数据结构与算法 第七次实验

景区旅游信息咨询系统（厦门市）

学号：22920212204392 姓名：黄勖

一、 实验目的

1. 了解图的基础实现方法与原理，理解图的基本操作的代码编写方式
2. 学会灵活按照实际应用的存储内容要求编写图的存储结构
3. 在图的基础上进而理解编写各种最短路的基础实现方法，理解最短路决策的基本操作的代码编写方式
4. 通过实验探索不同的图的相似点与区别，发现在操作实现上的异同

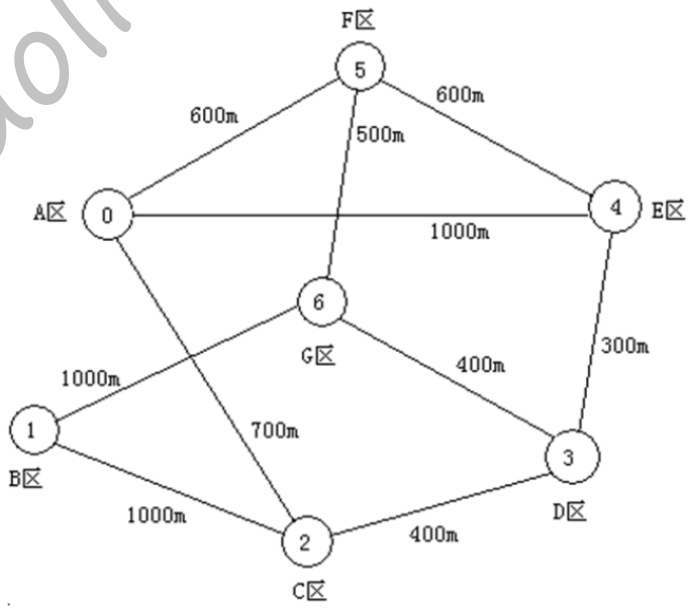
二、 问题描述

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。

三、 需求分析

在这个过程对问题进行抽象：在一张给定的连通的的每条边含有正权值的无向图中，选择起点与终点，要求寻找出包含这些点且权值和最小的通路。

如下图，对于这样的一个包含景点与道路信息的图，需要对其进行处理找出给定景点的最短路径。



四、 算法设计

这个问题也是对图的遍历的各种操作的实现。对于寻找导游路线，我们可以利用 DFS (深度优先搜索) 。这是一种图的遍历算法，其主要思想是沿着边的方向遍历图的每一条路径，直到遇到岔路或者搜索完整张图。

给定一张图，DFS 的过程如下：

[1] 初始化：从图中任意一个节点开始遍历，并将该节点标记为已遍历。

[2] 寻找下一个节点：寻找当前节点的所有相邻节点，如果某个相邻节点未被遍历，则递归地从该节点开始遍历。

[3] 回溯：当前节点的所有相邻节点均已被遍历，回到上一个节点继续遍历。

对于寻找最短路径，可以利用 Dijkstra 算法。Dijkstra 算法是用于解决最短路径问题的算法，主要应用于有向无环图（DAG）和有权图。

基本流程如下：

[1] 初始化：将起点设为已确定的最短路径，其余顶点的路径长度设为正无穷，并将起点加入已确定最短路径的顶点集合。

[2] 找出未确定最短路径的顶点中距离起点最近的顶点，将其加入已确定最短路径的顶点集合，并更新其它顶点的路径长度。

[3] 重复步骤 2，直到所有顶点都加入到已确定最短路径的顶点集合中。

[4] 输出最短路径。

更新顶点的路径长度的方法如下：

[1] 对于已确定最短路径的顶点，枚举与其相邻的顶点。

[2] 如果相邻顶点的路径长度大于已确定最短路径的顶点到起点的路径长度加上两点之间的边的权值，则将相邻顶点的路径长度更新为更小的值。

此外，我还为旅游咨询系统添加了展示往返路线规划图的功能，由于设置交通路线需要保证所有路线总长度最小来降低成本，因此我使用了最小生成树 Prim 算法。Prim 算法是用于解决最小生成树问题的一种算法。最小生成树是指一张图的所有顶点的子集，这个子集构成的树，其边权之和最小。

基本流程如下：

[1]初始化:将图中的任意一个顶点作为起点,将其加入最小生成树的顶点集合。

[2]找出未加入最小生成树的顶点中距离已加入最小生成树的顶点最近的顶点，并将其加入最小生成树的顶点集合。

[3]重复步骤 2，直到所有顶点都加入到最小生成树的顶点集合中。

Prim 算法的时间复杂度是 O(n^2)，空间复杂度是 O(n)。

五、 系统实现

（1）Program Structure

node 和 Tornode 结构体用于存储景点的信息和两个景点之间的边的信息。
MatGraph 结构体用于表示使用邻接矩阵表示的图，邻接矩阵是一个二维数组，其中 map[i][j]元素表示从第 i 个景点到第 j 个景点的边的权重。
BSTNode 结构体用于表示二叉搜索树中的节点，二叉搜索树是一种用于高效存储和搜索数据的数据结构。
ArcNode 结构体用于表示邻接表中的边，邻接表是表示图的另一种方法，其中每个顶点都有一个指向其相邻顶点的列表。
VNode 结构体用于表示邻接表中的头结点。
AdjGraph 结构体用于表示使用邻接表表示的图。
CreatMatGragh 函数用于使用邻接矩阵创建图的表示。它接受图中的顶点数和边数以及边的数组作为输入。
cmp 函数作为按照其 number 字段对 Node 结构体数组进行排序的比较器。
digit 函数用于计算数字的位数
transtr 函数将整数转换为字符串。
Read 函数从文档中读取数据并将其存储在 Node 结构体数组中。
MAtToList 函数将邻接矩阵存储的图转化为邻接表存储的图。
DispAdj 函数输出邻接表中的信息。
SearchBST 函数在二叉搜索树中查找编号为 k 的节点。
TopSort 函数为拓扑排序，用于确定有向图的一种排列，使得对于图中的每一条有向边(u,v)，均有 u 在 v 之前排列。
DFS 函数对邻接表进行深度优先搜索。
Display 函数显示所有景点的名字。
CreadMat 函数用于创建邻接矩阵。
InsertBST 函数在二叉排序树中插入节点。
CreatBST 函数用于创建二叉树。
Dijkstra 函数使用 Dijkstra 算法在图中查找两个顶点之间的最短路径。
Prim 函数用于生成最小生成树。

（2）Function Implementation

```
458 void PageInfor() {
459     system("color 5E");
460     cout<<"\t\t\t   ********************************************"<<endl;
461     cout<<"\t\t\t   *                                          *"<<endl;
462     cout<<"\t\t\t   *                                          *"<<endl;
463     cout<<"\t\t\t   *\t       景区旅游信息咨询系统             *"<<endl;
464     cout<<"\t\t\t   *                 厦门市景区               *"<<endl;
465     cout<<"\t\t\t   *                                          *"<<endl;
466     cout<<"\t\t\t   *                                          *"<<endl;
467     cout<<"\t\t\t   *\t"<<" ☆ "<<" 1  创建景区景点分布图       ☆     *"<<endl;
468     cout<<"\t\t\t   *                                          *"<<endl;
469     cout<<"\t\t\t   *\t"<<" ☆ "<<" 2  输出景区景点分布图       ☆     *"<<endl;
470     cout<<"\t\t\t   *                                          *"<<endl;
471     cout<<"\t\t\t   *\t"<<" ☆ "<<" 3  输出导游路线             ☆     *"<<endl;
472     cout<<"\t\t\t   *                                          *"<<endl;
473     cout<<"\t\t\t   *\t"<<" ☆ "<<" 4  输出最佳导游路线         ☆     *"<<endl;
474     cout<<"\t\t\t   *                                          *"<<endl;
475     cout<<"\t\t\t   *\t"<<" ☆ "<<" 5  输出最短路径             ☆     *"<<endl;
476     cout<<"\t\t\t   *                                          *"<<endl;
477     cout<<"\t\t\t   *\t"<<" ☆ "<<" 6  输出往返路线规划图       ☆     *"<<endl;
478     cout<<"\t\t\t   *                                          *"<<endl;
479     cout<<"\t\t\t   *\t"<<" ☆ "<<" 7  退出系统                 ☆     *"<<endl;
480     cout<<"\t\t\t   *                                          *"<<endl;
481     cout<<"\t\t\t   *                                          *"<<endl;
482     cout<<"\t\t\t   ********************************************"<<endl;
483
484     cout<<"功能选择 >>  :    ";
485
486 }
```

PageInfor 是主要的页面展示函数，展示了我设计的具体功能，注意每一个功能都依托于前一个功能。

1 - 创建景区分布图实现了建图初始化的需求。

2 - 输出景区分布图会把地图景点名称与长度信息输出。

3 - 会输出所有可能的旅游路线。（DFS）

主要流程如下：

1. 先把所有景点创建成二叉排序树。
2. 中序遍历二叉树，输出所有景点的名称。
3. 把邻接矩阵图转化成邻接表表示的图。
4. 循环遍历所有的景点，深度优先遍历图，把每个景点的编号存储到一个数组 TourMap 中。
5. 每遍历完一个景点，就输出一个方案，把每个景点的名称输出出来。

4 - 构建城市旅游景点的最佳导游路线。

主要流程如下：

1. 声明一个结构体数组 Gr，用来存储边的信息，包括边的起点、终点、权值。
2. 循环遍历景点地图 TourMap，把每一条边的信息存储到 Gr 数组中。
3. 声明一个图的数组 GT，每一个图对应一种可能的导游路线。然后用每条边的信息创建邻接矩阵表示的无向图。
4. 再次循环遍历每个图，计算每个图的边数。

5. 找到边数等于景点数减一的图，这种图就是回路图，也就是最佳导游路线。

6. 打印出这种图的每一个景点的名称。

5 - 这里是实验实现的主要内容。Dijkstra 函数是一个用于在图中查找两个顶点之间的最短路径的函数。它接受当前顶点，图的表示，二叉搜索树和终点顶点作为输入。

函数开始时，它设置了图中的顶点数和边数，并使用 memset 函数初始化了 vis 和 Path 数组。

然后，它对每个景点进行迭代，将其距离初始化为从当前顶点到它的距离。如果两个顶点之间存在一条边，则将其设置为当前顶点的前驱节点，否则将其设置为零。

然后，将当前顶点的访问标志设置为 1，并将其前驱节点设置为零。

然后，对于每个顶点，它将找到一个未访问的顶点，将其设置为当前顶点，并将其访问标志设置为 1。

然后，它将遍历所有未访问的顶点，并使用 Dijkstra 算法更新它们的距离。

最后，它使用 SearchBST 函数在二叉搜索树中搜索具有给定键的节点，并打印出两个景点之间的最短距离以及它们之间路径。

然后，它使用一个循环，查找前驱节点并打印出两个景点之间的路径。如果找到了前驱节点，则将其打印出来，并将其设置为下一个要搜索的节点。如果没有找到前驱节点，则打印换行符并退出程序。

6 - 输出往返路线规划图。这里主要实现了 Prim 生成最小生成树算法。

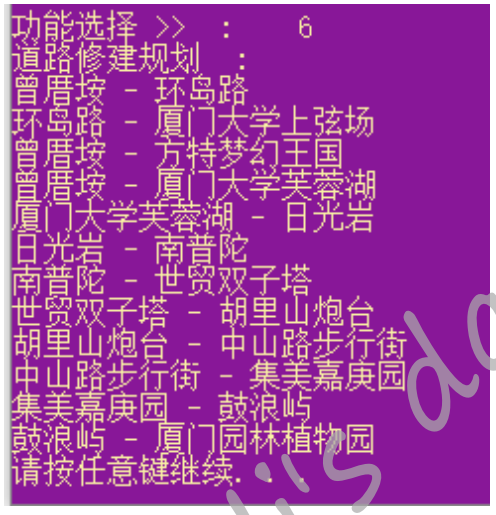
六、 测试分析

经过附在文件夹中的代码的测试得到的结果如下，经检验是正确的。

```
E:\大二上\数据结构与算法\实验\实验7_22920212204392_黄鼬\旅游咨询系统\deque.exe
功能选择 >>  :    2
☆所有景点信息 ☆
 景点名字          编号
南普陀   : 0008
胡里山炮台   : 0009
曾厝垵   : 0001
厦门大学上弦场   : 0002
厦门大学芙蓉湖   : 0005
日光岩   : 0006
鼓浪屿   : 0010
中山路步行街   : 0013
厦门园林植物园   : 0011
环岛路   : 0003
方特梦幻王国   : 0004
集美嘉庚园   : 0012
世贸双子塔   : 0007
 景区地图
   1   2   3   4   5   6   7   8   9   10   11   12   13
1   0  ∞  150  200  200  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞
2   ∞  0  150  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞
3   150  150  0  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞
4   200  ∞  ∞  0  ∞  ∞  ∞  ∞  ∞  500  400  ∞
5   200  ∞  ∞  ∞  0  50  ∞  ∞  ∞  ∞  ∞  ∞  ∞
6   ∞  ∞  ∞  ∞  50  0  ∞  140  ∞  ∞  ∞  ∞  ∞
7   ∞  ∞  ∞  ∞  ∞  ∞  0  120  100  ∞  ∞  ∞  ∞
8   ∞  ∞  ∞  ∞  ∞  140  120  0  ∞  ∞  ∞  ∞  ∞
9   ∞  ∞  ∞  ∞  ∞  ∞  ∞  100  ∞  0  ∞  ∞  ∞  160
10   ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  0  120  100  ∞
11   ∞  ∞  ∞  500  ∞  ∞  ∞  ∞  ∞  120  0  ∞  ∞
12   ∞  ∞  ∞  400  ∞  ∞  ∞  ∞  ∞  100  ∞  0  230
13   ∞  ∞  ∞  ∞  ∞  ∞  ∞  ∞  160  ∞  ∞  230  0

请按任意键继续. . .
```

```
功能选择 >>  :    3
 所有景点如下 :
1    曾厝垵   0001
2    厦门大学上弦场   0002
3    环岛路   0003
4    方特梦幻王国   0004
5    厦门大学芙蓉湖   0005
6    日光岩   0006
7    世贸双子塔   0007
8    南普陀   0008
9    胡里山炮台   0009
10    鼓浪屿   0010
11    厦门园林植物园   0011
12    集美嘉庚园   0012
13    中山路步行街   0013

 所有的导游路线  :

    方案     1   :
  曾厝垵-> 环岛路-> 厦门大学上弦场-> 方特梦幻王国-> 厦门园林植物园-> 鼓浪屿-> 集美嘉庚园-> 中山路步行街-> 胡里山炮台->
世贸双子塔-> 南普陀-> 日光岩-> 厦门大学芙蓉湖->

    方案     2   :
  厦门大学上弦场-> 环岛路-> 曾厝垵-> 方特梦幻王国-> 厦门园林植物园-> 鼓浪屿-> 集美嘉庚园-> 中山路步行街-> 胡里山炮台->
世贸双子塔-> 南普陀-> 日光岩-> 厦门大学芙蓉湖->

    方案     3   :
  环岛路-> 曾厝垵-> 方特梦幻王国-> 厦门园林植物园-> 鼓浪屿-> 集美嘉庚园-> 中山路步行街-> 胡里山炮台-> 世贸双子塔-> 南普
陀-> 日光岩-> 厦门大学芙蓉湖-> 厦门大学上弦场->

    方案     4   :
```

```
功能选择 >>  :    4

 最佳导游路线
厦门大学上弦场 -> 环岛路 -> 曾厝垵 -> 方特梦幻王国 -> 厦门园林植物园 -> 鼓浪屿 -> 集美嘉庚园 -> 中山路步行街 -> 胡里山炮
台 -> 世贸双子塔 -> 南普陀 -> 日光岩 -> 厦门大学芙蓉湖
请按任意键继续. . .
```



七、 项目重现方法

本项目使用 Dev-C++实现。如需使用只需要打开旅游咨询系统文件夹的“deque.exe”文件即可试验使用。

如需重现（编译）可以使用 Dev-C++编辑器打开“deque.dev”文件，里面包含整个项目的配置。如无 Dev-C++也可以用其他编辑器打开“main.cpp”进行编译。

关于图的信息存储在“map.txt”与“edge.txt”中，存储格式打开文本文件后可以很明了的看到。tmp 文件夹中为调试使用的文件，与重现项目无关。

八、 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，我尝试具体运用了图的操作，在实体机的实验中我能够更深刻地理解对这一部分数据结构的执行方式与特点，并且在编写代码的过程中，我通过不断的调试去寻找语句之间的问题和不足，在潜移默化中提高了我的代码编写能力，这是一次完成效果良好的实验！