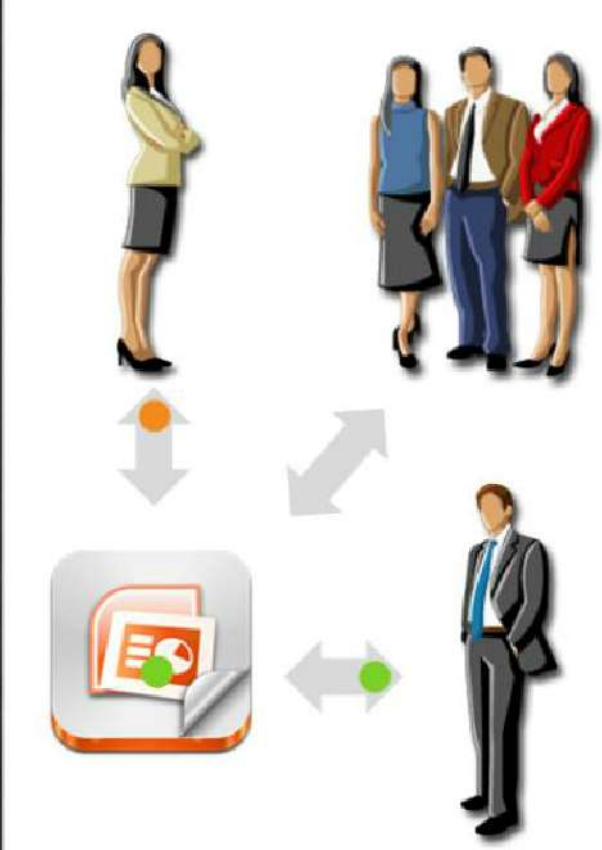
景区信息管理系统

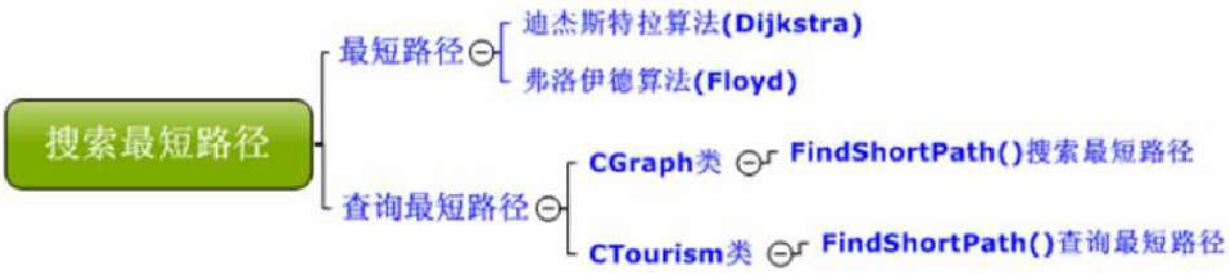


景区信息管理系统

搜索最短路径

主要内容

- 功能简介
- 设计思路
- 技术分析
- 实现
- 小结


搜索最短路径
最短路径
迪杰斯特拉算法(Dijkstra)
弗洛伊德算法(Floyd)
查询最短路径
CGraph类
FindShortPath()搜索最短路径
CTourism类
FindShortPath()查询最短路径

功能简介

在“旅游景点导航”的基础上，为景区信息管理系统增加搜索最短路径功能。

输入：(1) 起点编号int nVexStart；(2) 终点编号int nVexEnd。

处理：搜索两个景点之间的所有路径，找到其中距离最短的路径。

输出：在控制台输出最短路径及路径的总长度。

设计思路

1、算法设计

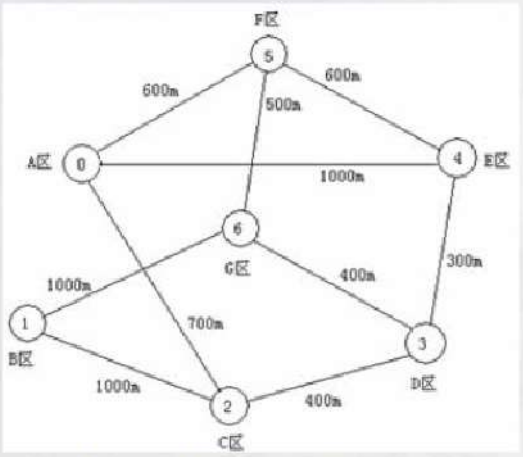
常用的求最短路径的算法有两个：

(1) 迪杰斯特拉(Dijkstra)算法

特点：求得从某个源点到其余各顶点的最短路径

(2) 弗洛伊德(Floyd)算法

特点：求得每一对顶点之间的最短路径



本程序首先使用Dijkstra算法来求得从某个起点到其余各顶点之间的最短路径，然后从中选出我们需要的那条。

对于同一顶点上的不同邻接点，按照其存储顺序进行访问。

2、类的设计

CGraph类

增加成员函数：int FindShortPath(int nVexStart, int nVexEnd, Edge aPath[])
输入：起始景点的编号v1和目的景点的编号v2
输出：最短路径
功能：通过Dijkstra算法求得v1到v2的最短路径

CTourism类

增加成员函数：void FindShortPath(void)
输入：无
输出：无
功能：通过调用m_Graph()函数查询两个景点间的最短路径和距离

迪杰斯特拉(Dijkstra)算法

按路径长度递增的次序产生最短路径。下图是从v1点到其余各顶点之间最短路径的求解过程。

终点	从v1到各终点的D值和最短路径的求解过程					
	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6
v0	不连通	1700m (v1, v2, v0)	1700m (v1, v2, v0)	1700m (v1, v2, v0)	1700m (v1, v2, v0)	
v2	1000m (v1, v2)					
v3	不连通	1400m (v1, v2, v3)	1400m (v1, v2, v3)			
v4	不连通	不连通	不连通	1700m (v1, v2, v3, v4)	1700m (v1, v2, v3, v4)	1700m (v1, v2, v3, v4)
v5	不连通	不连通	1500m (v1, v6, v5)	1500m (v1, v6, v5)		
v6	1000m (v1, v6)	1000m (v1, v6)				
vj	v2	v6	v3	v5	v0	v4
S	{v1, v2}	{v1, v2, v6}	{v1, v2, v6, v3}	{v1, v2, v6, v3, v5}	{v1, v2, v6, v3, v5, v0}	{v1, v2, v6, v3, v5, v0, v4}

如何通过Dijkstra算法得到两个顶点间的最短路径？

如何编程实现搜索最短路径功能？

编程实现

实现

为景区信息管理系统增加搜索最短路径功能。

使用Dijkstra算法求得两个顶点间的最短路径和最短距离，并显示在界面上。

具体实现步骤如下：

步骤一：导入工程

步骤二：搜索最短路径

步骤三：查询最短路径

步骤四：编译和运行。

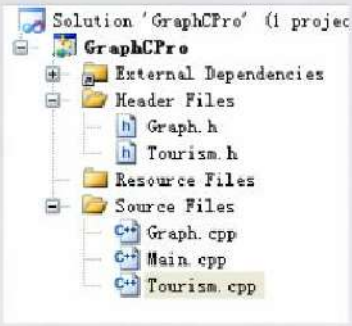
实现：导入工程

1、导入工程

在实现了旅游景点导航的功能之后，要在原有工程的基础上新增搜索最短路径功能。

(1) 打开Microsoft Visual Studio 2010开发工具。

(2) 导入“GraphCPro”工程。



实现：搜索最短路径

2、搜索最短路径

使用Dijkstra算法搜索最短路径，对于同一顶点上的多个邻接点，按照它们的存储顺序来访问。

在CGraph类中添加FindShortPath()方法，通过Dijkstra算法搜索源点到终点之间的最短路径。

```
int CGraph::FindShortPath(int nVexStart, int nVexEnd, Edge aPath[])
{
        // 初始化最短路径
        for(...;...;...){
                if (w是距离nVexStart最近的点 && !aVisited[w]){
                        aVisited[w] = true;
                }
                // 将w作为中间点计算nVexStart到所有顶点的最短距离
                if(有新的最短距离){
                        // 更新最短路径
                }
        }
}
```

实现：查询最短路径

3、查询最短路径

在CTourism类中添加**FindShortPath()**方法，通过调用m_Graph.FindShortPath函数，查询起始景点到目的景点之间的的最短路径和最短距离。

```
void CTourism::FindShortPath(void)
{
        //输入两个景点的编号
        //搜索最短路径
        //得到最短路径和最短距离
}
```

实现：编译和运行

4、编译和运行

在main()函数case 4语句中调用FindShortPath()函数，进入旅游景点导航功能。

编译和运行，得到结果：

小结

1、最短路径

2、Dijkstra算法