j < bus[t2].n; j++)

printf(" -->(%d) %s", j + 1, bus[t2].name[j].c);

printf("\n\n");

}

else

printf("不可达！\n");

}

printf("\n\t\t\t\t输入回车键返回查询菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

return;

}

运行截图如图所示：

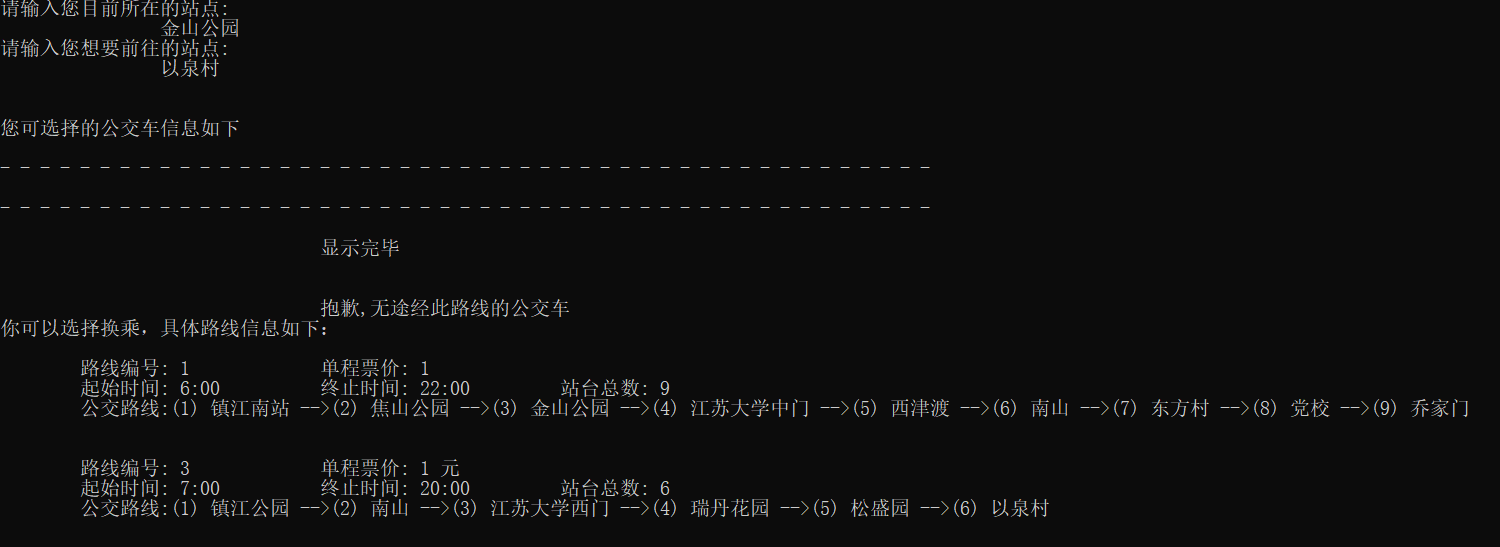


图3.4.4

对于旅游路线，两点之间的换乘与换乘的算法相似，此处不进行单独说明。

# 小结

本次课程设计难度我觉得较大，主要难度在于换乘，以及最短路径的规划。由于我是按照公交线路来进行画图，在公交线路未连接的两个站点假定距离无穷大。虽然用Floyd算法很容易就体现出最短路径，还有其中的路程，但是用我遇到了无法把站点与公交车编号连接起来的难题。最后我选择用另外一个结构体存储每个公交车站会有哪路公交车经过，同时也要在边输入新的站点时，边更新图的情况，这个问题比较困难。也卡了我许久。然后直达，一次换乘和多次换乘我也没有考虑到在换乘途中的等待问题，也没有设置公交车每一趟的出发时间和回来时间（在假设速度恒定的情况下），我觉得这是一个比较大的漏洞，光看距离并不是那么完善。这个问题仍待解决。

在做管理员系统时，公交信息的修改也比较繁杂，但难度我认为比换乘小。借此课程设计我认识到我还有很多不足，课内学的知识不足以支撑我完成一份完善的代码，在课外仍需学习更多的知识和技术内容。

# 参考文献

[1]谭浩强，C程序设计，北京清华大学学研大厦A座，清华大学出版社，2005年7月第三版

[2]秦玉平，马靖善,数据结构（C语言版）(第三版)，北京清华大学学研大厦A座，清华大学出版社，2005年10月第一版

# 6.附录（源代码）

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<stdlib.h>

#define INFINITY 999999999999 /\*无穷大\*/

#define MAX 40

int STATION\_NUM; //站点总数

struct station //途径站点信息

{

char c[20]; //站点名称

};

typedef struct bus //公交车信息

{

char num[20]; //公交车编号

int n; //站台数目

struct station name[12]; //站台名称

char topen[20]; //起始时间

char tclose[20]; //终止时间

int money; //票价

}BUS;

BUS bus[MAX];

typedef struct

{

int name; //站点名称

int lines[6]; //途径路线

}Vertex;

typedef struct

{

Vertex vexs[MAX]; //站点

int arc[MAX][MAX]; //邻接矩阵

int numVertexes, numEdges; //顶点数和边数

}MGraph;

int Distance[100][100];

//声明：

void init(); void save();int host\_menu(); int administrator\_menu();int visitor\_menu();

int query\_menu();void modbus\_menu(); void modstation\_menu(); char Test(char a, char b, char c);void add();void view(); void mod();

void delete(); void findnum();void administrator(); void menu();void visitor();void floyd();void searchnavi();int show();void show\_travel();

/\* 构造图 \*/

void CreateMGraph(MGraph\* G)

{

int i, j;

//请输入边数和顶点数

G->numEdges = 52;

G->numVertexes = 26;

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

{

G->vexs[i].name = i;

}

for (i = 0; i < 26; i++)//初始化每个站点通过路线的数组，设为零

{

for (j = 0; j < 6; j++)

{

G->vexs[i].lines[j] = 0;

}

}

G->vexs[1].lines[1] = 1;

G->vexs[2].lines[1] = 1; G->vexs[2].lines[2] = 2;

G->vexs[3].lines[1] = 1;

G->vexs[4].lines[1] = 1;

G->vexs[5].lines[2] = 2;

G->vexs[6].lines[1] = 1;

G->vexs[7].lines[2] = 2;

G->vexs[8].lines[1] = 1; G->vexs[8].lines[2] = 2;

G->vexs[9].lines[1] = 1; G->vexs[9].lines[2] = 2; G->vexs[9].lines[3] = 3;

G->vexs[10].lines[3] = 3;

G->vexs[11].lines[1] = 1;

G->vexs[12].lines[3] = 3;

G->vexs[13].lines[2] = 2; G->vexs[13].lines[5] = 5;

G->vexs[14].lines[1] = 1;

G->vexs[15].lines[3] = 3;

G->vexs[16].lines[5] = 5;

G->vexs[17].lines[2] = 2;

G->vexs[18].lines[4] = 4;

G->vexs[19].lines[3] = 3; G->vexs[19].lines[4] = 4;

G->vexs[20].lines[5] = 5;

G->vexs[21].lines[5] = 5;

G->vexs[22].lines[3] = 3;

G->vexs[23].lines[4] = 4;

G->vexs[24].lines[4] = 4;

G->vexs[25].lines[4] = 4; G->vexs[25].lines[5] = 5;

int t[MAX][MAX] = { 0 }; //构造无向带权图的邻接矩阵

t[1][2] = t[2][1] = 2;

t[2][3] = t[3][2] = 3;

t[2][5] = t[5][2] = 7;

t[3][4] = t[4][3] = 5;

t[4][6] = t[6][4] = 9;

t[5][7] = t[7][5] = 7;

t[6][9] = t[9][6] = 2;

t[7][8] = t[8][7] = 4;

t[8][9] = t[9][8] = 6;

t[8][11] = t[11][8] = 2;

t[9][10] = t[10][9] = 5;

t[9][12] = t[12][9] = 7;

t[11][14] = t[14][11] = 8;

t[12][15] = t[15][12] = 8;

t[13][9] = t[9][13] = 2;

t[13][16] = t[16][13] = 3;

t[13][17] = t[17][13] = 3;

t[15][19] = t[19][15] = 6;

t[16][20] = t[20][16] = 7;

t[18][19] = t[19][18] = 1;

t[19][22] = t[22][19] = 4;

t[19][23] = t[23][19] = 5;

t[20][21] = t[21][20] = 1;

t[21][25] = t[25][21] = 3;

t[23][24] = t[24][23] = 1;

t[24][25] = t[25][24] = 7;//?

for (i = 0; i < G->numVertexes; i++)//\* 初始化图

{

for (j = 0; j < G->numVertexes; j++)

{

if (i == j)

G->arc[i][j] = 0;

else

G->arc[i][j] = G->arc[j][i] = INFINITY;

}

}

for (i = 0; i < G->numVertexes; i++)//\* 复制邻接矩阵

{

for (j = 0; j < G->numVertexes; j++)

{

if (t[i][j] != 0)

{

G->arc[i][j] = t[i][j];

}

}

}

}

//初始化

void init() //初始化函数

{

FILE\* fp; //文件指针

int i;

if ((fp = fopen("bus.txt", "r+")) == NULL) //初次尝试打开"bus.txt"文本文件

{

printf("\n\t\t文件打开失败\n\n\t\t正在尝试创建新文件...\n");

fp = fopen("bus.txt", "w"); //创建"bus.txt"文本文件

if ((fp = fopen("bus.txt", "r+")) == NULL) //再次判断是否打开成功

{

printf("\t\t文件创建失败!!!\n");

return;

}

}

fp = fopen("bus.txt", "r+");

for (i = 0; i < MAX; i++) //将磁盘中的信息输出到内存中

if (fread(&bus[i], sizeof(struct bus), 1, fp) != 1)

break;

fclose(fp);

printf("\n\t\t初始化完成!!!\n\n");

return;

}

//将删改后的信息写入文档

void save() //信息保存函数

{

FILE\* fp = NULL;

int i;

if ((fp = fopen("bus.txt", "w+")) == NULL) //以读写的方式打开名称为"bus"的文本文件

{

printf("文件打开错误!!!\n");

return;

}

fp = fopen("bus.txt", "w+");

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++) //将内存中的数据输出到磁盘中

if (fwrite(&bus[i], sizeof(struct bus), 1, fp) != 1)

break;

printf("\n\t保存成功!!!\n");

fclose(fp);

return;

}

//主菜单（1级菜单）

int host\_menu()

{

printf("\n");

printf(" 主菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 1.管理员系统 \n");

printf(" 2.乘客系统 \n");

printf(" 3.退出 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入选择操作的序号:");

}

//管理员菜单

int administrator\_menu()

{

printf("\n");

printf(" 管理员菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 1.修改公交线路 \n");

printf(" 2.查询路线 \n");

printf(" 3.返回主菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入选择操作的序号:");

}

//游客菜单

int visitor\_menu()

{

printf("\n");

printf(" 游客菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 1.查询 \n");

printf(" 2.返回主菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入选择操作的序号:");

}

//路线查询菜单

int query\_menu()

{

printf("\n");

printf(" 查询菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 1.查询所有公交路线 \n");

printf(" 2.查询某条公交路线 \n");

printf(" 3.换乘查询 \n");

printf(" 4.旅游路线规划 \n");

printf(" 5.返回上一层菜单 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入选择操作的序号:");

}

//查询

void query(int n) {

char a;

while (1)

{

//system("cls");

query\_menu();

a = Test(getchar(), '1', '5'); //菜单检测输入函数

switch (a)

{

case '1': system("cls"); view(); break;

case '2': system("cls"); findnum(); break;

case '3': system("cls"); show(); searchnavi(); break;

case '4': system("cls"); show\_travel(); searchnavi(); break;

case '5': system("cls");

if (n == 1)

administrator();

else

visitor();

break;

}

}

return 0;

}

//修改公交信息菜单

void modbus\_menu() //修改菜单

{

printf("\n\t\t 修改菜单 ");

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

printf("\n\t\t\* 1、修改原有路线 \*");

printf("\n\t\t\* 2、新增路线 \*");

printf("\n\t\t\* 3、删除原有路线 \*");

printf("\n\t\t\* 4、返回到管理员菜单 \*");

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

printf("\n\n\t\t请输入您的选择:");

}

//修改具体路线菜单

void modload\_menu() {

printf("\n\t\t 修改具体路线菜单 ");

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

printf("\n\t\t\* 1、修改路线编号 \*");

printf("\n\t\t\* 2、修改车辆票价 \*");

printf("\n\t\t\* 3、修改行车路线 \*");

printf("\n\t\t\* 4、修改行车时间 \*");

printf("\n\t\t\* 5、返回上一层菜单 \*");

printf("\n\n\t\t请输入您的选择:");

}

//修改具体路线菜单

void modstation\_menu() //路线修改菜单

{

printf("\n\t\t 路线修改菜单 ");

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

printf("\n\t\t\* 1、 站 点 添 加 \*");

printf("\n\t\t\* 2、 站 点 修 改 \*");

printf("\n\t\t\* 3、 站 点 删 除 \*");

printf("\n\t\t\* 4、 路 线 重 置 \*");

printf("\n\t\t\* 5、 返 回 菜 单 \*");

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

printf("\n\n\t\t请输入您的选择:");

}

//检测输入是否正确的函数

char Test(char a, char b, char c)

{

while (a<b || a>c)

{

printf("\n\t\t\t输入错误，请重新输入: ");

fflush(stdin); //清除键盘缓冲区

a = getchar();

}

return a;

}

//添加新的路线

void add()

{

FILE\* fp = NULL; //文件指针

int i, j;

char cc[20];

for (i = 0; i < MAX; i++)

{

if (\*bus[i].num != '\0')

continue;

else

{

printf("\n添加第%d辆公交车路线记录:\n", i + 1);

printf("\n请输入路线编码:\n"); //路线编码

scanf("%s", cc);

for (j = 0; j < MAX; j++) //检验是否重复

if (strcmp(bus[j].num, cc) == 0)

{

printf("\n与已有路线编码重复，按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

strcpy(bus[i].num, cc);

printf("\n请输入途径站台总数(>=2): "); //站台总数

scanf("%d", &bus[i].n);

printf("\n");

if (bus[i].n < 2 || bus[i].n>12)

{

while (bus[i].n < 2 || bus[i].n>12)

{

printf("\n站台总数应满足(2<=n<=25)，请重新输入: ");

scanf("%d", &bus[i].n);

printf("\n");

}

}

for (j = 0; j < bus[i].n; j++)

{

printf("请输入第%d个站台名称 ", j + 1); //站台名称

scanf("%s", bus[i].name[j].c);

}

printf("\n自动生成公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

{

printf(" ----> (%d) %s", j + 1, bus[i].name[j].c);

}

printf("\n\n请输入公交车的起始时间: "); //起始时间

scanf("%s", bus[i].topen);

printf("请输入公交车的终止时间: "); //终止时间

scanf("%s", bus[i].tclose);

printf("\n请输入公交车的票价: "); //票价

scanf("%d", &bus[i].money);

printf("\n第%d辆公交车路线记录创建成功!!!\n", i + 1);

save();

printf("\n\t按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

}

if (i == MAX)

printf("\n\n\n\t空间已满，不能录入!!!\n");

printf("\n\t按回车键返回!!!\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

//显示所有公交信息

void view()

{

int i, j, min;

show();

printf("\n\n 公交车信息库");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

}

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n\t\t公交车信息显示完毕!!!\n");

printf("\n\t\t输入回车键返回主菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

//修改公交信息函数

void mod()

{

int i, j, m = 0;

char t, cc[20], mod[20];

printf("请输入要修改信息的公交车路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++) //查找所输入的公交车

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n\n\t\t已找到!!!\n");

while (1)

{

system("cls");

printf("\n\n 正在修改的公交车信息\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

modload\_menu();

fflush(stdin); //清除键盘缓冲区

t = Test(getchar(), '1', '5'); //菜单检测输入函数

//system("cls");

switch (t)

{

case '1': //修改路线编号

{

int k;

printf("\n请输入新的公交车编号:");

scanf("%s", mod);

for (k = 0; \*bus[k].num != '\0' && k < MAX; k++)

{

if (strcmp(bus[k].num, mod) == 0)

{

printf("与已有编号重复,按回车键返回主菜单\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

}

strcpy(bus[i].num, mod);

save();

break;

}

case '2': //修改车辆信息

{

printf("\n请输入新的公交车票价:");

scanf("%d", &bus[i].money);

save();

break;

}

case '3': //修改行车路线

{

void modstation\_menu(); //路线修改菜单

int k;

char z;

while (1)

{

//system("cls");

modstation\_menu();

fflush(stdin); //清除键盘缓冲区

z = Test(getchar(), '1', '6'); //菜单检测输入函数

//system("cls");

switch (z)

{

case '1': //添加站点

{

if (bus[i].n + 1 > 12) //判断是否满足条件

{

printf("\n站台总数达到25个，无法添加新站点\n\n按回车键返回\n");

fflush(stdin); //清除键盘缓冲区

getchar();

system("cls");

return;

}

printf("\n请输入需要添加第几个站点：");

scanf("%d", &k);

while (bus[i].n + 1 < k)

{

printf("目前共%d个站点，无法添加第%d个站点\n请重新输入：", bus[i].n, k);

scanf("%d", &k);

printf("\n");

}

bus[i].n = bus[i].n + 1;

for (j = bus[i].n; j > k - 1; j--)

{

bus[i].name[j] = bus[i].name[j - 1];

}

printf("\n请输入新添加的站点名称：");

scanf("%s", bus[i].name[k - 1].c);

save();

break;

}

case '2': //修改站点

{

printf("\n请输入需要修改第几个站点：");

scanf("%d", &k);

printf("\n请输入新的站点名称：");

scanf("%s", bus[i].name[k - 1].c);

save();

break;

}

case '3': //删除站点

{

printf("\n请输入需要删除第几个站点：");

scanf("%d", &k);

for (j = k - 1; j < bus[i].n; j++)

{

bus[i].name[j] = bus[i].name[j + 1];

}

bus[i].n = bus[i].n - 1;

save();

break;

}

case '4': //重置路线

{

printf("\n请输入新的途径站台总数(2<=n<=25): ");

scanf("%d", &bus[i].n);

printf("\n");

while (bus[i].n < 2 || bus[i].n>12) //判断是否满足条件

{

printf("\n站台总数应满足(2<=n<=25)，请重新输入: ");

scanf("%d", &bus[i].n);

printf("\n");

}

for (j = 0; j < bus[i].n; j++)

{

printf("请输入新的第%d个站台名称: ", j + 1);

scanf("%s", bus[i].name[j].c);

}

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

save();

break;

}

case '5': //返回菜单

{

printf("\n\t\t按回车键返回菜单\n");

modbus\_menu();

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return; //返回主菜单

}

}

}

}

case '4': //修改行车时间

{

printf("\n\n请输入公交车的起始时间(格式为：时:分): ");

scanf("%s", bus[i].topen);

printf("请输入公交车的终止时间(格式为：时:分): ");

scanf("%s", bus[i].tclose);

save();

break;

}

case '5':

{

printf("\n\t\t按回车键返回主菜单\n");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return; //返回主菜单

}

}

}

}

}

if (m == 0)

printf("\n\n\t\t无此公交车信息!!!\n");

printf("\n\t\t按回车键返回\n");

fflush(stdin); //清除键盘缓冲区

getchar();

system("cls");

return;

}

//删除公交信息函数

void delete() //信息删除函数

{

int i, j, t, m = 0;

char cc[20];

printf("\n请输入要删改信息的公交车路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++) //查找所输入的公交车

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n已找到:\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

printf("即将删除该公交车信息!!!\n确定请按1：");

scanf("%d", &t);

if (t == 1)

{

if (i + 1 == MAX)

memset(&bus[i], '\0', sizeof(struct bus)); //最后一辆直接进行清空

else

for (j = i; \*bus[j].num != '\0' && j + 1 < MAX; j++) //后面的公交车信息补上删去的公交车信息

bus[j] = bus[j + 1];

printf("编号为%s的公交车信息删除完毕!\n", cc);

save();

break;

}

}

}

if (m == 0)

printf("\n\n\t\t无此公交车信息!!!\n");

printf("\n\n\t输入回车键返回主菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

//根据公交车号查询

void findnum()

{

int i, j, m = 0;

char cc[20];

printf("请输入您想要查找的公交路线编号:\n");

scanf("%s", cc);

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

if (strcmp(bus[i].num, cc) == 0)

{

m = 1;

printf("\n已找到:\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s", bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

break;

}

}

if (m == 0)

printf("\n\t\t无此公交车信息\n");

printf("\n\t\t输入回车键返回查询菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

//system("cls");

return;

}

//路线导航查询函数(?)功能不全 暂定

void modbus() {

char a;

while (1)

{

//system("cls");

modbus\_menu();

a = Test(getchar(), '1', '4'); //菜单检测输入函数

switch (a)

{

case '1': system("cls"); mod(); break;

case '2': system("cls"); add(); break;

case '3': system("cls"); delete(); break;

case '4': system("cls"); administrator(); break;

}

}

return 0;

}

//游客功能

void visitor() {

char a;

while (1)

{

system("cls");

visitor\_menu();

a = Test(getchar(), '1', '2'); //菜单检测输入函数

switch (a)

{

case '1': system("cls"); query(2); break;

case '2': system("cls"); menu(); break;

}

}

return 0;

}

//主菜单

void menu() {

char a;

int n=0;

while (1)

{

system("cls");

host\_menu();

a = Test(getchar(), '1', '3'); //菜单检测输入函数

switch (a)

{

case '1': system("cls");

while (n != 1) {

printf("请输入密码：");

scanf("%d", &n);

if (n == 1) {

administrator();

}

else {

printf("密码错误！请重新输入");

scanf("%d", &n);

}

}

break;

case '2': system("cls"); visitor(); break;

case '3': exit(1); return 0;

}

}

return 0;

}

//管理员功能

void administrator(){

char a;

while (1)

{

//system("cls");

administrator\_menu();

a = Test(getchar(), '1', '3'); //菜单检测输入函数

switch (a)

{

case '1': system("cls"); modbus(); break;

case '2': system("cls"); query(1); break;

case '3': system("cls"); menu(); break;

}

}

return 0;

}

void searchnavi() //车辆换乘查询函数

{

int i, j, k, m = 0;

int t1, t2, n = 0;

char c1[20], c2[20];

printf("请输入您目前所在的站点:\n\t\t");

scanf("%s", c1);

printf("请输入您想要前往的站点:\n\t\t");

scanf("%s", c2);

printf("\n\n您可选择的公交车信息如下\n");

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

for (i = 0; \*bus[i].num != '\0' && i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

for (k = j + 1; k < bus[i].n; k++)

if (strcmp(bus[i].name[j].c, c1) == 0 && strcmp(bus[i].name[k].c, c2) == 0)

{

m = 1;

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[i].num, bus[i].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[i].topen, bus[i].tclose, bus[i].n);

printf("\t公交路线:(1) %s",bus[i].name[0].c);

for (j = 1; j < bus[i].n; j++)

printf(" ---->(%d) %s", j + 1, bus[i].name[j].c);

printf("\n\n");

}

}

printf("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n\n");

printf("\t\t\t\t显示完毕\n");

if (m == 0)

{

printf("\n\n\t\t\t\t抱歉,无途经此路线的公交车\n");

for (i = 0; i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

if (strcmp(bus[i].name[j].c, c1) == 0)

t1 = i;

}

for (i = 0; i < MAX; i++)

{

for (j = 0; j < bus[i].n; j++)

if (strcmp(bus[i].name[j].c, c2) == 0)

t2 = i;

}

for (i = 0; i <bus[t1].n; i++)

{

for (j = 0; j < bus[t2].n; j++)

{

if (strcmp(bus[t1].name[i].c,bus[t2].name[j].c) == 0)

{

n = 1;

}

}

}

if (n == 1)

{

printf("你可以选择换乘，具体路线信息如下：\n");

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d \n", bus[t1].num, bus[t1].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[t1].topen, bus[t1].tclose, bus[t1].n);

printf("\t公交路线:(1) %s", bus[t1].name[0].c);

for (j = 1; j < bus[t1].n; j++)

printf(" -->(%d) %s", j + 1, bus[t1].name[j].c);

printf("\n\n");

printf("\n");

printf("\t路线编号: %-6s\t单程票价: %d 元\n", bus[t2].num, bus[t2].money);

printf("\t起始时间: %-6s\t终止时间: %-6s\t站台总数: %d\n", bus[t2].topen, bus[t2].tclose, bus[t2].n);

printf("\t公交路线:(1) %s", bus[t2].name[0].c);

for (j = 1; j < bus[t2].n; j++)

printf(" -->(%d) %s", j + 1, bus[t2].name[j].c);

printf("\n\n");

}

else

printf("不可达！\n");

}

printf("\n\t\t\t\t输入回车键返回查询菜单:");

fflush(stdin); //清除键盘缓冲区

getchar();

return;

}

int dist[MAX][MAX], path[MAX][MAX];//dist 数组用来记录最短路径的权值,path 数组用 来记录路径

int show() {

printf(" 各个公交车站编号 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t1.镇江南站\t2.焦山公园\t3.金山公园\t4.江苏大学中门\t5.渡口\n");

printf("\t6.西津渡\t7.镇江新区\t8.东方村\t9.南山\t10.镇江中学\n");

printf("\t11.党校\t12.江苏大学西门\t13.京谷大厦\t14.乔家门\t15.瑞丹花园\n");

printf("\t16.彩虹城\t17.东昌\t18.印江桥\t19.松盛园\t20.北固山\n");

printf("\t21.汽车客运站\t22.以泉村\t23.宝塔山公园\t24.我家山水\t25.丁卯开发区\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void show\_travel() {

printf(" 旅游攻略查询 \n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t1.北固山(编号20）\t2.焦山公园（编号2）\t3.金山公园（编号3）\t4.南山（编号9）\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void floyd(MGraph \*G)

{

int i, j, v;

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

for (j = 0; j < G->numVertexes; j++)

{

if (i == j) dist[i][j] = 0;

dist[i][j] = G->arc[i][j];

path[i][j] = -1;

}

for (v = 0; v < G->numVertexes; v++)

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

for (j = 0; j < G->numVertexes; j++)

{

if (dist[i][v] + dist[v][j] < dist[i][j])

{

dist[i][j] = dist[i][v] + dist[v][j];

path[i][j] = v;

}

}

}

int main() {

char a;

int n;

scanf("%d", &n);

STATION\_NUM = n;

init();

menu();

return 0;

}