中 北 大 学

数据结构课程设计说明书

学　院、系：软件学院

专　　业：软件工程

班　　级：20130401 班

学 生 姓 名：Xxx　学 号：20130xxxxx

设 计 题 目：校园导航问题

起 迄 日 期：2021 年 6 月 22 日~2021 年 7 月 2 日

指 导 教 师：李玉蓉

日期: 2021 年 7 月 2 日

1 设计目的

（1）了解并掌握数据结构与算法的设计方法，具备初步的独立分析和设计能力；

（2）初步掌握软件开发过程的问题分析、系统设计、程序编码、测试等基本方法和技能；

（3）提高综合运用所学的理论知识和方法独立分析和解决问题的能力；

（4）训练用系统的观点和软件开发一般规范进行软件开发，培养软件工作者所应具备的科学的工作方法作风。

2 任务概述

（1）设计学校的平面图（已包括 15 个不同的场所）。每两个场所间有不同的路径，且路长也不同（图形结构可以通过键盘输入数据后采用创建图的算法生成图形）；

（2）可以展示已输入的地点信息

（3）可以展示已输入的道路信息

（4）可以修改，删除已输入的地点信息

（5）可以修改，删除已输入的道路信息

（6）提供起始点能自动找出从此地点到达其它地点的最短路径

（7）提供起始点与终点能自动找出从任意场所到达另一场所的最短路径

3 本设计采用的数据结构

```
typedef struct {
    char name[JD];
    char xinxi[1000];
} VertexType;//地图信息的定义
typedef struct {
    VertexType vexs[JD];
    int edges[JD][JD];
    int v,e;
}MGraph;//地图（无向图）的边和节点的定义
```

4 系统功能模块结构图及主要函数说明

4.1 系统功能模块结构图

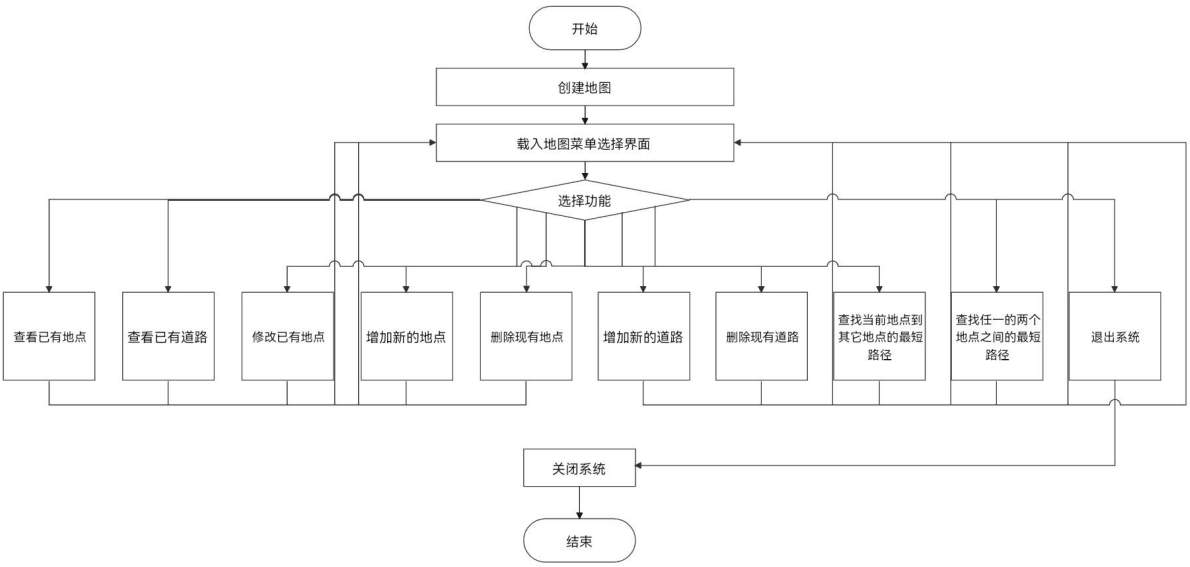


图 4.1 系统功能模块结构图

4.2 节点及信息数据结构类型

```
typedef struct {
    char name[JD];//定义地点名称
    char xinxi[1000];//定义地点信息
} VertexType;//地图信息的定义
typedef struct {
    VertexType vexs[JD];
    int edges[JD][JD];
    int v,e;
}MGraph;//地图（无向图）的边和节点的定义
```

4.3 创建地图

```
M.v=15;//已录入的地点数量
M.e=23;//已录入地点的道路的数量
int i,j,k;
for(i=0; i<JD; i++){
    for(j=0; j<JD; j++){
        M.edges[i][j]=INF;
    }
}//先创建默认边为无穷(长度为:INF)的地图
strcpy(M.vexs[0].name,"XXX");//写入地点名字
```

strcpy(M.vexs[0].xinxi,"XXXXXX");//写入地点介绍

edges[0][1]=M.edges[1][0]=230;//将 0 号节点到 1 号节点的路径赋值为 230，因为是无向图，所以 1 号顶点到 0 号顶点的路径长度也应赋值为 230

4.4 主函数

先初始化地图后，进入菜单，由于为了每次操作执行完后可以返回菜单，所以用了 while 永真循环

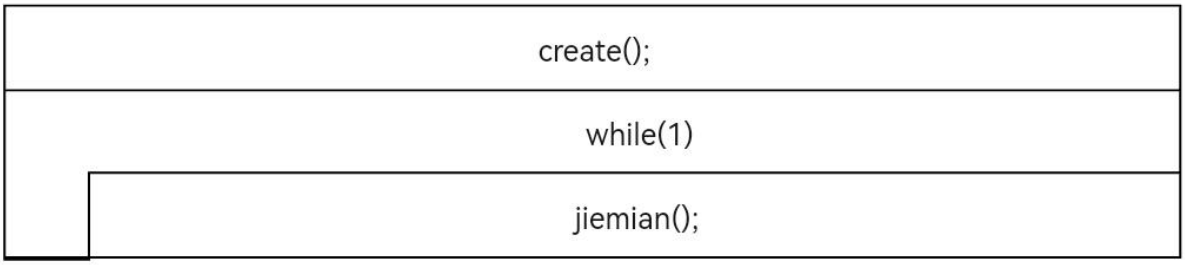


图 4.2 主函数 N-S 图

4.5 菜单界面

调用 setColor 函数来实现终端显示颜色的更改，用 switch 语句用来选择相应功能

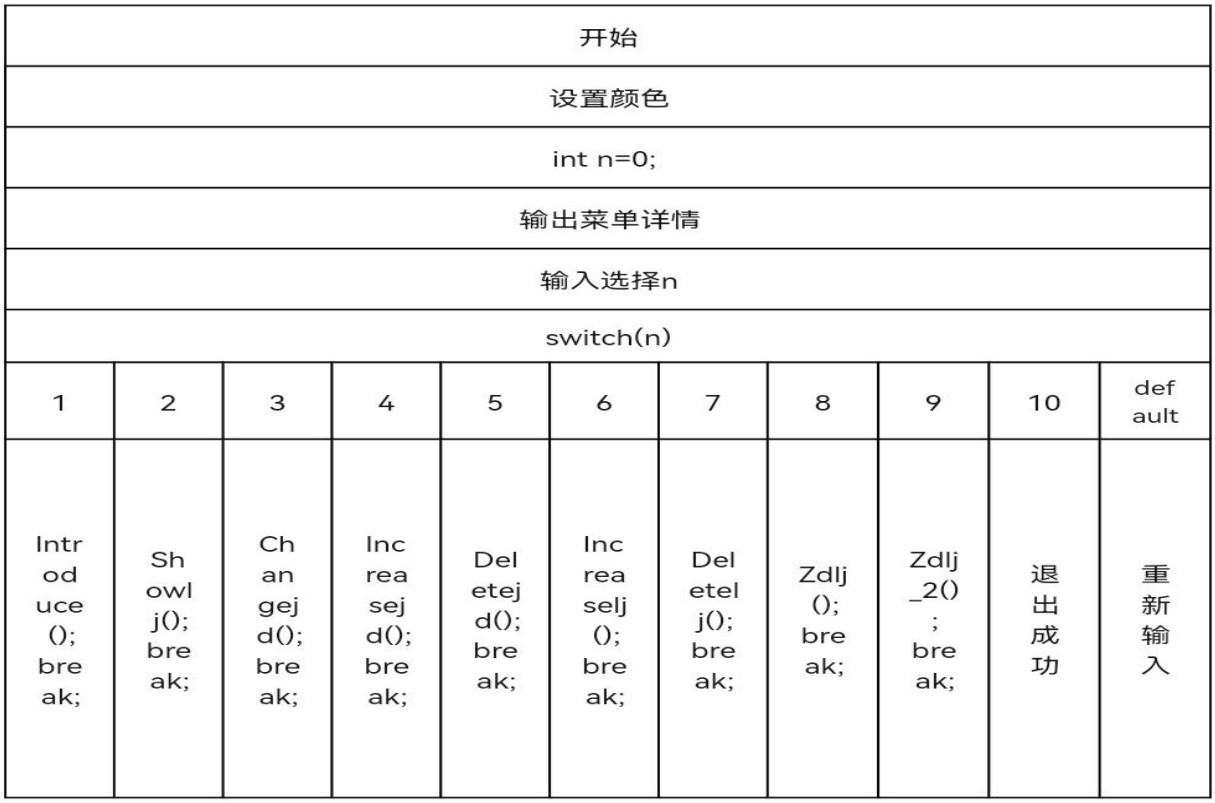
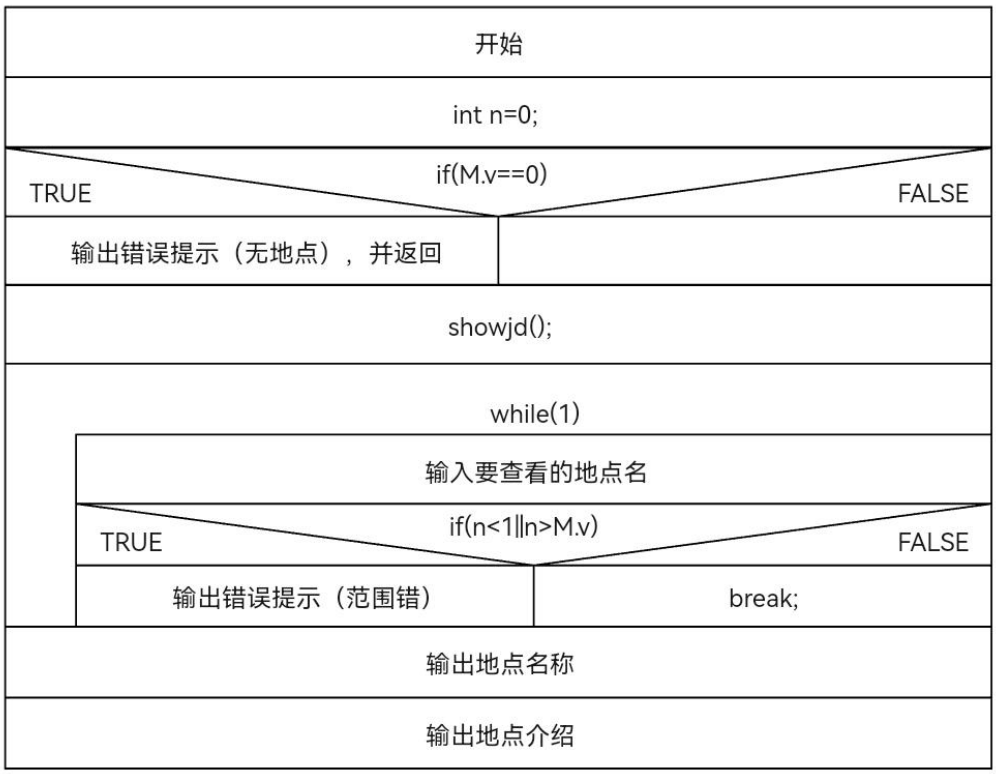


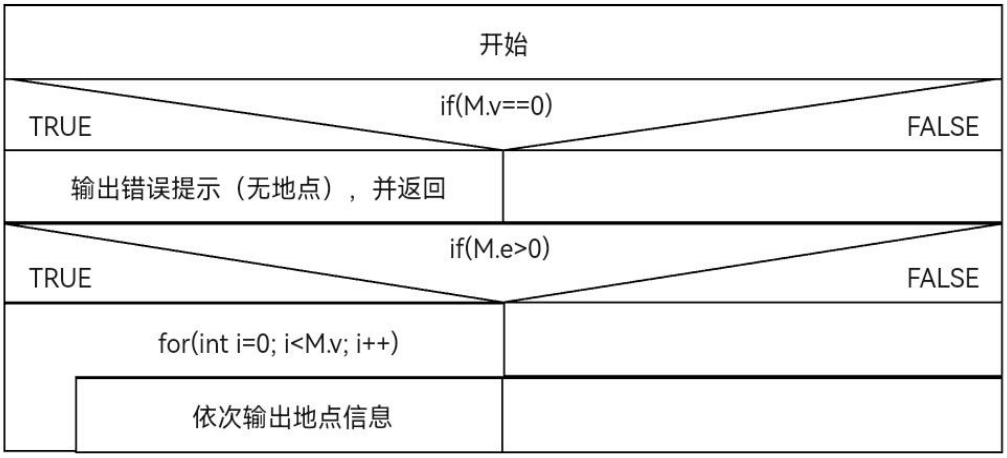
图 4.3 菜单界面 N-S 图

4.6 查看已有地点子函数

先通过当前节点数的数值来判断是否有地点可以查看，再判断地点代号是否符合范围，最后都符合后利用数组信息调出来展示相应信息



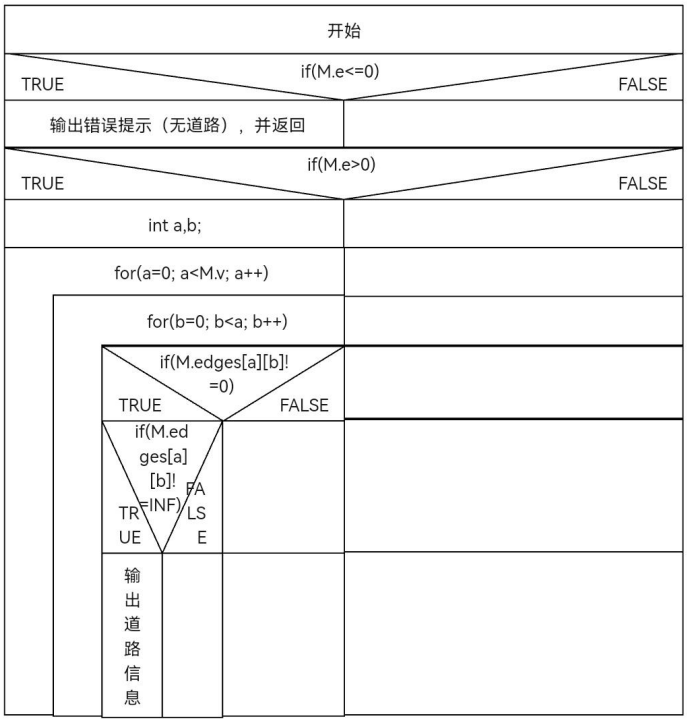
4.4 查看已有地点详情 N-S 图



4.5 展示已有地点 N-S 图

4.7 查看已有道路子函数

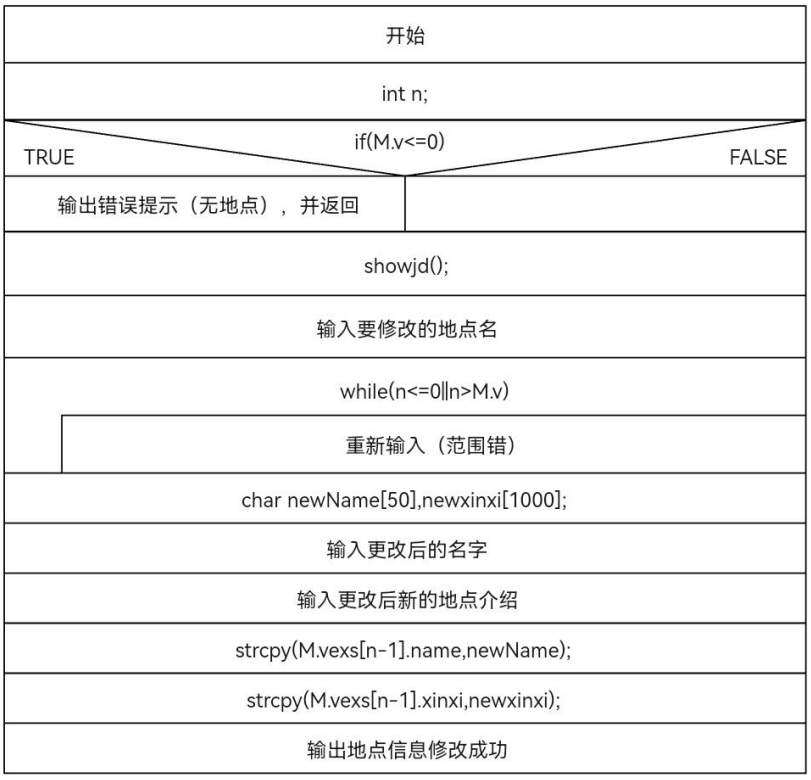
先通过当前边数的数值来判断是否有道路可以查看，可以查看时通过 2 个 for 循环嵌套来输出构造的二维数组的下三角内容来实现边的长度以及所有无向边不重复的输出



4.6 查看已有道路 N-S 图

4.8 修改已有地点子函数

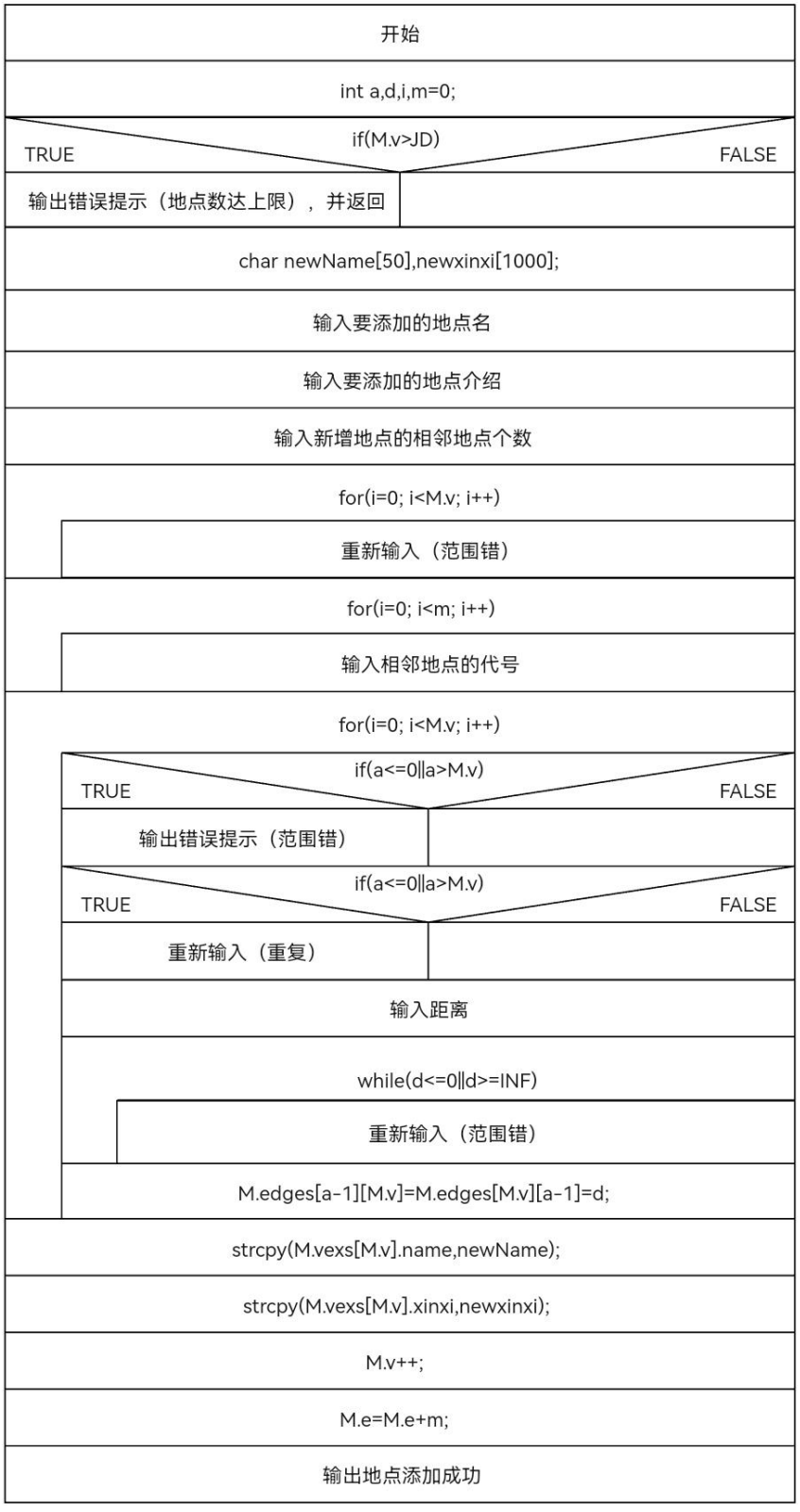
先通过当前节点数的数值来判断是否有地点可以更改，再判断地点代号是否符合范围，符合的话，通过字符串复制语句(strcpy)来将更改后的信息写入一维数组中实现修改



4.7 修改已有地点 N-S 图

4.9 增加新的地点子函数

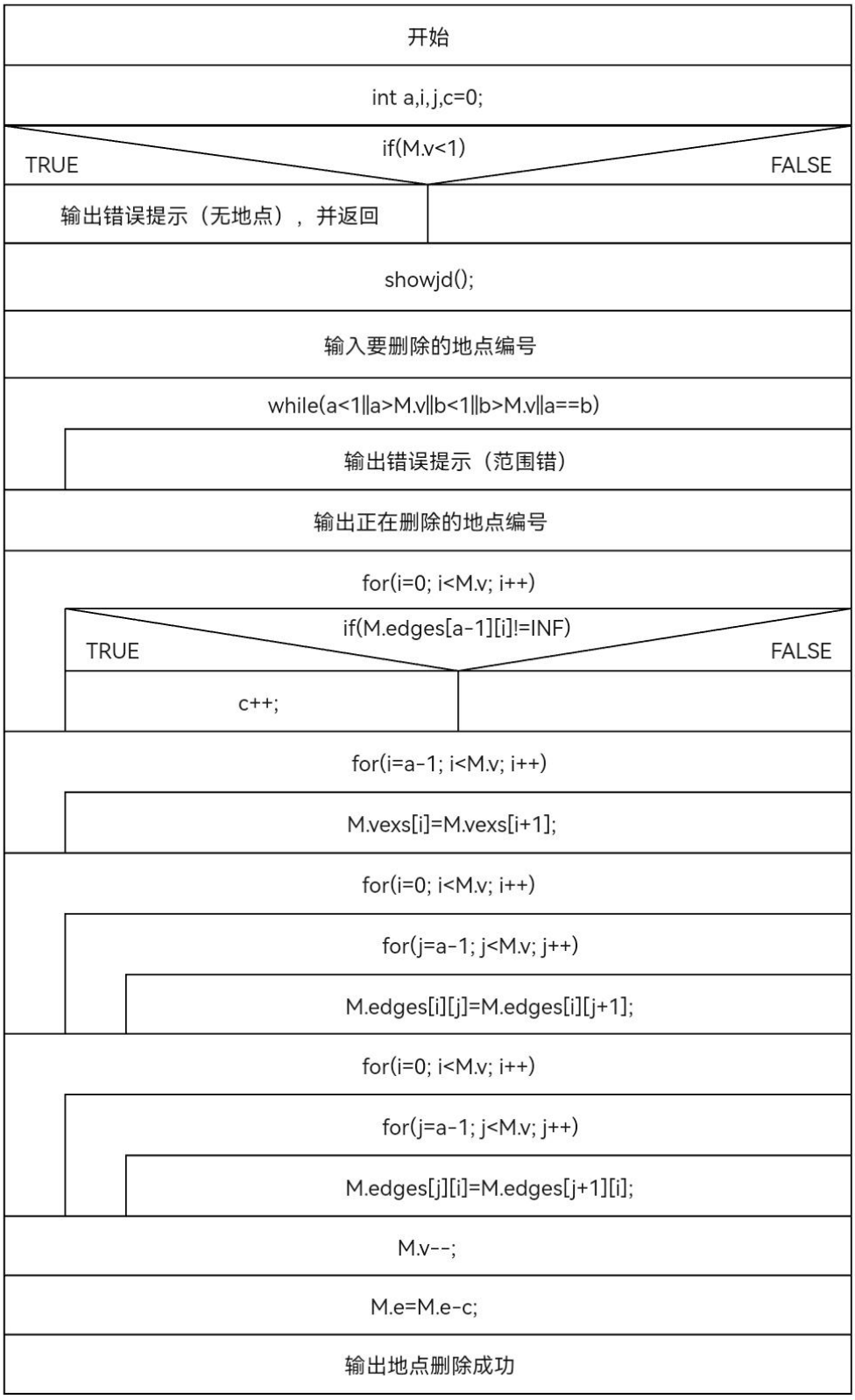
先通过当前节点数的数值来判断是否可以添加地点，可以的话通过字符串复制语句(strcpy)来写入新的地点信息，在通过输入相邻的节点数来将数据准确存储在对应的一维数组位置上



4.8 增加新的地点 N-S 图

4.10 删除现有地点子函数

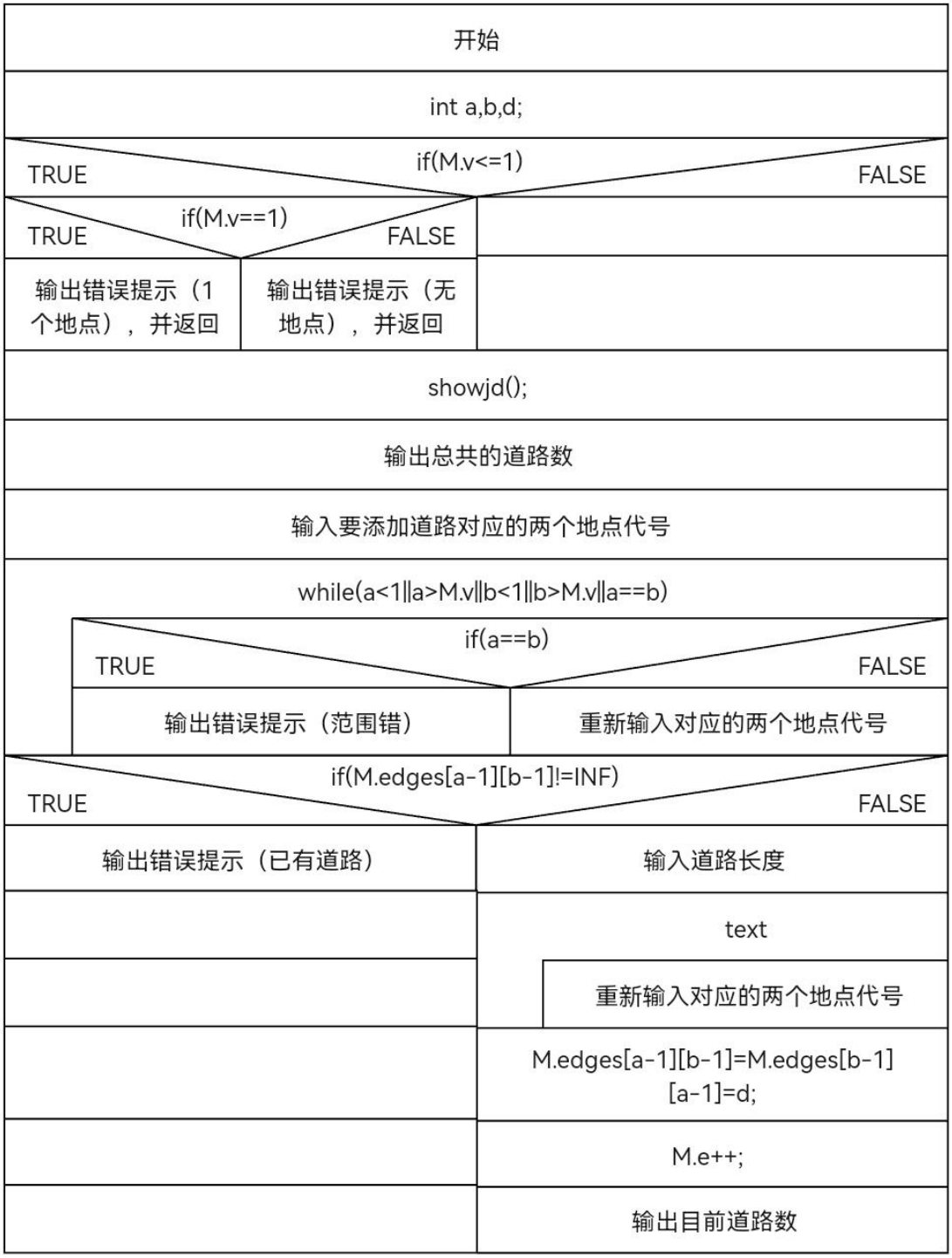
先通过当前节点数的数值来判断是否可以删除地点，可以删除时通过分别对无向图的 2 条无向边进行更改来实现



4.9 删除现有地点 N-S 图

4.11 增加新的道路子函数

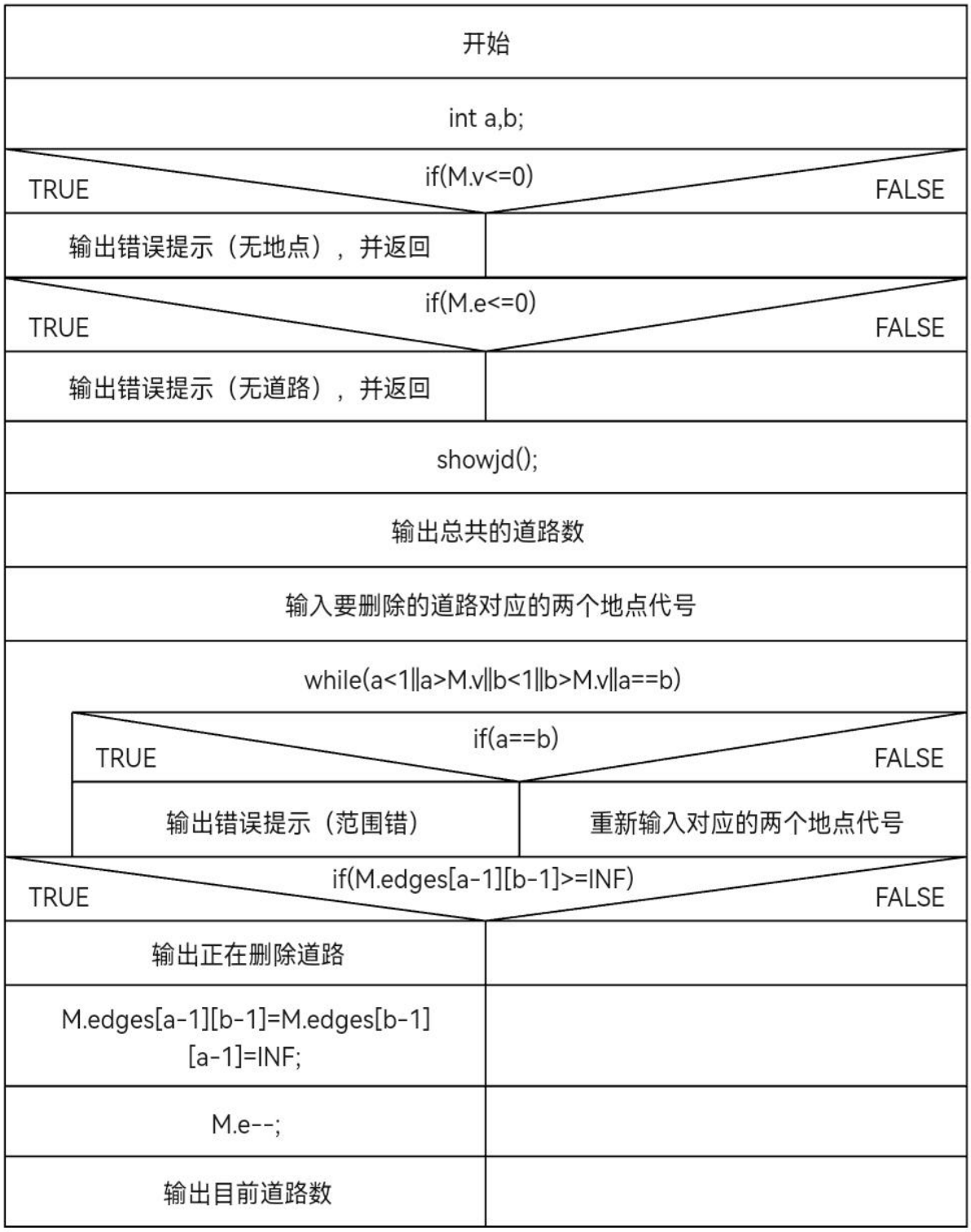
先通过当前节点数的数值来判断是否可以添加道路，并对输入的地点代号也进行区间判断，符合条件时写入二维数组边的长度



4.10 增加新的道路 N-S 图

4.12 删除现有道路子函数

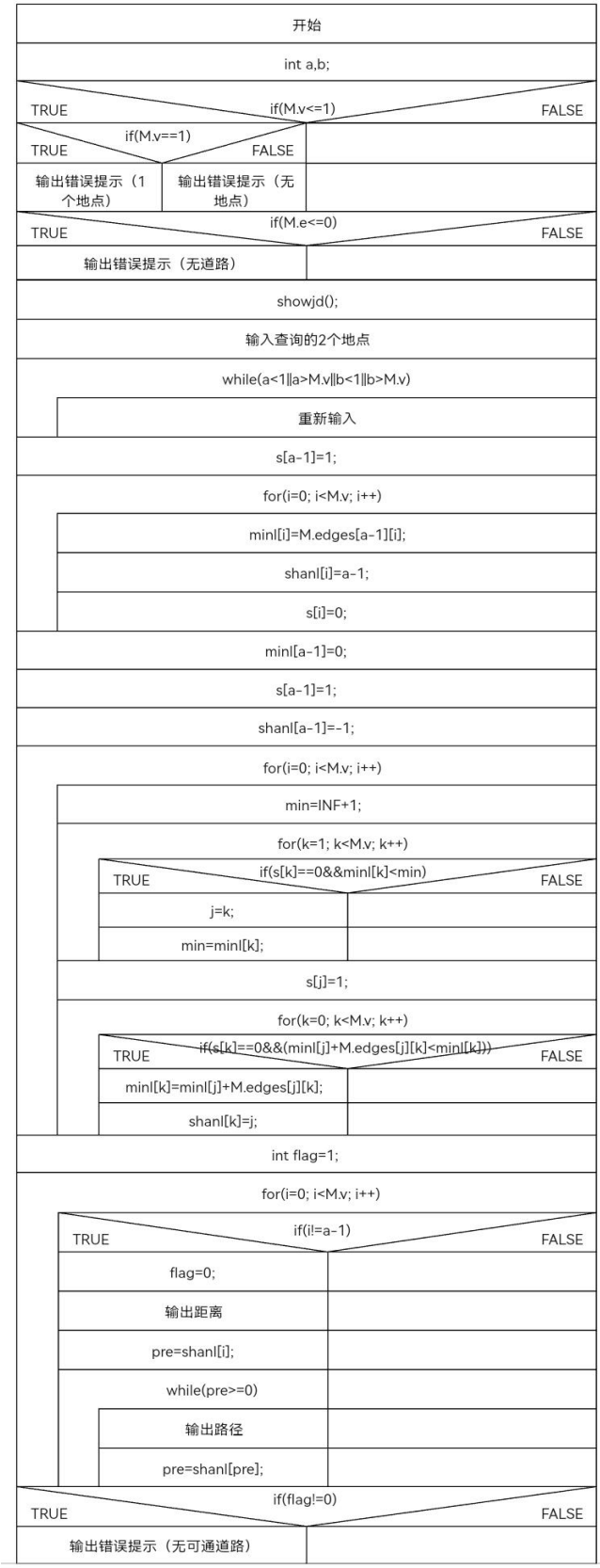
先通过当前节点数的数值和边数的数值来判断是否可以删除道路，当可以时将边的长度设为 INF 的值（假定为无穷路径）即代表道路删除成功



4.11 删除现有道路 N-S 图

4.13 查找当前地点到其它地点的最短路径子函

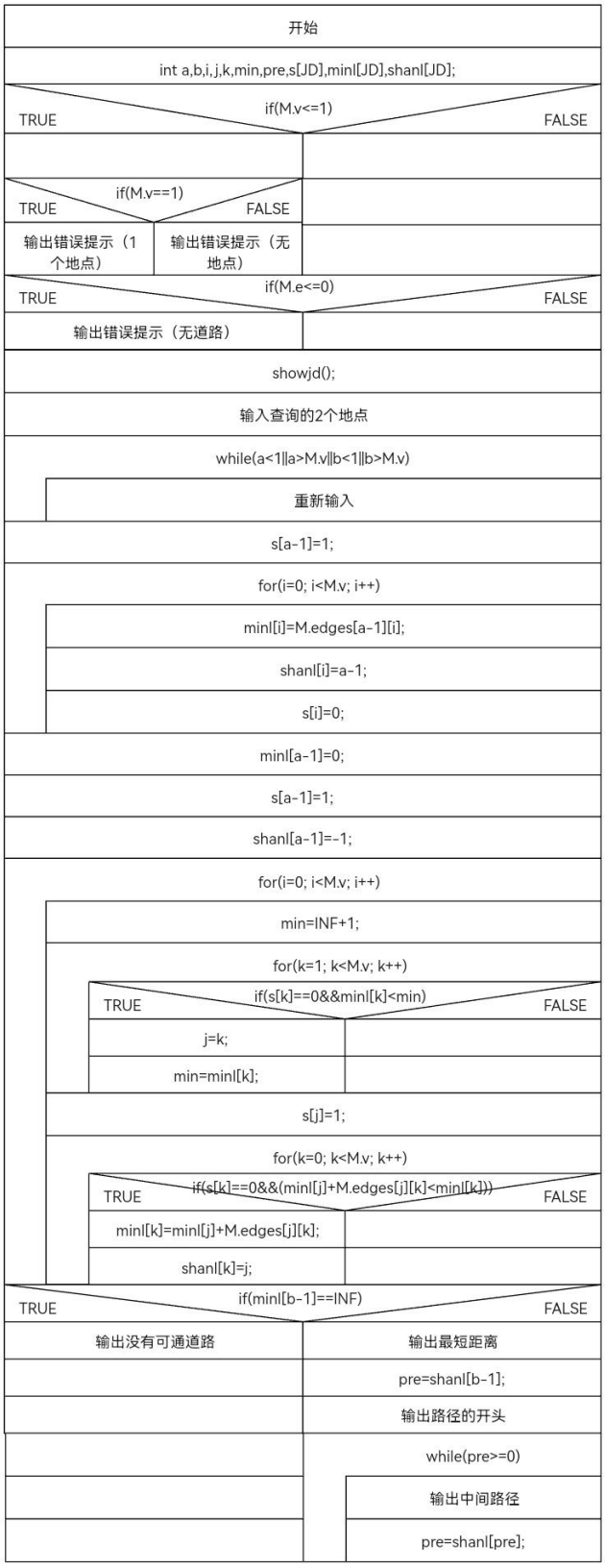
用 Dijkstra 算法求无向网 M 中定点到其余定点的最短路径及其带权长度



4.12 查找当前地点到其它地点的最短路径 N-S 图

4.14 查找任一的两个地点之间的最短路径子函数

用 Dijkstra 算法求无向网 M 中定点到另一定点的最短路径及其带权长度



4.13 查找 2 地点间的最短路径 N-S 图

5 程序运行数据及其结果

本程序采用的数据来源是即可以键盘直接输入，也可以用已有数据。

输入 1、2、3、4、5、6、7、8、9 或 10，选择要使用的功能。

图 5.1　初始运行菜单页面

查看已有地点子函数界面，输入相应数字查看详情

图 5.2 查看已有地点子函数菜单页面

输入想要查询的地点的编号

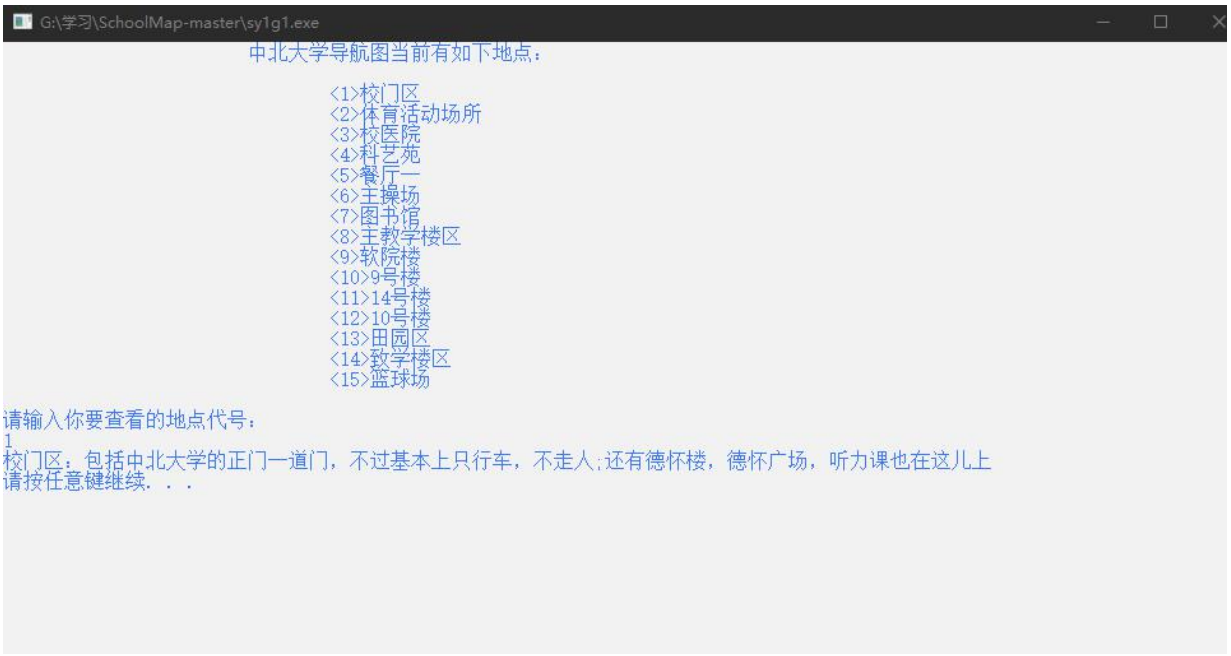


图 5.3 查看已有地点详情页面

图 5.4 查看已有道路详情页面

输入要改地点序号后，依照提示输入进行更改

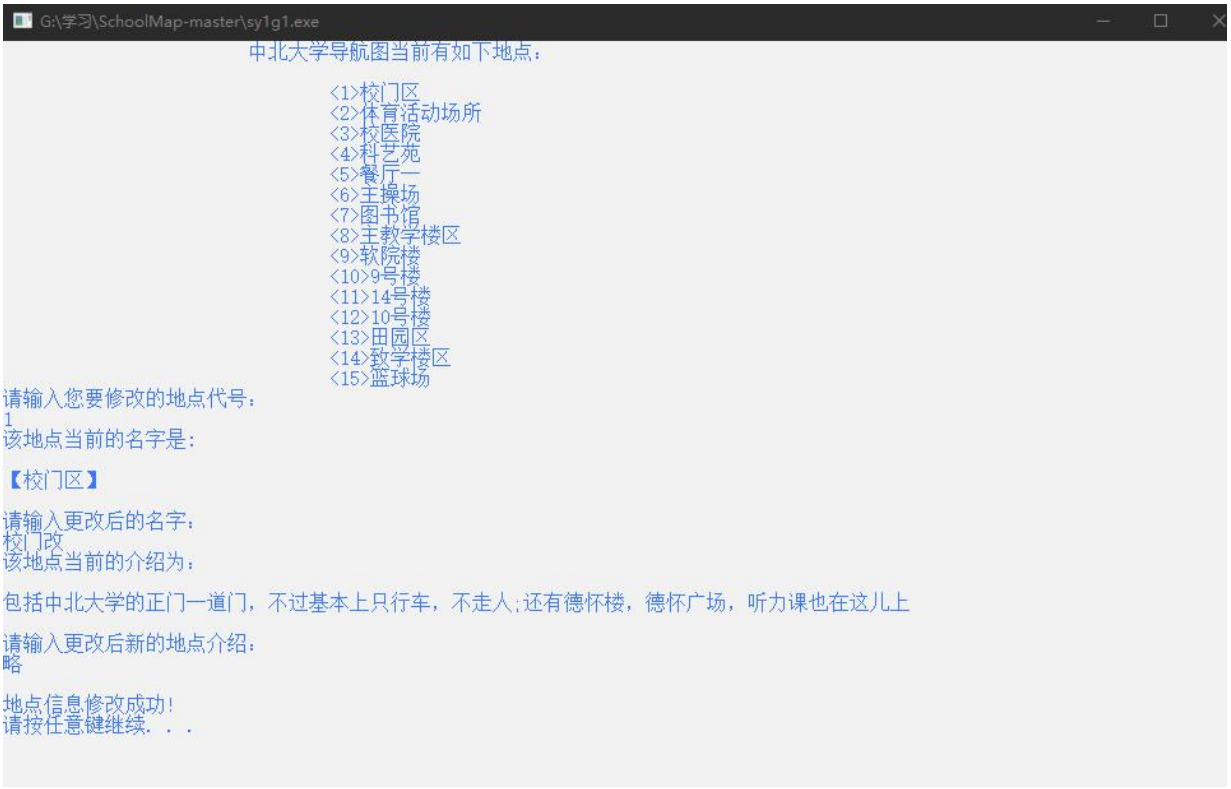


图 5.5 修改已有地点详情页面

输入要添加地点序号后，依照提示输入进行添加

图 5.6 添加地点详情页面

输入要删除地点序号，即可删除成功

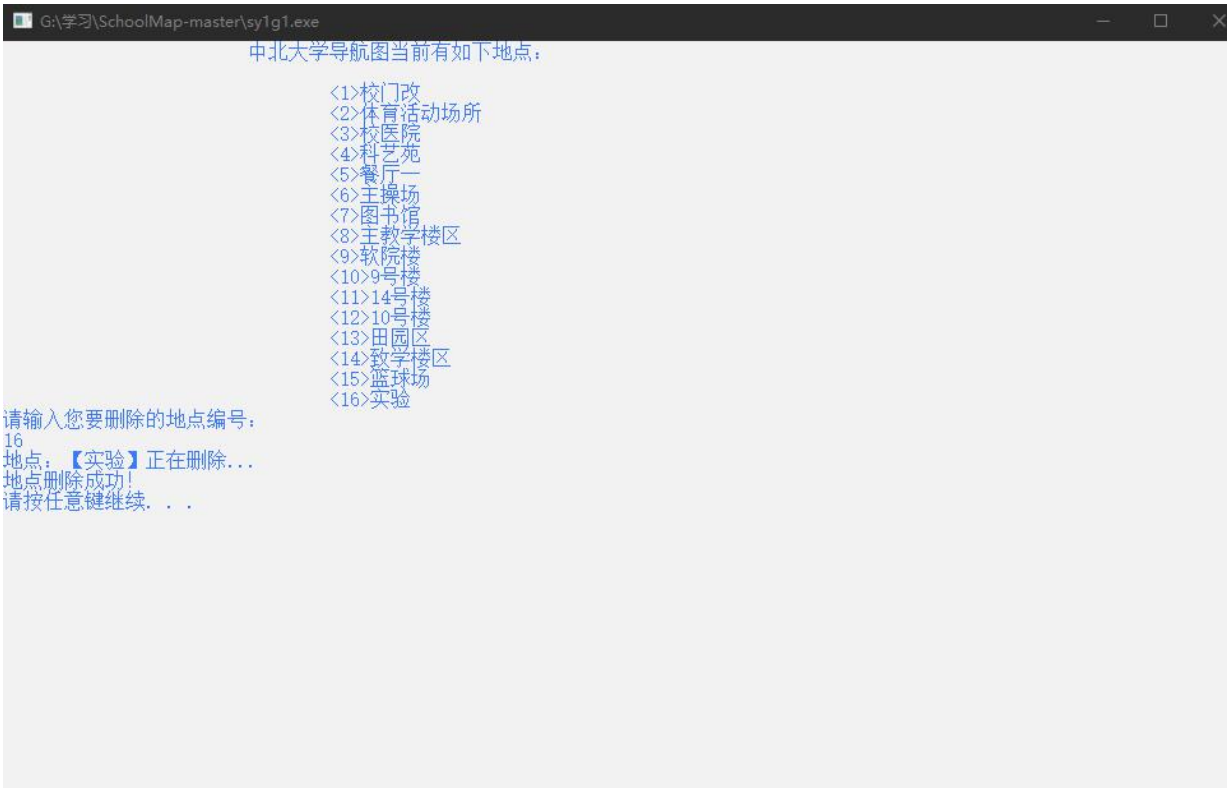


图 5.7 删除地点详情页面

输入要添加道路的两个地点代号，依照提示输入进行添加

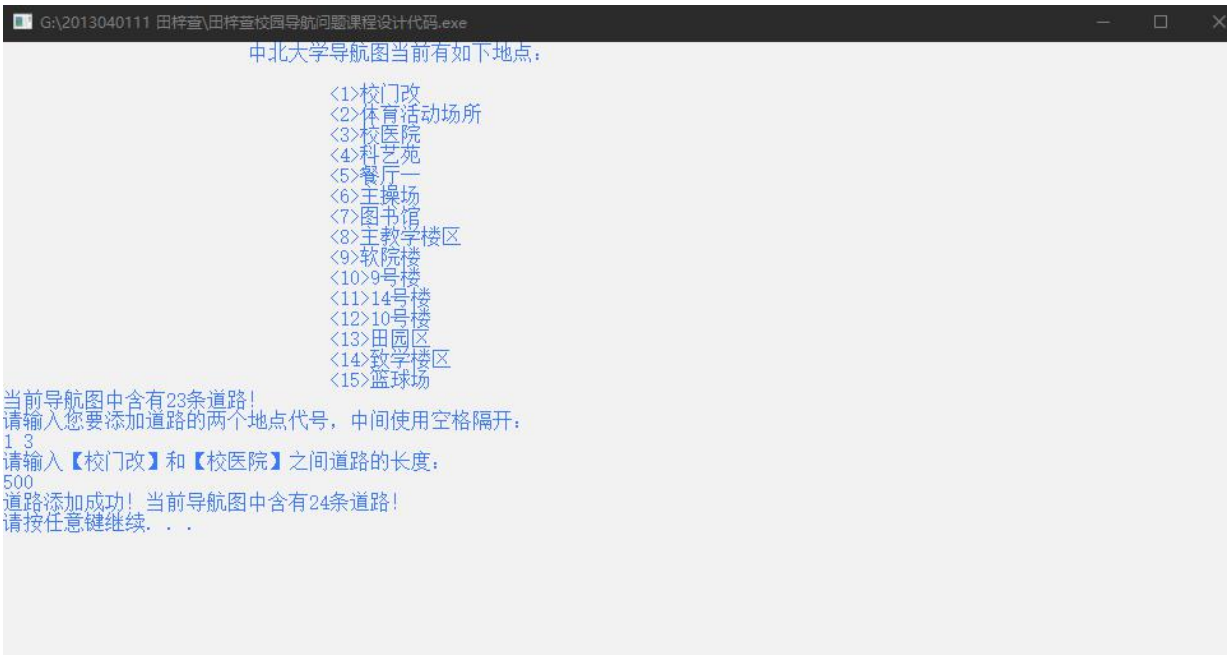


图 5.8 添加道路详情页面

输入要删除道路的两个地点代号，依照提示输入进行添加

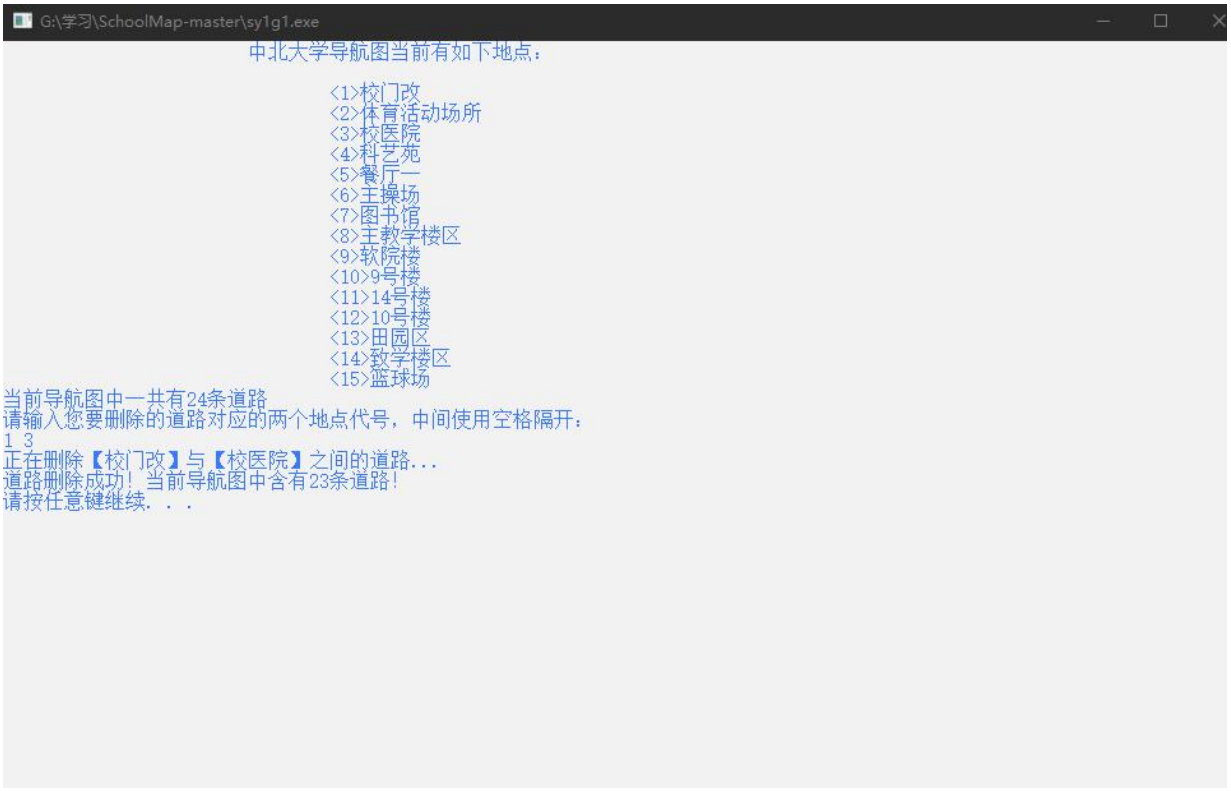


图 5.9 删除道路详情页面

输入当前地点以查找

图 5.10 查找当前地点到其它地点的最短路径详情页面

输入 2 个地点代号以查找

图 5.11 查找任一的两个地点之间的最短路径详情页面

输入 10 以退出

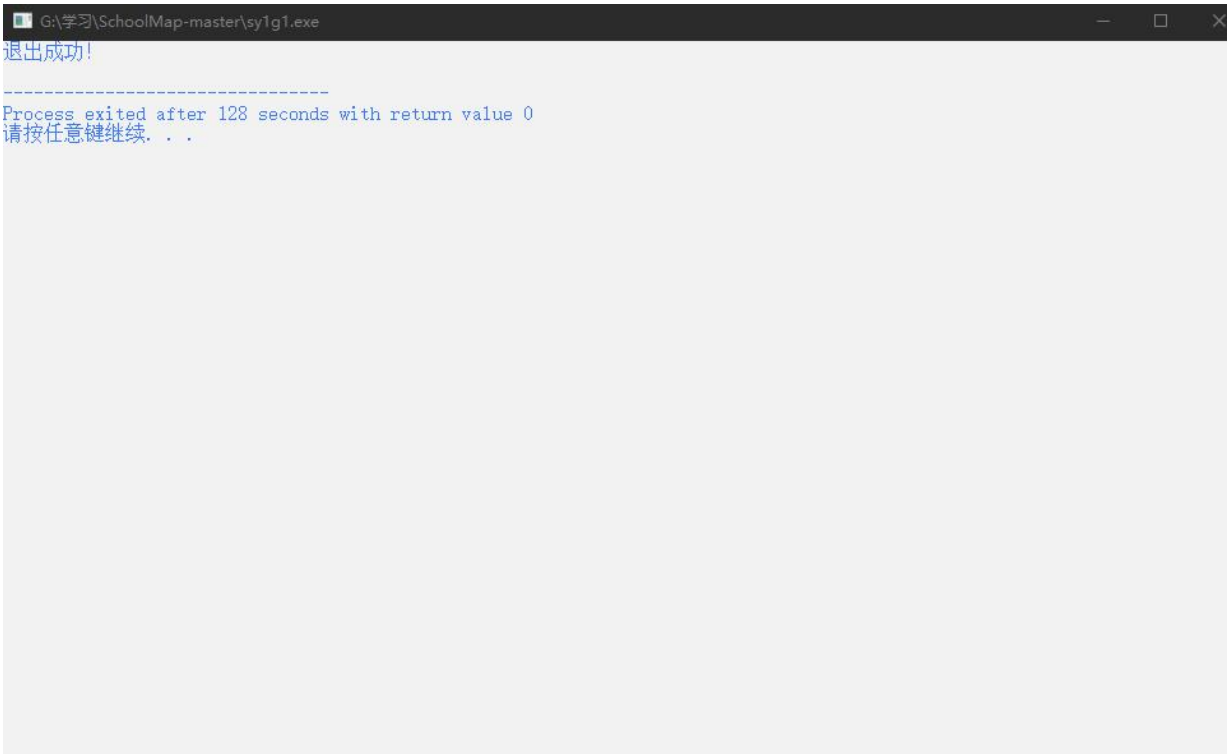


图 5.12 退出

图 5.10 和图 5.11 的结果证明如下表：

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	230 (1,2)												
3			770 (1,4,3)	770 (1,4,3)	770 (1,4,8)								
4	470 (1,4)	470 (1,4)											
5			570 (1,4,5)										
6				750 (1,4,5,6)									
7			870 (1,4,7)	870 (1,4,7)	870 (1,4,7)	870 (1,7,4)	870 (1,7,4)						
8		770 (1,2,8)	770 (1,2,8)	770 (1,2,8)	770 (1,2,8)								
9						850 (1,2,8,9)							
10						960 (1,2,8,10)	960 (1,2,8,10)	960 (1,2,8,10)					
11							1050 (1,2,8,9,11)	1050 (1,2,8,9,11)	1050 (1,2,8,9,11)				
12										1300 (1,2,8,9,11,12)	1300 (1,2,8,9,11,12)	1300 (1,2,8,9,11,12)	
13											1510 (1,2,8,9,14,13)	1510 (1,2,8,9,14,13)	1510 (1,2,8,9,14,13)
14							1160 (1,2,8,9,14)	1160 (1,2,8,9,14)	1160 (1,2,8,9,14)	1160 (1,2,8,9,14)			
15										1250 (1,2,8,9,11,15)	1250 (1,2,8,9,11,15)		
确定的节点	2	4	5	6	3,8	9	7	10	11	14	15	12	13

图 5.12 结果证明表

拟创建的中北大学校园导航图：

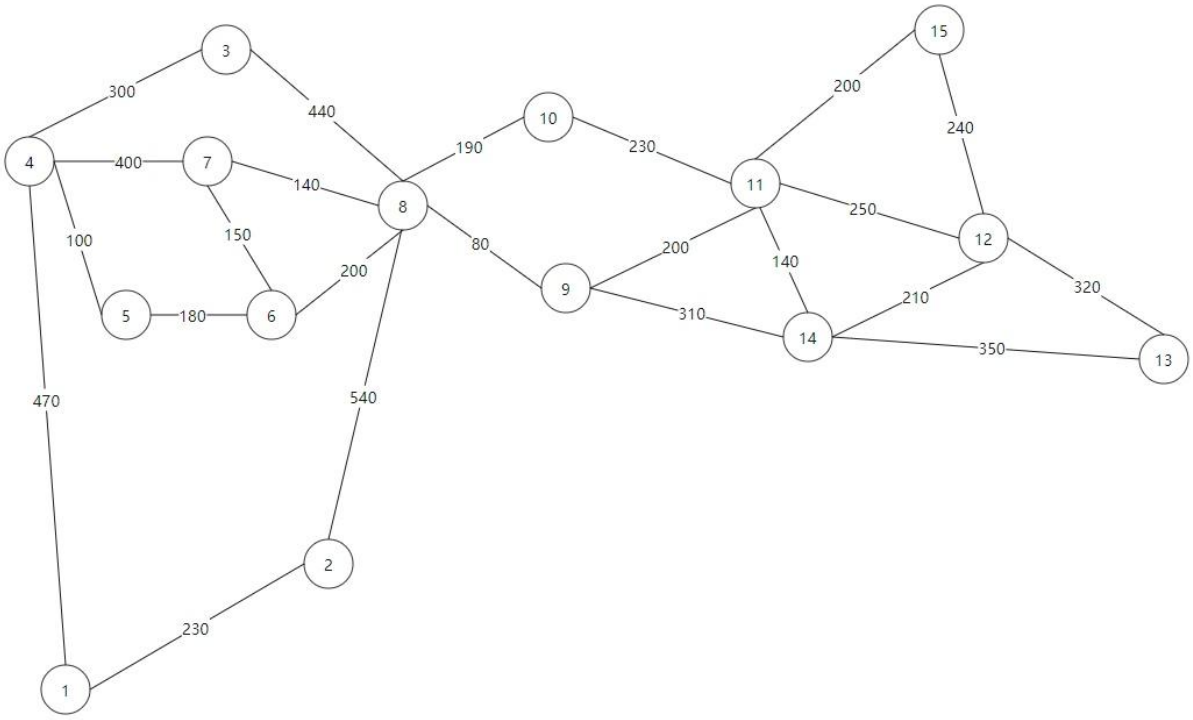


图 5.12 校园导航图

6 课程设计心得

这次课程设计中，我的收获就是学会了用 N-S 图来表达自己的想法，并根据 N-S 图来逐步实现程序的功能，以及函数之间的调用（包括递归调用）更加深刻的了解。开始的时候，我写菜单界面一直跳转有误，后来终于解决了。并且一开始我画 N-S 图较困难，需要近一个小时才能清楚的根据自己的想法画出图来，后来画多了，就更加了解它的功能，十分得心应手，能够比较快而准确的画出来。

在这次课程设计中，我首先对系统的整体功能进行了构思，然后用结构化分析方法进行分析，将整个系统清楚的划分为 10 个模块，再根据每个模块的功能编写代码。而且尽可能的将模块细分，最后在进行函数的调用。我在函数的编写过程中，我用到了 for 循环、while 循环和 switch 语句，函数之间的调用（包括递归调用），并且对于 getchar()的作用有了更深的理解，而且还参考网上资料调用颜色函数使得界面更美观。虽然在调试的过程中也遇到了一些困难，但经过我耐心的修改，终于功夫不负有心人，我成功实现所有功能，但是还是有所不足，比如如果输入数据类型错误的判断有待改进。

附录：

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <windows.h>
#include <string.h>
#define JD 50
#define INF 999999
typedef struct {
    char name[JD];/*定义地点名称*/
    char xinxi[1000];/*定义地点信息*/
} VertexType;/*地图信息的定义*/
typedef struct {
    VertexType vexs[JD];
    int edges[JD][JD];
    int v,e;
}MGraph;/*地图（无向图）的边和节点的定义*/
static MGraph M;
void setColor();
void create();
void showjd();
void Showlj();
void jiemian();
void Introduce();
void Changejd();
void Increasejd();
void Deletejd();
void Increaselj();
void Deletelj();
void Zdlj();
void Zdlj_2();
int main(){
    create();
    while(1){
        system("cls");
        jiemian();
    }
    return 0;
```

```
}
void setColor(unsigned short ForeColor,unsigned short BackGroundColor){
    HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(handle, ForeColor + BackGroundColor * 0x10);
}/*设置输出界面的颜色的函数*/
void jiemian(){
    setColor(9, 15);
    system("cls");
    int n=0;
    printf("\n\n                欢迎光临中北大学主校区校园导航中心！        \n\n");
    printf("\n                    请选择您要进行的操作：                \n");
    printf("\n                1.查看已有地点                            \n");
    printf("                2.查看已有道路                            \n");
    printf("                3.修改已有地点                            \n");
    printf("                4.增加新的地点                            \n");
    printf("                5.删除现有地点                            \n");
    printf("                6.增加新的道路                            \n");
    printf("                7.删除现有道路                            \n");
    printf("                8.查找当前地点到其它地点的最短路径        \n");
    printf("                9.查找任一的两个地点之间的最短路径        \n");
    printf("                10.退出系统！                             \n");
    printf("\n                    注意:请务必输入整型数字！                \n\n");
    scanf("%d",&n);
    getchar();
    switch(n){
        case 1: Introduce(); break;
        case 2: Showlj(); break;
        case 3: Changejd(); break;
        case 4: Increasejd(); break;
        case 5: Deletejd(); break;
        case 6: Increaselj(); break;
        case 7: Deletelj(); break;
        case 8: Zdlj(); break;
        case 9: Zdlj_2(); break;
        case 10: system("cls"); printf("退出成功！\n"); exit(0);
        default: printf("您的输入有误，请重新输入！\n"); system("pause"); break;
```

```
    }
}
void create() {
    M.v=15;//已录入的地点数量
    M.e=23;//已录入地点的道路的数量
    int i,j,k;
    for(i=0; i<JD; i++){
        for(j=0; j<JD; j++){
            M.edges[i][j]=INF;
        }
    }
    strcpy(M.vexs[0].name,"校门区");
    strcpy(M.vexs[0].xinxi,"包括中北大学的正门一道门，不过基本上只行车，不走人;还有德怀楼，德怀广场，听力课也在这儿上");
    strcpy(M.vexs[1].name,"体育活动场所");
    strcpy(M.vexs[1].xinxi,"包括一个大的标准足球场和乒乓球馆，乒乓球课在此处上课");
    strcpy(M.vexs[2].name,"校医院");
    strcpy(M.vexs[2].xinxi,"打疫苗、体检以及生病时来这儿");
    strcpy(M.vexs[3].name,"科艺苑");
    strcpy(M.vexs[3].xinxi,"文艺表演的地方");
    strcpy(M.vexs[4].name,"餐厅一");
    strcpy(M.vexs[4].xinxi,"饭品种多，但价格也随之上涨，有超市、药店、唯一一个银行、杂货店");
    strcpy(M.vexs[5].name,"主操场");
    strcpy(M.vexs[5].xinxi,"最大的一个标准化操场，有真草坪的足球场，适合锻炼身体");
    strcpy(M.vexs[6].name,"图书馆");
    strcpy(M.vexs[6].xinxi,"书非常多，分专业存放，很好的自习场所");
    strcpy(M.vexs[7].name,"主教学楼区");
    strcpy(M.vexs[7].xinxi,"楼顶有中北大学四个大字的楼，也就是 11 号教学楼，很多课在这儿上，空教室也很多，楼前有一个大广场（行知广场），很多活动在这儿举行");
    strcpy(M.vexs[8].name,"软院楼");
    strcpy(M.vexs[8].xinxi,"今年新有的软件学院牌子，楼不大，但里面人才不少哦");
    strcpy(M.vexs[9].name,"9 号楼");
    strcpy(M.vexs[9].xinxi,"很多课也在这儿上，教室较多，理学院主要在这儿");
    strcpy(M.vexs[10].name,"14 号楼");
    strcpy(M.vexs[10].xinxi,"四层楼大教室，在这儿的课少");
```

```
    strcpy(M.vexs[11].name,"10 号楼");
    strcpy(M.vexs[11].xinxi,"暂时没来这儿上过课，只在这儿考过试，离宿舍最近的教学楼了");
    strcpy(M.vexs[12].name,"田园区");
    strcpy(M.vexs[12].xinxi,"包括文澜宿舍以及田园食堂，田园实验室，食堂吃的不错，至少便宜，食堂上是实验室，从这儿走出一个又一个的软件人才。");
    strcpy(M.vexs[13].name,"致学楼区");
    strcpy(M.vexs[13].xinxi,"包括致学广场和致学楼，有卖纪念品的地方，还有个小食堂");
    strcpy(M.vexs[14].name,"篮球场");
    strcpy(M.vexs[14].xinxi,"打篮球的地方");
    M.edges[0][1]=M.edges[1][0]=230;
    M.edges[0][3]=M.edges[3][0]=470;
    M.edges[1][7]=M.edges[7][1]=540;
    M.edges[2][3]=M.edges[3][2]=300;
    M.edges[2][7]=M.edges[7][2]=440;
    M.edges[3][4]=M.edges[4][3]=100;
    M.edges[3][6]=M.edges[6][3]=400;
    M.edges[4][5]=M.edges[5][4]=180;
    M.edges[5][6]=M.edges[6][5]=150;
    M.edges[5][7]=M.edges[7][5]=200;
    M.edges[6][7]=M.edges[7][6]=140;
    M.edges[7][8]=M.edges[8][7]=80;
    M.edges[7][9]=M.edges[9][7]=190;
    M.edges[8][10]=M.edges[10][8]=200;
    M.edges[8][13]=M.edges[13][8]=310;
    M.edges[9][10]=M.edges[10][9]=230;
    M.edges[10][11]=M.edges[11][10]=250;
    M.edges[10][13]=M.edges[13][10]=140;
    M.edges[10][14]=M.edges[14][10]=200;
    M.edges[11][12]=M.edges[12][11]=320;
    M.edges[11][13]=M.edges[13][11]=210;
    M.edges[11][14]=M.edges[14][11]=240;
    M.edges[12][13]=M.edges[13][12]=350;
}
void showjd(){
    if(M.v==0){
        printf("          当前导航图中没有地点！\n\n");
        system("pause");
```

```
            return ;
        }
        if(M.v>0){
            printf("                中北大学导航图当前有如下地点：    \n\n");
            for(int i=0; i<M.v; i++){
                printf("\t\t            <%d>%s        \n",i+1,M.vexs[i].name);
            }
        }
}/*展示图中所有节点*/
void Showlj(){
        system("cls");
        if(M.e<=0){
            printf("          当前导航图中没有道路！\n\n");
            system("pause");
            return ;
        }
        if(M.e>0){
            printf("                中北大学导航图当前有如下%d 条道路：    \n\n",M.e);
            int a,b;
            for(a=0; a<M.v; a++){
                for(b=0; b<a; b++){
                    if(M.edges[a][b]!=0){
                        if(M.edges[a][b]!=INF){
                            printf("\t      【 %s 】      <---->    【 %s 】 ，距 离 是 %d  米
\n",M.vexs[a].name,M.vexs[b].name,M.edges[a][b]);
                        }
                    }
                }
            }
        }
        system("pause");
}/*展示无向图中所有边*/
void Introduce(){
        system("cls");
        int n=0;
        if(M.v==0){
            printf("本导航图暂时无地点！\n\n");
```

```
            system("pause");
            return ;
        }
        showjd();
        printf("\n 请输入你要查看的地点代号：\n");
        while(1){
            scanf("%d",&n);
            if(n<1||n>M.v) printf("您的输入有误，请重新输入！\n");
            else break;
        }
        printf("%s：",M.vexs[n-1].name);
        printf("%s\n",M.vexs[n-1].xinxi);
        system("pause");
}/*查看某一地点详情*/
void Changejd(){
        system("cls");
        int n;
        if(M.v<=0){
            printf("导航图中无地点，无法操作！\n");
            system("pause");
            return ;
        }
        showjd();
        printf("请输入您要修改的地点代号：\n");
        scanf("%d",&n);
        while(n<=0||n>M.v){
            printf("您的输入有误，请重新输入！\n");
            scanf("%d",&n);
        }
        char newName[50],newxinxi[1000];
        printf("该地点当前的名字是:\n\n【%s】\n\n 请输入更改后的名字：\n",M.vexs[n-1].name);
        scanf("%s",newName);
        getchar();
        printf("该地点当前的介绍为：\n\n%s\n\n 请输入更改后新的地点介绍：\n",M.vexs[n-1].xinxi);
        scanf("%s",newxinxi);
        getchar();
        strcpy(M.vexs[n-1].name,newName);
```

```
    strcpy(M.vexs[n-1].xinxi,newxinxi);
    printf("\n 地点信息修改成功!\n");
    system("pause");
}/*改变地点详情*/
void Increasejd(){
    system("cls");
    int a,d,i,m=0;
    if(M.v>JD){
        printf("地点已达最大上限 50 个，当前无法添加景点！\n");
        system("pause");
        return ;
    }
    char newName[50],newxinxi[1000];
    printf("请输入您要添加的地点名：\n");
    scanf("%s",newName);
    getchar();
    printf("请输入【%s】地点的介绍，最多可输入 200 字：\n",newName);
    scanf("%s",newxinxi);
    getchar();
    showjd();
    printf("请输入新增地点的相邻地点个数:\n");
    scanf("%d",&m);
    while(m<0||m>M.v){
        printf("您的输入有误，请重新输入！\n");
        scanf("%d",&m);
    }
    for(i=0; i<m; i++){
        printf("请输入第%d 个相邻地点的代号：\n",i+1);
        scanf("%d",&a);
        while(a<=0||a>M.v||M.edges[a-1][M.v]!=INF){
            if(a<=0||a>M.v) printf("您的输入有误，请重新输入！范围在 1~%d 之间。\n",M.v);
            if(M.edges[a-1][M.v]!=INF) printf("请不要输入重复的相邻地点，重新输入：\n");
            scanf("%d",&a);
        }
        printf("请输入【%s】与【%s】之间的距离：\n",newName,M.vexs[a-1].name);
        scanf("%d",&d);
        while(d<=0||d>=INF){
```

```
            printf("您输入的距离有误！请重新输入：\n");
            scanf("%d",&d);
        }
        M.edges[a-1][M.v]=M.edges[M.v][a-1]=d;
    }
    strcpy(M.vexs[M.v].name,newName);
    strcpy(M.vexs[M.v].xinxi,newxinxi);
    M.v++;
    M.e=M.e+m;
    printf("地点添加成功!\n");
    system("pause");
}/*增加新的节点*/
void Deletejd(){
    system("cls");
    int a,i,j,c=0;
    if(M.v<1){
        printf("导航图中无地点，无法删除！\n");
        system("pause");
        return ;
    }
    showjd();
    printf("请输入您要删除的地点编号：\n");
    scanf("%d",&a);
    while(a<1||a>M.v){
        printf("您的输入有误，请重新输入！范围在 1~%d 之间。\n",M.v);
        scanf("%d",&a);
    }
    printf("地点：【%s】正在删除...\n",M.vexs[a-1].name);
    for(i=0; i<M.v; i++){
        if(M.edges[a-1][i]!=INF) c++;
    }
    for(i=a-1; i<M.v; i++){
        M.vexs[i]=M.vexs[i+1];
    }
    for(i=0; i<M.v; i++){
        for(j=a-1; j<M.v; j++) M.edges[i][j]=M.edges[i][j+1];
    }
```

```
    for(i=0; i<M.v; i++){
        for(j=a-1; j<M.v; j++) M.edges[j][i]=M.edges[j+1][i];
    }
    M.v--;
    M.e=M.e-c;
    printf("地点删除成功！\n");
    system("pause");
}/*删除节点*/
void Increaselj(){
    system("cls");
    int a,b,d;
    if(M.v<=1){
        if(M.v==1){
            printf("导航图中只有一个地点，无法添加道路！\n");
            system("pause");
            return ;
        }else{
            printf("导航图中无地点，无法添加道路！\n");
            system("pause");
            return ;
        }
    }
    showjd();
    printf("当前导航图中含有%d 条道路！\n",M.e);
    printf("请输入您要添加道路的两个地点代号，中间使用空格隔开：\n");
    scanf("%d %d",&a,&b);
    while(a<1||a>M.v||b<1||b>M.v||a==b){
        if(a==b) printf("您输入的地点代号相同，请重新输入！\n");
        else printf("您的输入有误，请重新输入！范围在 1~%d 之间。\n",M.v);
        scanf("%d %d",&a,&b);
    }
    if(M.edges[a-1][b-1]!=INF) printf("【%s】与【%s】之间已经存在一条道路，无需再次添加！
\n",M.vexs[a-1].name,M.vexs[b-1].name);
    else{
        printf("请输入【%s】和【%s】之间道路的长度：\n",M.vexs[a-1].name,M.vexs[b-1].name);
        scanf("%d",&d);
        while(d<=0||d>=INF){
```

```
            printf("您输入的长度有误，请重新输入！\n");
            scanf("%d",&d);
        }
        M.edges[a-1][b-1]=M.edges[b-1][a-1]=d;
        M.e++;
        printf("道路添加成功！当前导航图中含有%d 条道路！\n",M.e);
    }
    system("pause");
}/*增加新的边*/
void Deletelj(){
    int a,b;
    system("cls");
    if(M.v<=0){
        printf("导航图中无地点，无法删除！\n");
        system("pause");
        return ;
    }
    if(M.e<=0){
        printf("导航图中无道路，无法删除！\n");
        system("pause");
        return ;
    }
    showjd();
    printf("当前导航图中一共有%d 条道路\n",M.e);
    printf("请输入您要删除的道路对应的两个地点代号，中间使用空格隔开：\n");
    scanf("%d %d",&a,&b);
    while(a<1||a>M.v||b<1||b>M.v||a==b){
        if(a==b) printf("您输入的地点代号相同，请重新输入！\n");
        else printf("您的输入有误，请重新输入！范围在 1~%d 之间，地点代号不同。\n",M.v);
        scanf("%d %d",&a,&b);
    }
    if(M.edges[a-1][b-1]>=INF)  printf("%s 与 %s 之 间 没 有 道 路 ， 无 法 删 除 ！
\n",M.vexs[a-1].name,M.vexs[b-1].name);
    else{
        printf("正在删除【%s】与【%s】之间的道路...\n",M.vexs[a-1].name,M.vexs[b-1].name);
        M.edges[a-1][b-1]=M.edges[b-1][a-1]=INF;
        M.e--;
```

```
            printf("道路删除成功！当前导航图中含有%d 条道路！\n",M.e);
        }
        system("pause");
}/*删除边*/
void Zdlj(){
        system("cls");
        int a,i,j,k,min,pre;
        int b[JD],minl[JD],shanl[JD];/*存放原来的节点和已生成的终点、存放最短路径的长度、存放上一条路的位置*/
        if(M.v<=1){
            if(M.v==1){
                printf("导航图中只有一个地点，无法查询最短路径！\n");
                system("pause");
                return ;
            }else{
                printf("导航图中无地点，无法查询最短路径！\n");
                system("pause");
                return ;
            }
        }
        if(M.e<=0){
            printf("导航图中无道路，无法查询最短路径！\n");
            system("pause");
            return ;
        }
        showjd();
        printf("请输入您所在地点位置的代号：\n");
        scanf("%d",&a);
        while(a<1||a>M.v){
            printf("您的输入有误，请重新输入！范围在 1~%d 之间。\n",M.v);
            scanf("%d",&a);
        }
        b[a-1]=1;
        for(i=0; i<M.v; i++){
            minl[i]=M.edges[a-1][i];
            shanl[i]=a-1;
            b[i]=0;
```

```
    }
    minl[a-1]=0;
    b[a-1]=1;
    shanl[a-1]=-1;
    for(i=0; i<M.v; i++){
        min=INF+1;
        for(k=1; k<M.v; k++){
            if(b[k]==0&&minl[k]<min){
                j=k;
                min=minl[k];
            }
        }
        b[j]=1;
        for(k=0; k<M.v; k++){
            if(b[k]==0&&(minl[j]+M.edges[j][k]<minl[k])){
                minl[k]=minl[j]+M.edges[j][k];
                shanl[k]=j;
            }
        }
    }
    int flag=1;
    for(i=0; i<M.v; i++){
        if(i!=a-1){
            if(minl[i]!=INF){
                flag=0;
                printf("%d 米: %s",minl[i],M.vexs[i].name);
                pre=shanl[i];
                while(pre>=0){
                    printf("    <--->    %s",M.vexs[pre].name);
                    pre=shanl[pre];
                }
                printf("\n");
            }
        }
    }
    if(flag!=0) printf("【%s】与任何地点之间都没有可通道路！\n",M.vexs[a-1].name);
    system("pause");
}/*迪杰斯特拉算法改*/
```

```
void Zdlj_2(){
    system("cls");
    int a,b,i,j,k,min,pre,s[JD],minl[JD],shanl[JD];/*存放原来的节点和已生成的终点、存放最短路径的长度、存放上一条路的位置*/
    if(M.v<=1){
        if(M.v==1){
            printf("导航图中只有一个地点，无法查询最短路径！\n");
            system("pause");
            return ;
        }else{
            printf("导航图中无地点，无法查询最短路径！\n");
            system("pause");
            return ;
        }
    }
    if(M.e<=0){
        printf("导航图中无道路，无法查询最短路径！\n");
        system("pause");
        return ;
    }
    showjd();
    printf("请输入您要查询距离的两个地点代号,中间用空格隔开：\n");
    scanf("%d %d",&a,&b);
    while(a<1||a>M.v||b<1||b>M.v){
        printf("您的输入有误，请重新输入！范围在 1~%d 之间。\n",M.v);
        scanf("%d %d",&a,&b);
    }
    s[a-1]=1;
    for(i=0; i<M.v; i++){
        minl[i]=M.edges[a-1][i];
        shanl[i]=a-1;
        s[i]=0;
    }
    minl[a-1]=0;
    s[a-1]=1;
    shanl[a-1]=-1;
    for(i=0; i<M.v; i++){
```

```
            min=INF+1;
            for(k=1; k<M.v; k++){
                if(s[k]==0&&minl[k]<min){
                    j=k;
                    min=minl[k];
                }
            }
            s[j]=1;
            for(k=0; k<M.v; k++){
                if(s[k]==0&&(minl[j]+M.edges[j][k]<minl[k])){
                    minl[k]=minl[j]+M.edges[j][k];
                    shanl[k]=j;
                }
            }
        }
        if(minl[b-1]==INF) printf(" 【 %s 】 与 【 %s 】 地 点 之 间 没 有 可 通 道 路 ！
\n",M.vexs[a-1].name,M.vexs[b-1].name);
        else {
            printf(" 【 %s 】 <---> 【 %s 】 之 间 的 最 短 距 离 是 %d 米 。
\n",M.vexs[b-1].name,M.vexs[a-1].name,minl[b-1]);
            pre=shanl[b-1];
            printf("路径为：%s",M.vexs[b-1].name);
            while(pre>=0){
                printf(" <---> %s",M.vexs[pre].name);
                pre=shanl[pre];
            }
        }
        printf("\n");
        system("pause");
}/*迪杰斯特拉算法改版*/
```