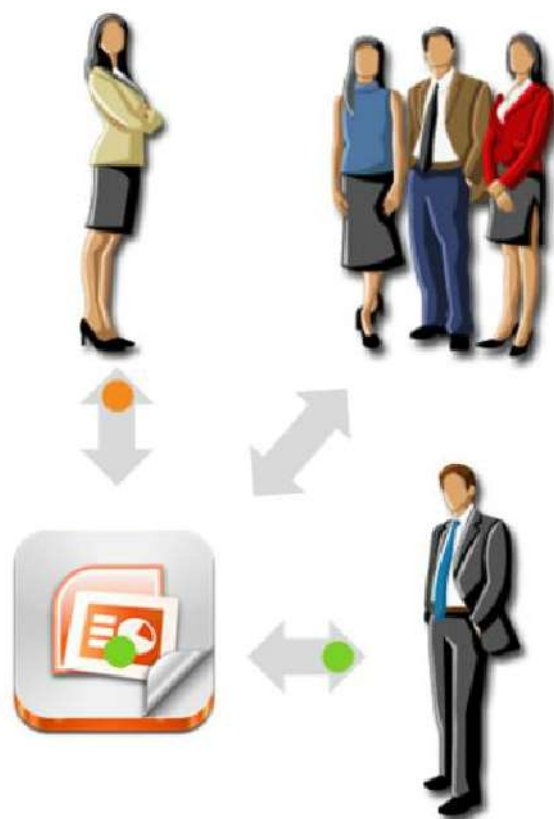


景区信息管理系统



软酷网
www.RuanKo.com

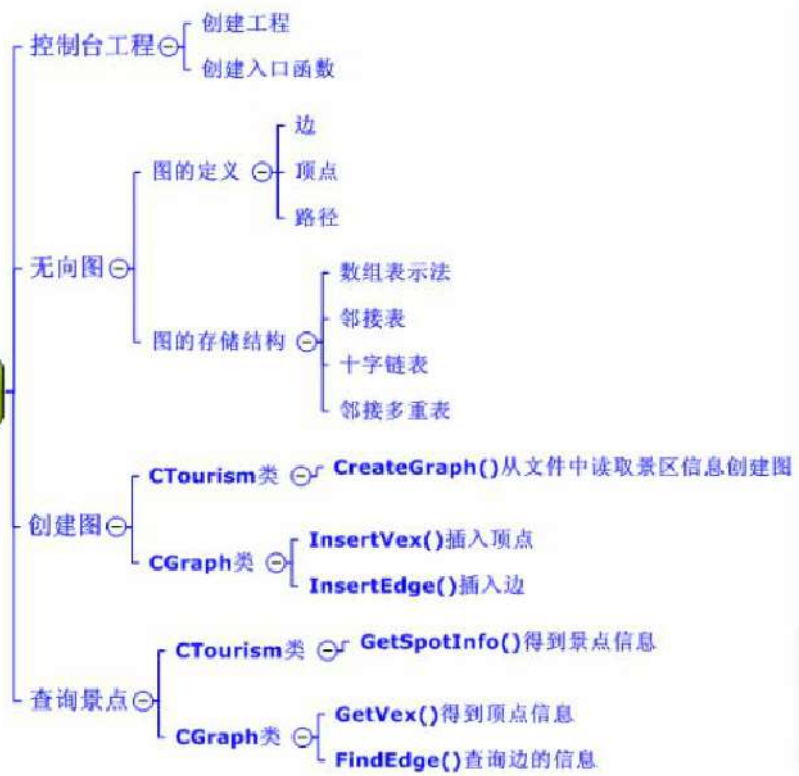
创建图和查询景点

主要内容

- 功能简介
- 设计思路
- 技术分析
- 实现
- 小结

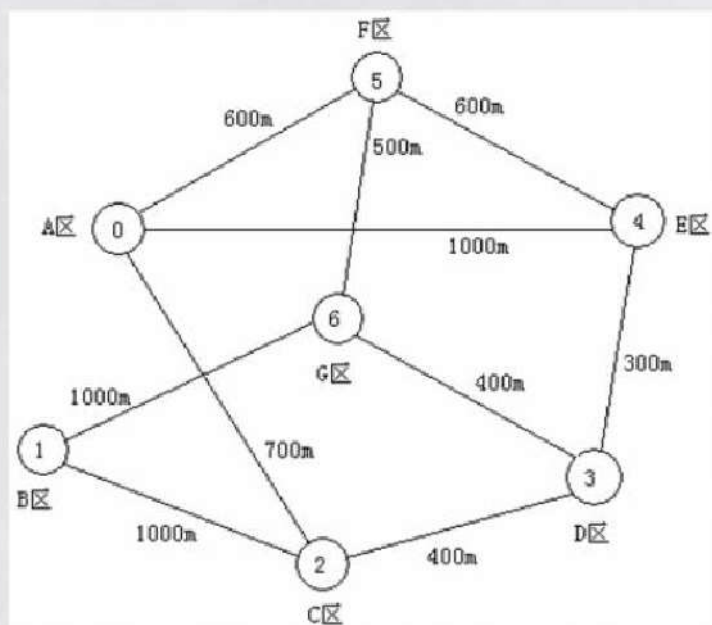


创建图/查询景点



现有一个景区，景区里面有若干个景点，景点之间满足以下条件：

- 1、某些景点之间铺设了道路(连通)
- 2、这些道路都是可以双向行驶的(无向图)
- 3、从任意一个景点出发都可以游览完整个景区(连通图)



景区的数据包含 **景点信息**和景点之间的**道路信息**。

景点信息：景点编号、名字和介绍

道路信息：景点1，景点2、两个景点之间的距离。

若道路信息中没有某两个景点的信息，就表示这两个景点之间没有直接的路径。

编号	名字	介绍
0	A区
1	B区
2	C区
3	D区
4	E区
5	F区
6	G区

景点信息

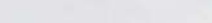



景点1	景点2	距离(m)
A区	C区	700
A区	E区	1000
A区	F区	600
B区	C区	1000
B区	G区	1000
C区	D区	400
D区	E区	300
D区	G区	400
E区	F区	600
F区	G区	500

道路信息

景区数据保存在两个文本文件中。Vex.txt文件用来存储景点信息；Edge.txt文件用来存储道路信息。

Vex.txt文件格式：

景点个数
景点编号
景点名字
景点介绍
景点编号
景点名字
景点介绍
.....



Vex.txt - 记事本

文件(F) 编辑(E) 格式(O)

7

0

A区

风景优美，气候宜人。

1

B区

风景优美，气候宜人。




2

C区

风景优美，气候宜人。

Edge.txt文件格式：

景点1的编号	景点2的编号	道路的长度
景点1的编号	景点2的编号	道路的长度
景点1的编号	景点2的编号	道路的长度
.....		



Edge.txt - 记事本

文件(F) 编辑(E) 格式(O) 通

0	2	700
0	4	1000
0	5	600

1、创建景区景点图

输入

- (1)景点信息(编号、名字和介绍)
- (2)道路信息(两端景点的编号和道路的长度)

处理：系统根据输入的景区信息创建景区景点图。

输出：创建成功，显示图的信息；否则显示失败的信息。

2、查询景点信息

输入：景点编号

处理：系统通过编号查找到相应的景点，得到景点的相关信息

输出：

- (1)景点的名字、介绍
- (2)周边景点名字及距离

使用Microsoft Visual Studio2010开发工具**创建控制台工程GraphCPro**，用C++语言开发景区信息管理系统。

1、程序结构

(1) CGraph类

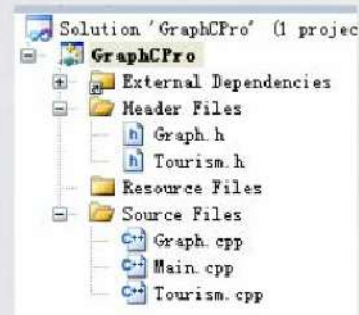
定义图的数据结构，实现图的相关操作。

(2) CTourism类

实现景区信息管理系统的相关功能。

(3) Main.cpp文件

包含程序入口函数int main(void)。



2、界面设计

在int main(void)函数中输出菜单，将系统功能列出来，供用户选择。

```
C:\ F:\GraphCPro.exe
===== 景区信息管理系统 =====
1. 创建景区景点图
2. 查询景点信息
3. 旅游景点导航
4. 搜索最短路径
5. 铺设电路规划
0. 退出
请输入操作编号(0~5): _
```

3、数据结构设计

(1) 图的存储

当保存图结构时，即要保存顶点信息，也要保存边。图可用数组或链表来存储。
数组表示，常用一维数组来保存顶点的集合，使用二维数组来保存边的集合；
链表表示，常用邻接表、十字链表等方式存储图的顶点和边的信息。

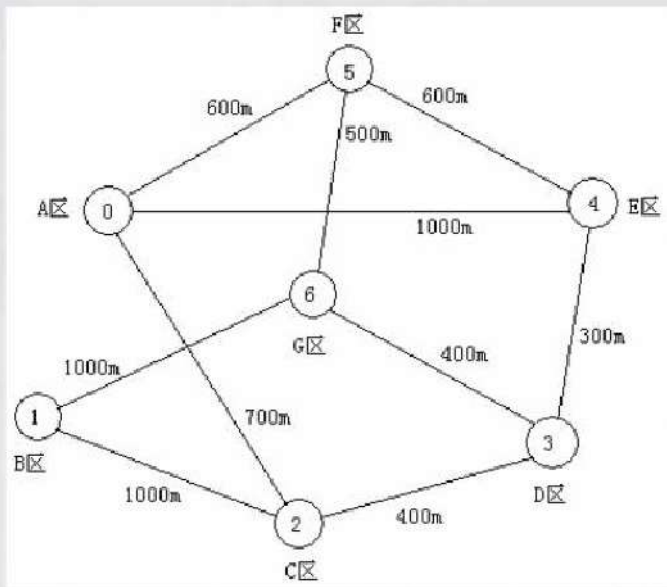
定义一维数组 `Vex m_aVexs[20]` 保存顶点信息，最多允许有20个顶点。

定义二维数组(邻接矩阵) `int m_AdjMatrix[20][20]` 保存边的集合，数组中每个元素的值即为边的权值。

(2)景区景点图

景区的地图可以看做是一个**带权无向图**，使用邻接矩阵来保存。

- (1) 景区中的所有**景点即为图的顶点**
- (2) 当两个景点之间铺设的有道路时，表示两个顶点相连，为一条边
- (3) 两个景点之间的**距离**，即为边的**权值**。**权值为0表示两个顶点不相连。**



	0	1	2	3	4	5	6
0	0	0	700	0	1000	600	0
1	0	0	1000	0	0	0	1000
2	700	1000	0	400	0	0	0
3	0	0	400	0	300	0	400
4	1000	0	0	300	0	600	0
5	600	0	0	0	600	0	500
6	0	1000	0	400	0	500	0

4、类设计

(1) CGraph类

CGraph类为程序的数据结构类，用于存储景区景点图，实现图的相关操作。

① 数据成员，访问权限private

`int m_aAdjMatrix[20][20];` —— 邻接矩阵

`Vex m_aVexs[20];` —— 顶点数组

`int m_nVexNum;` —— 顶点个数

② 成员函数，访问权限public

`void Init(void)`，初始化图结构。

`int InsertVex(Vex sVex)`，将顶点添加到顶点数组中。

`int InsertEdge(Edge sEdge)`，将边保存到邻接矩阵中。

`Vex GetVex(int v)`，查询指定顶点信息。

`int FindEdge(int v, Edge aEdge[])`，查询与指定顶点相连的边。

`int GetVexnum(void)`，获取当前顶点数。

(2)CTourism类

CTourism类为景点旅游信息管理功能类，用于实现系统的主要功能。

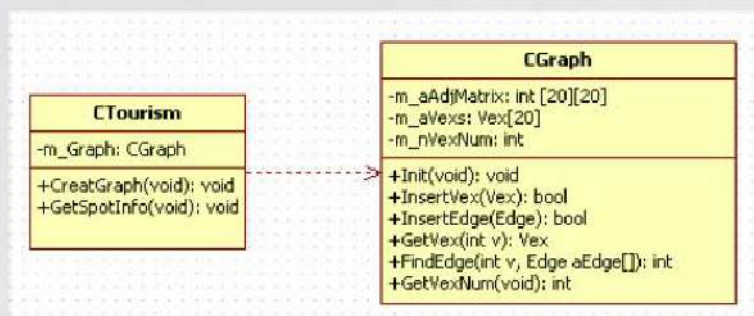
① 数据成员，访问权限private

CGraph m_Graph; —— 景区景点图

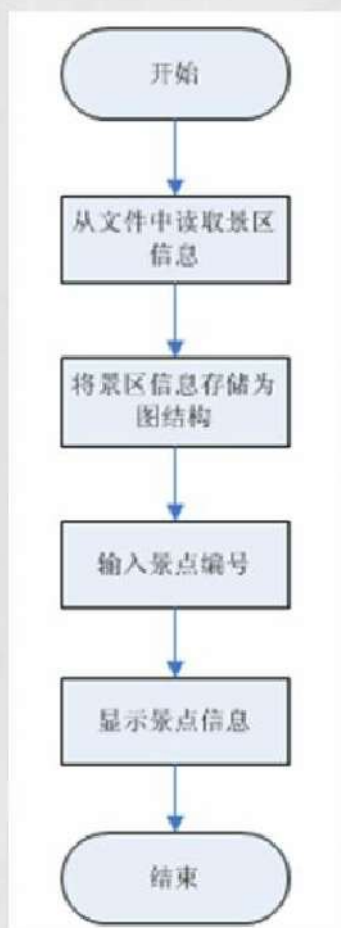
② 成员函数，访问权限public

void CreateGraph(void)，读取文件，创建景区景点图。

void GetSpotInfo(void)，查询指定景点信息，显示到相邻景点的距离。



5、业务流程



1、控制台工程

(1)创建工程

(2)创建入口函数

2、图

(1)图的定义：边、顶点、路径

(2)图的存储结构

(3)图的创建

3、文件操作

(1)fopen()

(2)fgets()

(3)fclose()

- 1、如何将景区信息表示为图的结构？
- 2、如何保存图的顶点、边和权值？

编程实现

使用Microsoft Visual Studio2010开发工具**创建控制台工程**，用C++语言开发景区旅游信息管理系统。

从**文件**中读取景区信息，然后**创建景区景点图**。

用户可以通过输入**景点编号**来**查询景点**的相关信息。

步骤一：创建工程

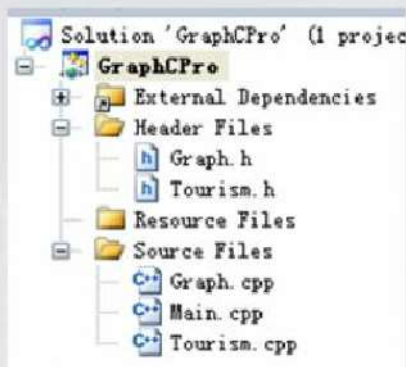
步骤二：定义图

步骤三：创建景区景点图

步骤四：查询景点信息

步骤一：创建工程

- 1、使用Microsoft Visual Studio2010开发工具创建空的控制台工程GraphCPro。
- 2、使用类向导添加CGraph类，用来存储图的信息和实现图的相关操作。
系统会生成Graph.h文件和Graph.cpp文件。
- 3、添加CTourism类，用来实现景区管理系统的相关业务操作。
系统会生成Tourism.h和Tourism.cpp文件。
- 4、添加Main.cpp文件，用来定义程序入口函数。



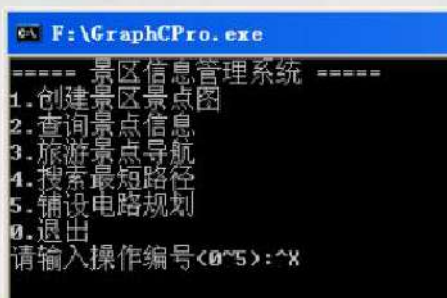
5、定义程序入口函数。

(1) 在Main.cpp文件中，添加int main(void)函数作为程序入口函数。

(2) 在main函数里添加以下代码，显示菜单。

```
int main(void)
{
    while(bRunning)
    {
        // 输出界面
        cout<<"===== 景区信息管理系统 =====<<endl;
        cout<<"1. 创建景区景点图"<<endl;
        cout<<"2. 查询景点信息"<<endl;
        cout<<"3. 旅游景点导航"<<endl;
        cout<<"4. 搜索最短路径"<<endl;
        cout<<"5. 铺设电路规划"<<endl;
        cout<<"0. 退出"<<endl;
        cout<<"请输入操作编号(0~5):"<<endl;
    }
}
```

编译运行：



```
F:\GraphCPro.exe
===== 景区信息管理系统 =====
1. 创建景区景点图
2. 查询景点信息
3. 旅游景点导航
4. 搜索最短路径
5. 铺设电路规划
0. 退出
请输入操作编号(0~5):^X
```

步骤二：定义图

在CGraph类中定义**Vex**和**Edge**结构体，来分别存储图的顶点和边的信息。

定义**顶点数组**和**邻接矩阵**，用来保存图的信息。

1、定义顶点和边

(1) Vex结构体

```
struct Vex
{
    int num;           // 景点编号
    char name[20];     // 景点名字
    char desc[1024];   // 景点介绍
};
```

(2) Edge结构体

```
struct Edge
{
    int vex1;          // 边的第一个顶点
    int vex2;          // 边的第二个顶点
    int weight;        // 权值
};
```

2、初始化图

(1)在CGraph类添加私有成员变量，定义邻接矩阵m_aAdjMatrix，顶点信息数组m_aVexs，当前顶点数目m_nVexNum。

```
Class CGraph
{
private:
    int m_aAdjMatrix [20][20];    // 邻接矩阵
    Vex m_aVexs [20];             // 顶点信息数组
    int m_nVexNum;                 // 当前图的顶点个数
}
```

(2)在CGraph类中创建void Init(void)函数，用来实现图的初始化。

其中权值信息都初始化为0，表示所有的顶点之间都不存在边。

步骤三：创建景区景点图

1、读取文件

工程目录下有两个文件Vex.txt和Edge.txt，分别存入了景点信息和道路信息。读取这两个文件，创建景区景点图。

(1)在CTourism类中创建void CreateGraph(void)函数，读取文件，获得景点信息和道路信息。

景点信息保存为结构体Vex；道路信息保存为结构体Edge。

(2)编译和运行。

在控制台输出读取到的信息。

2、插入顶点信息

在CGraph类中添加**bool InsertVex(Vex sVex)**函数，通过传入Vex结构体，将顶点的相关信息插入到顶点信息数组中。

```
bool CGraph::InsertVex(Vex sVex)
{
    if (m_nVexNum == MAX_VERTEX_NUM)
    {
        // 顶点已满
        return false;
    }

    m_aVexs[m_nVexNum++] = sVex; // 插入顶点信息
    return true;
}
```


3、插入边的信息

在CGraph类中添加**bool InsertEdge(Edge sEdge)**函数，通过传入Edge结构体，将边的相关信息插入到权值矩阵中。

```
bool CGraph::InsertEdge(Edge sEdge)
{
    if (sEdge.vex1 < 0 || sEdge.vex1 >= m_nVexNum || sEdge.vex2 < 0 || sEdge.vex2 >= m_nVexNum)
    {
        // 下标越界
        return false;
    }

    // 插入边的信息
    m_aAdjMatrix[sEdge.vex1][sEdge.vex2] = sEdge.weight;
    m_aAdjMatrix[sEdge.vex2][sEdge.vex1] = sEdge.weight;

    return true;
}
```

4、创建景区景点图

在CTourism类中添加一个私有数据成员**CGraph m_Graph**，通过调用m_Graph对象的函数，将景点信息保存成图。

(1)在CreateGraph()函数中，调用**m_Graph.Init()**函数，初始化图。

(2)调用**m_Graph.InsertVex()**函数，将顶点信息插入到图中。

(3)调用**m_Graph.InsertEdge()**函数，将边的信息插入到图中。

```
void CTourism::CreateGraph(void)
{
    ////////////////////////////////// 1 初始化图
    m_Graph.Init();

    ////////////////////////////////// 2 设置图的顶点
    // ...
    m_Graph.InsertVex(vex);

    ////////////////////////////////// 3 设置图的边
    // ...
    m_Graph.InsertEdge(edge);
}
```

(4)编译和运行

```
C:\WINDOWS\system32\cmd.exe
===== 创建景区景点图 =====
顶点数目: 7
----- 顶点 -----
0-A区
1-B区
2-C区
3-D区
4-E区
5-F区
6-G区
----- 边 -----
(v0,v2) 700
(v0,v4) 1000
(v0,v5) 600
(v1,v2) 1000
(v1,v6) 1000
(v2,v3) 400
(v3,v4) 300
(v3,v6) 400
(v4,v5) 600
(v5,v6) 500
```

步骤四：查询景点信息

显示所有景点的编号和名字供用户选择。

输入景点编号，显示景点详细信息。查询所有与该景点相邻的景点，显示到达这些景点的距离。

1、显示景点列表

(1) 在CGraph类中创建**Vex GetVex(int v)**函数，通过传入的**顶点编号**，得到对应的顶点信息，并**返回顶点结构体**。

```
⊞ Vex CGraph::GetVex(int v)
{
    return m_aVexs[v];
}
```

(2) 在CTourism类中创建**void GetSpotInfo(void)**函数，调用**m_Graph.GexVex()**函数，得到所有顶点信息，并输出到界面上。

(3) 编译和运行。

2、查询相邻景点

(1)在CGraph类中添加**int FindEdge(int v, Edge aEdge[])**函数，查询所有与指定顶点相连的边。

```
int CGraph::FindEdge(int v, Edge aEdge[])
{
    int k = 0;
    for(int i = 0; i < m_nVexNum; i++)
    {
        //得到边的信息
        .....
        k++;
    }
    return k; //返回边的条数
}
```

(2)在CTourism类的GetSpotInfo()函数中调用**m_Graph.FindEdge()**函数，得到所有相邻的边。根据边找到所有相邻顶点，在控制台下输出到这些相邻景点的距离。

(3)编译和运行。

```
C:\WINDOWS\system32\cmd.exe

===== 查询景点信息 =====
0-A区
1-B区
2-C区
3-D区
4-E区
5-F区
6-G区
请输入想要查询的景点编号：2
C区
风景优美，气候宜人。门票20元。
----- 周边景区 -----
C区->A区 700m
C区->B区 1000m
C区->D区 400m
```

1、图的定义 $G=(V,E)$

2、图的存储

(1) 顶点的存储：一维数组

(2) 边的存储：关系矩阵

3、图的创建