第七次实验

学号：33920212204567　　姓名：任宇

一、 实验目的

了解并实现 Dijkstra 和 Floyd 等图的相关算法，培养进行综合性程序设计、据结构和算法设计等方面的能力。

二、 实验内容

（一）问题描述

设计与实现南普陀、胡里山炮台、曾厝垵和厦门大学思明校区主要景点(如上弦场、芙蓉湖等)的旅游咨询系统，为游客提供游程最短的最优决策方案：

（1）设计旅游系统的景点地图；

（2）实现为用户提供从当前景点出发，到终点景点的路线查询服务，打印景点地图等功能。

（二）需求分析

（1）建立一个主菜单，方便用户选择功能，如查看景点地图、查询一个景点到其他景点的游览路线，查询两景点间最短路径等；

（2）系统需要能够创建和输出景点地图，即从文本读取相关景点信息和路径长度信息，创建景点图并输出图的邻接矩阵；

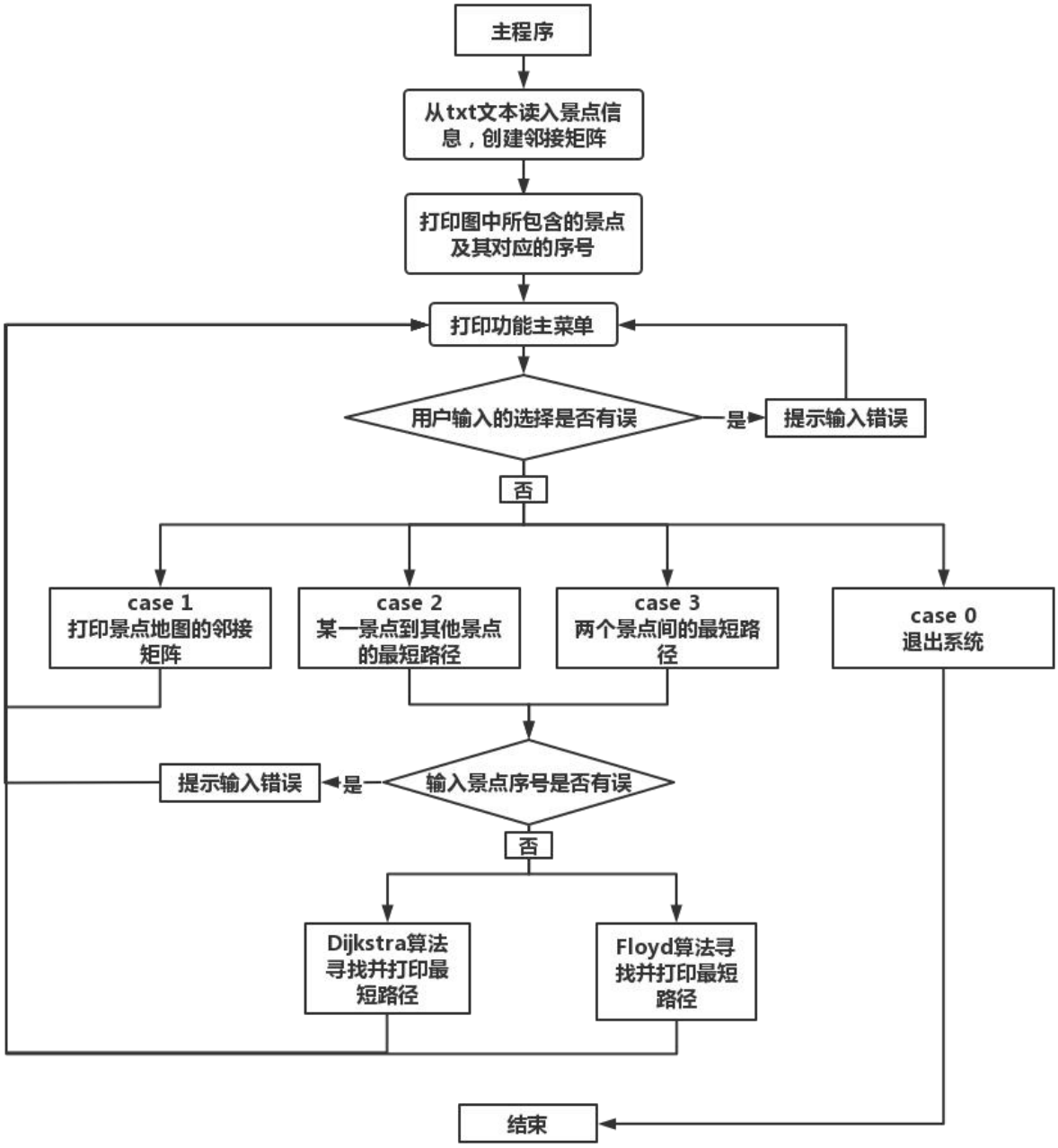
（3）对于已规划好起点但不确定后续游览景点的用户，提供游览线路推荐功能：即用户输入一个起始景点，输出该点到其他所有顶点的最短路径；

（4）对于已规划好起点和终点景点的用户，应提供输出两个景点间的最短路径的功能；同时，也应输出最短路径上的沿途景点，作为参考建议提供给用户。

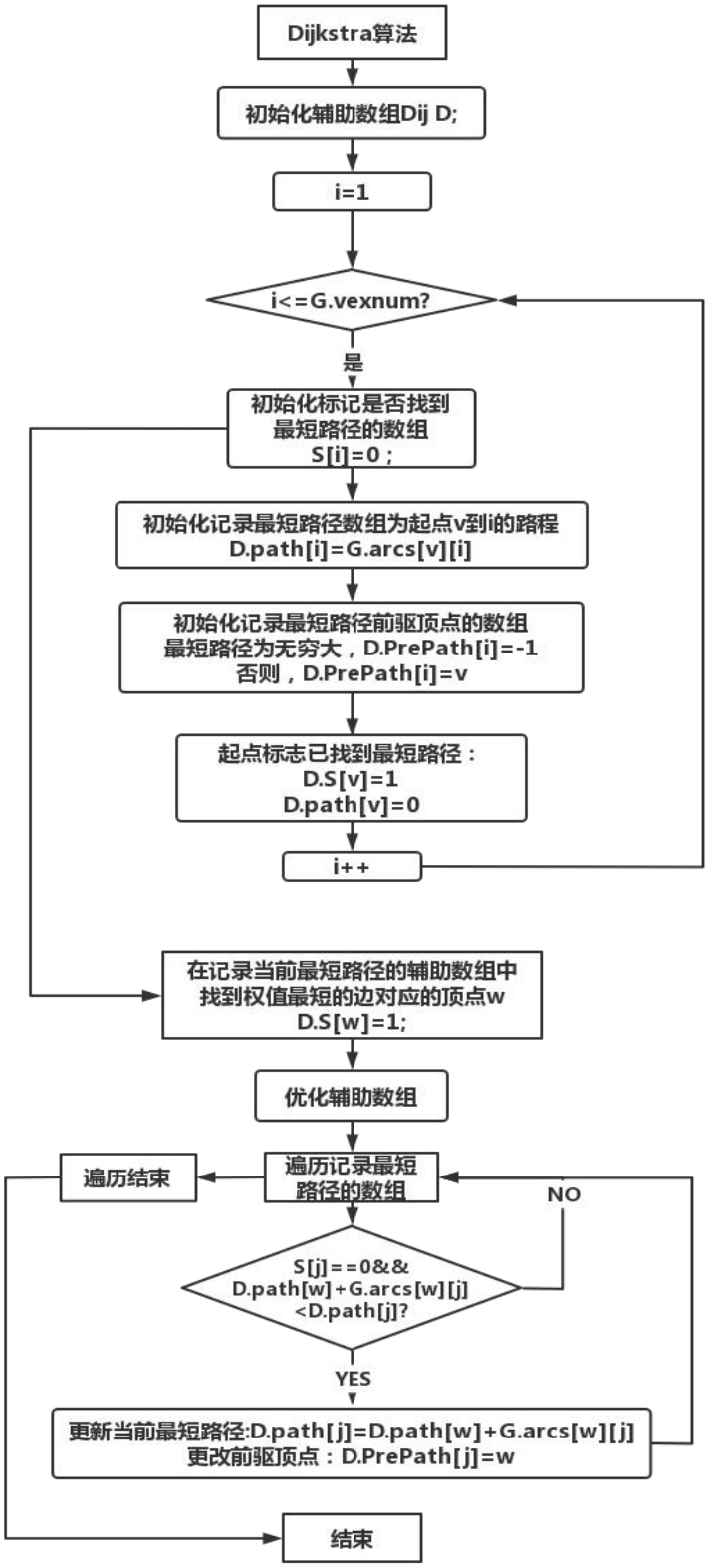
（三）算法设计

主要使用到 Dijkstra 算法和 Floyd 算法，算法主要流程图如下：

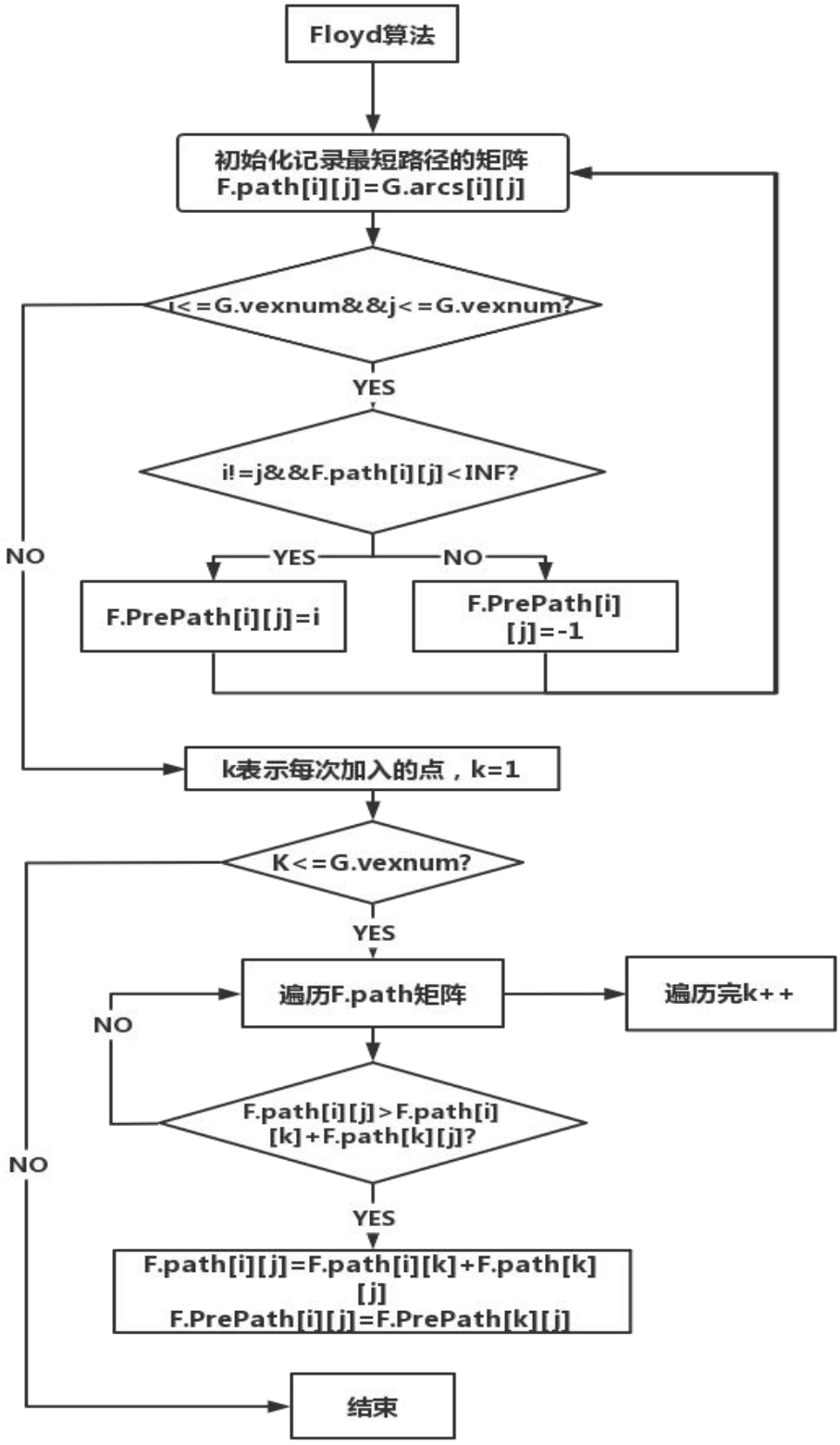
主程序：


主程序
从txt文本读入景点信息，创建邻接矩阵
打印图中所包含的景点及其对应的序号
打印功能主菜单
用户输入的选择是否有误
是
提示输入错误
否
case 1
打印景点地图的邻接矩阵
case 2
某一景点到其他景点的最短路径
case 3
两个景点间的最短路径
case 0
退出系统
提示输入错误
是
输入景点序号是否有误
否
Dijkstra算法寻找并打印最短路径
Floyd算法寻找并打印最短路径
结束

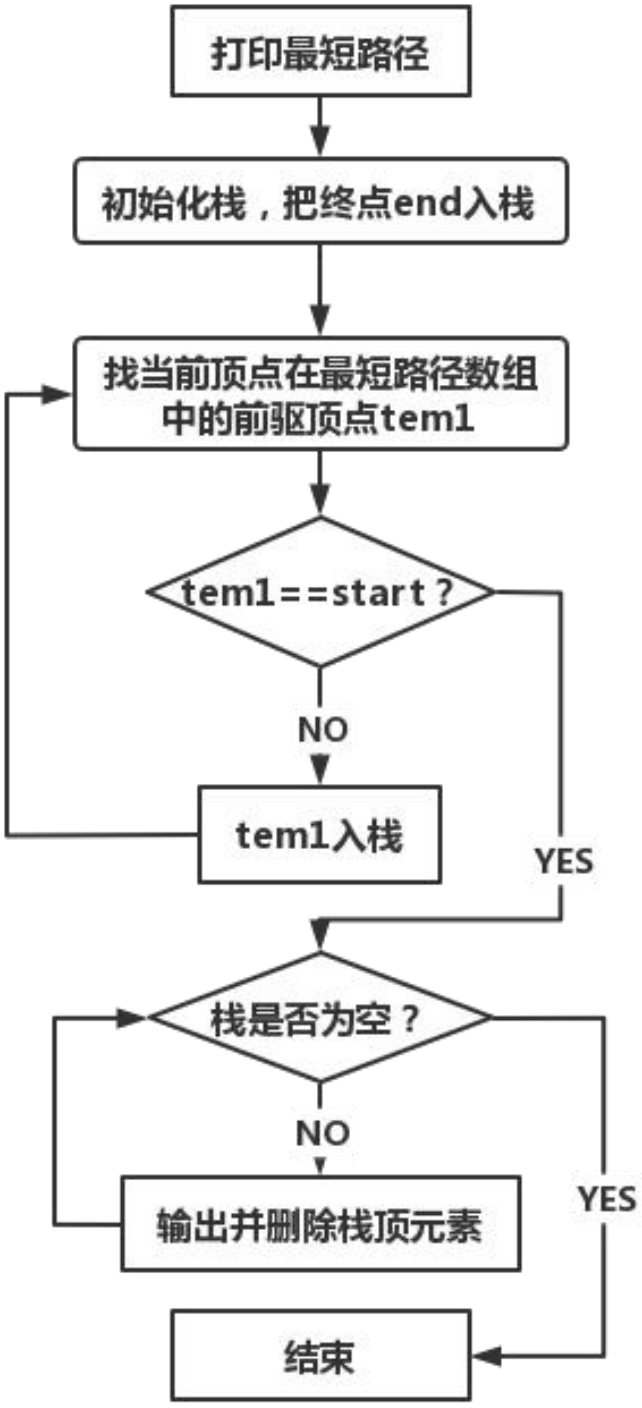
Dijkstra 算法:


Dijkstra算法
初始化辅助数组Dij D;
i=1
i<=G.vexnum?
是
初始化标记是否找到
最短路径的数组
S[i]=0；
初始化记录最短路径数组为起点v到i的路程
D.path[i]=G.arcs[v][i]
初始化记录最短路径前驱顶点的数组
最短路径为无穷大，D.PrePath[i]=-1
否则，D.PrePath[i]=v
起点标志已找到最短路径：
D.S[v]=1
D.path[v]=0
i++
在记录当前最短路径的辅助数组中
找到权值最短的边对应的顶点w
D.S[w]=1;
优化辅助数组
遍历记录最短
路径的数组
遍历结束
NO
S[j]==0&&
D.path[w]+G.arcs[w][j]
<D.path[j]?
YES
更新当前最短路径:D.path[j]=D.path[w]+G.arcs[w][j]
更改前驱顶点：D.PrePath[j]=w
结束

Floyd 算法:


Floyd算法
初始化记录最短路径的矩阵
F.path[i][j]=G.arcs[i][j]
i<=G.vexnum&&j<=G.vexnum?
YES
NO
i!=j&&F.path[i][j]<INF?
YES
NO
F.PrePath[i][j]=i
F.PrePath[i]
[j]=-1
k表示每次加入的点，k=1
K<=G.vexnum?
YES
NO
遍历F.path矩阵
遍历完k++
F.path[i][j]>F.path[i]
[k]+F.path[k][j]?
NO
YES
F.path[i][j]=F.path[i][k]+F.path[k]
[j]
F.PrePath[i][j]=F.PrePath[k][j]
结束

打印最短路径:


打印最短路径
初始化栈，把终点end入栈
找当前顶点在最短路径数组中的前驱顶点tem1
tem1==start？
NO
tem1入栈
YES
栈是否为空？
NO
输出并删除栈顶元素
YES
结束

（四）系统实现

1．主程序：

```
#include "lab7.h"

int main() {
    AMGraph G;
    CreateMap(&G);
    printf("提供路线咨询服务的景点有：\n");
    Print(G);
    printf("\n");
    Menu();
    int choice;
    scanf("%d", &choice);
    while (choice)
    {
        switch (choice)
        {
        case 1:
            PrintMap(G);
            break;
        case 2:
            Print(G);
            printf("\n请输入您想查询的景点对应的序号：\n");
            int x;
            scanf("%d", &x);
            if (x<1 || x>G.vexnum)
            {
                printf("输入错误！\n");
            }
            else {
                Dijkstra(G, x);
                Print_Dij(G, x);
            }
            break;
        case 3:
            Print(G);
            printf("\n请分别输入起点和终点的对应序号：\n");
            int x1, x2;
            scanf("%d %d", &x1, &x2);
            if (x1<1 || x1>G.vexnum || x2<1 || x2>G.vexnum||x1==x2)
            {
                printf("输入错误！！\n");
            }
            else
            {
                floyd(G);
```

```
45                  Print_Floyd(G, x1, x2);
46              }
47              break;
48          default:
49              printf("输入错误，请重新输入！\n");
50              break;
51          }
52          Menu();
53          scanf("%d", &choice);
54      }
55      printf("\n望您满意！\n");
56      printf("正在退出...\n");
57      Sleep(2000);
58      return 0;
59  }
```

程序执行如下：

```
提供路线咨询服务的景点有:
1  - 南普陀寺          2  - 胡里山炮台          3  - 白城沙滩          4  - 曾厝垵
5  - 厦门大学上弦场    6  - 厦门大学鲁迅纪念馆  7  - 厦门大学芙蓉隧道  8  - 厦门大学芙蓉湖
9  - 厦门大学思源谷    10 - 厦门大学人类博物馆

查询服务如下:
*************************************************
*     1-查看景点地图                            *
*     2-查询某个景点到其他景点的最短路线        *
*     3-查询两个景点间的最短路线                *
*     0-退出系统                                *
*************************************************

请输入您的选择:
```

2．创建图以及打印图的邻接矩阵：

```
//定义需要用的数据结构
//邻接矩阵
typedef struct
{
    char vexs[MaxNum + 1][NameLength];//景点名称表
    double arcs[MaxNum + 1][MaxNum + 1];//邻接矩阵
    int vexnum , arcnum;
}AMGraph;
```

```
106     //求景点对应的序号
107     int Locate(AMGraph G, char* name)
108     {
109         for (int i = 1; i <= G.vexnum; i++)
110         {
111             if (strcmp(G.vexs[i], name) == 0)
112             {
113                 return i;
114             }
115         }
116     }
117
118     //创建邻接矩阵
119     void CreateMap(AMGraph* G)
120     {
121         FILE* fp = fopen("sights.txt", "r");
122         fscanf(fp, "%d", &G->vexnum);
123         for (int i = 1; i <= G->vexnum; i++)
124         {
125             fscanf(fp, "%s", G->vexs[i]);
126         }
127         fclose(fp);
128
129         //初始化矩阵
130         for (int i = 0; i <= G->vexnum; i++)
131         {
132             for (int j = 0; j <= G->vexnum; j++)
133             {
134                 G->arcs[i][j] = INF;
135             }
136         }
138     fp = fopen("PathDistance.txt", "r");
139     int n = 0;//记录边数
140     char name1[30], name2[30];
141     double distance;
142     while (fscanf(fp, "%s%s%lf", name1, name2, &distance) != EOF)
143     {
144         n++;
145         int i = Locate(*G, name1);
146         int j = Locate(*G, name2);
147         G->arcs[i][j] = distance;
148         G->arcs[j][i] = distance;
149     }
150     G->arcnum = n;//当前边数
151 }
```

```
153     //打印邻接矩阵
154   void PrintMap(AMGraph G)
155   {
156       printf("\n景点及其对应序号为：\n");
157       Print(G);
158       printf("\n景点地图为：\n");
159       printf("序号  ");
160       for (int i = 1; i <= G.vexnum; i++) {
161           printf("%-4d", i);
162       }
163       printf("\n");
164       for (int i = 1; i <= G.vexnum; i++)
165       {
166           printf("%-5d", i);
167           for (int j = 1; j <= G.vexnum; j++)
168           {
169               if (G.arcs[i][j] == INF) {
170                   printf("%-5c", '*');
171                   continue;
172               }
173               printf("%-5.1lf", G.arcs[i][j]);
174           }
175           printf("\n");
176       }
177       printf("*号表示两景点没有直达路径.\n");
178   }
```

sights.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
10
南普陀寺
胡里山炮台
白城沙滩
曾厝垵
厦门大学上弦场
厦门大学鲁迅纪念馆
厦门大学芙蓉隧道
厦门大学芙蓉湖
厦门大学思源谷
厦门大学人类博物馆
```

第 1 行，第 1 列

PathDistance.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

南普陀寺	胡里山炮台	3.9
南普陀寺	白城沙滩	4.6
白城沙滩	曾厝垵	3.5
胡里山炮台	曾厝垵	3.2
白城沙滩	厦门大学上弦场	1.2
南普陀寺	厦门大学上弦场	1.1
厦门大学上弦场	厦门大学鲁迅纪念馆	0.4
厦门大学上弦场	厦门大学思源谷	1.1
厦门大学上弦场	厦门大学芙蓉隧道	1.4
厦门大学鲁迅纪念馆	厦门大学芙蓉湖	0.3
厦门大学鲁迅纪念馆	厦门大学人类博物馆	0.3
厦门大学芙蓉湖	厦门大学思源谷	0.9
厦门大学芙蓉湖	厦门大学芙蓉隧道	1.2
厦门大学芙蓉湖	厦门大学上弦场	0.7

程序执行如下：

```
请输入您的选择：
1

景点及其对应序号为：
1  - 南普陀寺            2  - 胡里山炮台          3  - 白城沙滩            4  - 曾厝垵
5  - 厦门大学上弦场      6  - 厦门大学鲁迅纪念馆  7  - 厦门大学芙蓉隧道    8  - 厦门大学芙蓉湖
9  - 厦门大学思源谷      10 - 厦门大学人类博物馆
景点地图为：
序号  1   2   3   4   5   6   7   8   9   10
1     *   3.9 4.6 *   1.1 *   *   *   *   *
2     3.9 *   *   3.2 *   *   *   *   *   *
3     4.6 *   *   3.5 1.2 *   *   *   *   *
4     *   3.2 3.5 *   *   *   *   *   *   *
5     1.1 *   1.2 *   *   0.4 1.4 0.7 1.1 *
6     *   *   *   *   0.4 *   *   0.3 *   0.3
7     *   *   *   *   1.4 *   *   1.2 *   *
8     *   *   *   *   0.7 0.3 1.2 *   0.9 *
9     *   *   *   *   1.1 *   *   0.9 *   *
10    *   *   *   *   *   0.3 *   *   *   *
*号表示两景点没有直达路径.
```

3．Dijkstra 算法求一个景点到其他所有顶点的最短路径：

```
77      //Dijkstra算法
78      typedef struct
79      {
80          int S[MaxNum + 1];              //标记是否已求出最短路径
81          double Path[MaxNum + 1];    //记录到i的最短路径
82          int PrePath[MaxNum + 1];        //记录前驱顶点
83      }Dij;
```

```
Dij D;
void Dijkstra(AMGraph G, int v)
{
    //初始化
    for (int i = 1; i <= G.vexnum; i++)
    {
        D.S[i] = 0;
        D.Path[i] = G.arcs[v][i];
        if (D.Path[i] < INF)
            D.PrePath[i] = v;
        else
            D.PrePath[i] = -1;//不存在边vi，i的前驱顶点置为-1
    }
    D.S[v] = 1;
    D.Path[v] = 0;

    for (int i = 2; i <= G.vexnum; i++)
    {
        //找到权值最短的边对应的顶点w
        double min = INF;
        int w;
        for (int j = 1; j <= G.vexnum; j++)
        {
            if (!D.S[j] && D.Path[j] < min)
            {
                min = D.Path[j];
                w = j;
            }
        }
        D.S[w] = 1;
        //优化
        for (int j = 1; j <= G.vexnum; j++)
        {
            if (!D.S[j] && D.Path[w] + G.arcs[w][j] < D.Path[j])
            {
                D.Path[j] = D.Path[w] + G.arcs[w][j];
                D.PrePath[j] = w;
            }
        }
    }
}

void Print_Dij(AMGraph G, int start) {
    printf("从%s出发，到其他景点的最短路径为：\n", G.vexs[start]);
    for (int end = 1; end <= G.vexnum; end++)
    {
        if (end == start)
        {
            continue;
        }
```

```
231             Stack S;
232             InitStack(&S);
233             Push(&S, end);
234             //从end找最短路径的前驱节点，若不是start则进栈
235             int tem1, tem2;
236             for (tem1 = D.PrePath[end]; tem1 != start; tem1 = D.PrePath[tem2])
237             {
238                 Push(&S, tem1);
239                 tem2 = tem1;
240             }
241             printf("%s", G.vexs[start]);
242             tem1 = start;//tem1记录前驱顶点
243             double sum = 0;
244             while (!StackIsEmpty(S))
245             {
246                 Pop(&S, &tem2);
247                 printf("-%.1lfkm->%s", G.arcs[tem1][tem2], G.vexs[tem2]);
248                 sum += G.arcs[tem1][tem2];
249                 tem1 = tem2;
250             }
251             printf("\n总路程为%.1lfkm.\n\n", sum);
252         }
253     }
```

程序执行如下：

```
请输入您的选择：
2
1  - 南普陀寺          2  - 胡里山炮台          3  - 白城沙滩          4  - 曾厝垵
5  - 厦门大学上弦场      6  - 厦门大学鲁迅纪念馆  7  - 厦门大学芙蓉隧道    8  - 厦门大学芙蓉湖
9  - 厦门大学思源谷      10 - 厦门大学人类博物馆
请输入您想查询的景点对应的序号：
3
从白城沙滩出发，到其他景点的最短路径为：
白城沙滩-1.2km->厦门大学上弦场-1.1km->南普陀寺
总路程为2.3km.

白城沙滩-1.2km->厦门大学上弦场-1.1km->南普陀寺-3.9km->胡里山炮台
总路程为6.2km.

白城沙滩-3.5km->曾厝垵
总路程为3.5km.

白城沙滩-1.2km->厦门大学上弦场
总路程为1.2km.

白城沙滩-1.2km->厦门大学上弦场-0.4km->厦门大学鲁迅纪念馆
总路程为1.6km.

白城沙滩-1.2km->厦门大学上弦场-1.4km->厦门大学芙蓉隧道
总路程为2.6km.

白城沙滩-1.2km->厦门大学上弦场-0.7km->厦门大学芙蓉湖
总路程为1.9km.
```

```
白城沙滩-1.2km->厦门大学上弦场-1.1km->厦门大学思源谷
总路程为2.3km.

白城沙滩-1.2km->厦门大学上弦场-0.4km->厦门大学鲁迅纪念馆-0.3km->厦门大学人类博物馆
总路程为1.9km.
```

4 .Floyd 算法求两点间最短路径：

```
259         //初始化
260         for (int i = 1; i <= G.vexnum; i++)
261         {
262             for (int j = 1; j <= G.vexnum; j++)
263             {
264                 if (i == j)
265                 {
266                     F.Path[i][j] = 0;
267                 }
268                 else
269                 {
270                     F.Path[i][j] = G.arcs[i][j];
271                 }
272                 if (F.Path[i][j] < INF && i != j)
273                 {
274                     F.PrePath[i][j] = i;
275                 }
276                 else
277                 {
278                     F.PrePath[i][j] = -1;
279                 }
280             }
281         }
282         //k表示每次加的顶点
283         for (int k = 1; k <= G.vexnum; k++)
284         {
285             for (int i = 1; i <= G.vexnum; i++)
286             {
287                 for (int j = 1; j <= G.vexnum; j++)
288                 {
289                     if (F.Path[i][j] > F.Path[i][k] + F.Path[k][j])
290                     {
291                         F.Path[i][j] = F.Path[i][k] + F.Path[k][j];
292                         F.PrePath[i][j] = F.PrePath[k][j];
293                     }
294                 }
295             }
296         }
297     }
298
299     void Print_Floyd(AMGraph G, int start, int end)
300     {
301         printf("%s到%s的最短路径为：\n\n", G.vexs[start], G.vexs[end]);
302         Stack S;
303         InitStack(&S);
304         Push(&S, end);
305         int tem1, tem2;
306         for (tem1 = F.PrePath[start][end]; tem1 != start; tem1 = F.PrePath[start][tem2])
307         {
308             Push(&S, tem1);
309             tem2 = tem1;
310         }
311         printf("%s", G.vexs[start]);
312         tem1 = start;//tem1记录前驱顶点
313         double sum = 0;
314         while (!StackIsEmpty(S)) {
315             Pop(&S, &tem2);
316             printf("-%.1lfkm->%s", G.arcs[tem1][tem2], G.vexs[tem2]);
317             sum += G.arcs[tem1][tem2];
318             tem1 = tem2;
319         }
320         printf("\n总路程为%.1lfkm.\n\n", sum);
321     }
```

程序执行如下：

```
请输入您的选择：
3
1  - 南普陀寺          2  - 胡里山炮台          3  - 白城沙滩          4  - 曾厝垵
5  - 厦门大学上弦场    6  - 厦门大学鲁迅纪念馆  7  - 厦门大学芙蓉隧道  8  - 厦门大学芙蓉湖
9  - 厦门大学思源谷    10 - 厦门大学人类博物馆
请分别输入起点和终点的对应序号：
2 8
胡里山炮台到厦门大学芙蓉湖的最短路径为：

胡里山炮台-3.9km->南普陀寺-1.1km->厦门大学上弦场-0.7km->厦门大学芙蓉湖
总路程为5.7km.
```

（五）测试分析

测试内容	预期结果	测试结果
对用户输入的错误服务选项作出应对	提示输入错误并重新进入选择界面	基本通过
对用户输入的错误景点序号作出应对	提示输入有误，并重新进入选择服务界面。	基本通过

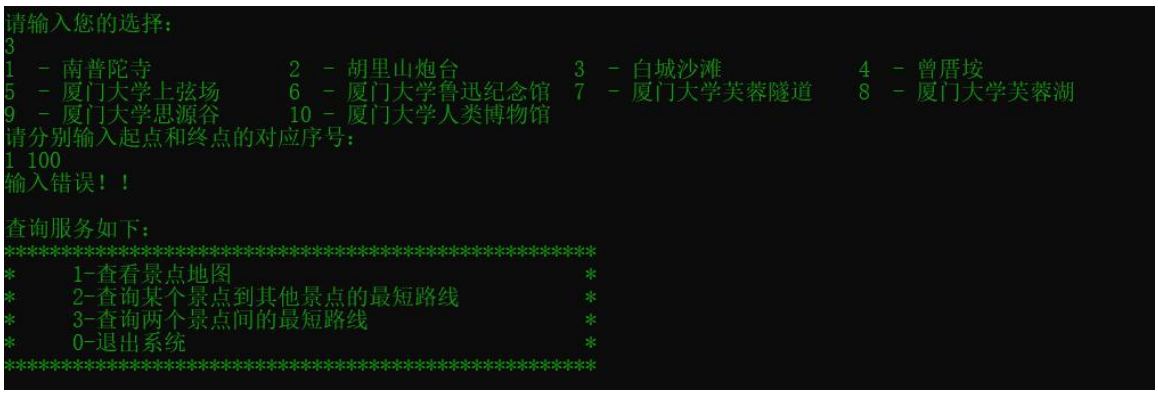
程序执行如下图：

```
请输入您的选择：
4
输入错误，请重新输入！

查询服务如下：
**********************************************************
*       1-查看景点地图                                   *
*       2-查询某个景点到其他景点的最短路线               *
*       3-查询两个景点间的最短路线                       *
*       0-退出系统                                       *
**********************************************************
```

```
请输入您的选择：
2
1  - 南普陀寺          2  - 胡里山炮台          3  - 白城沙滩          4  - 曾厝垵
5  - 厦门大学上弦场    6  - 厦门大学鲁迅纪念馆  7  - 厦门大学芙蓉隧道  8  - 厦门大学芙蓉湖
9  - 厦门大学思源谷    10 - 厦门大学人类博物馆
请输入您想查询的景点对应的序号：
15
输入错误！

查询服务如下：
**********************************************************
*       1-查看景点地图                                   *
*       2-查询某个景点到其他景点的最短路线               *
*       3-查询两个景点间的最短路线                       *
*       0-退出系统                                       *
**********************************************************
```



三、 实验小结（即总结本次实验所得到的经验与启发等）：

在本次实验中，我运用了 Dijkstra 算法和 Floyd 算法，并结合读入文本操作、创建并打印邻接矩阵、顺序栈的操作等，简单实现了一个旅游咨询系统，为游客提供游程最短的最优决策方案。这次较为综合的程序设计实验让我明白编写算法只是整个程序设计过程最基础的一部分。通过这次实验，我加深了对程序、算法设计的理解。