**PJ3实验文档**

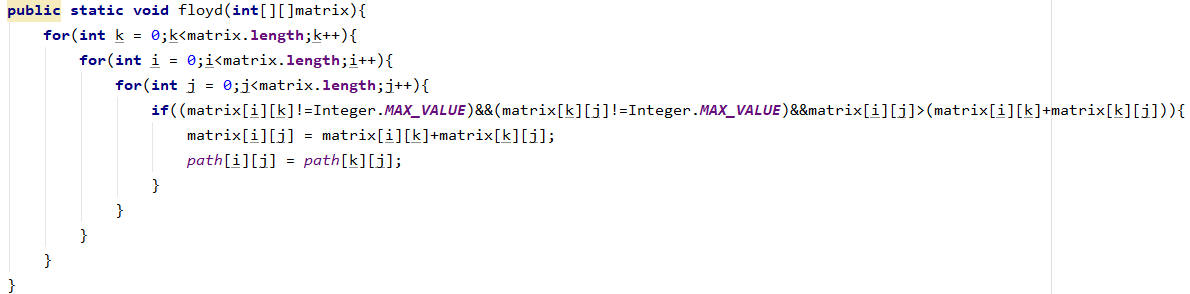
程茜 19302010084

1. **最短路径算法实现**

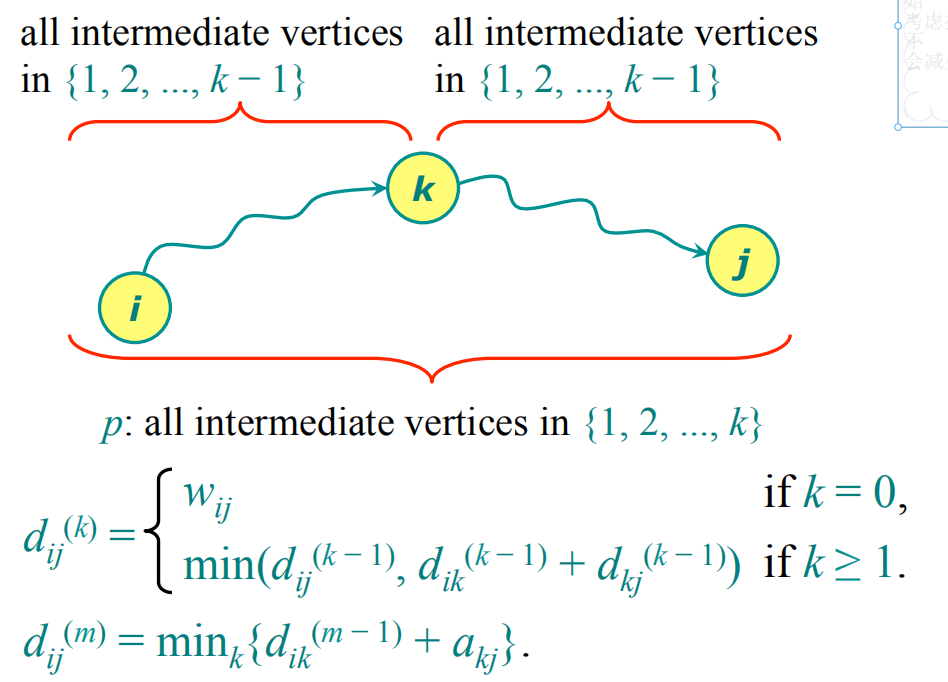
（1）采用的**Floyd-Warshall**算法，输入为一个矩阵和一个保存各个节点的数组，用于路径的输出。算法用到了两个矩阵，一个矩阵（**matrix**）记录从i到j的最短路径长度，还有一个辅助矩阵（**path**）记录从i到j的最短路径中j的前一个节点。

（2）邻接矩阵matrix中若两点没有路径则值为无穷大，即**Integer.MAX\_VALUE**，定义natrix[i][i]=0。然后首先对辅助矩阵path初始化为path[i][j]=i。

（3）算法主体如下：

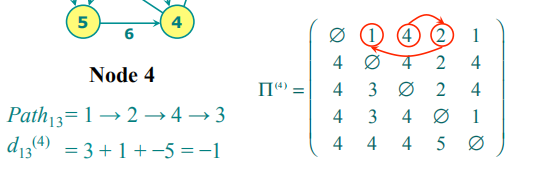


算法的思路是在一条目前已知的从i到j的最短路径中考虑加不加入点k，如果加入点k后路径变短则加入，反之不加入，算法的时间复杂度为O(n3)



1. **最短路径打印**

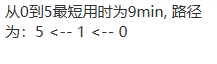
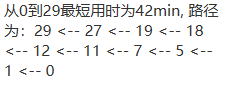
最短路径的打印需要借助辅助矩阵，如下图所示。如果从i到j的最短路径值为无穷大，则说明没有相应路径，则输出相关内容。反之，要输出从i到j的最短路径，只需要从path[i][j]的值开始，一直到path[i][m]=i，找到的即为最短路径，代码如下：





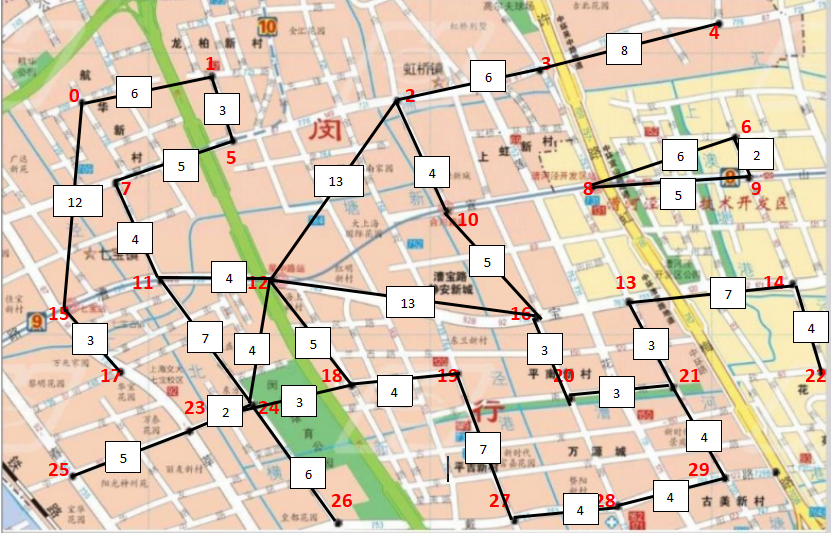
利用了StringBuider类来逐个几率路径上经过的点，最后返回一个String字符串。

输出示例：

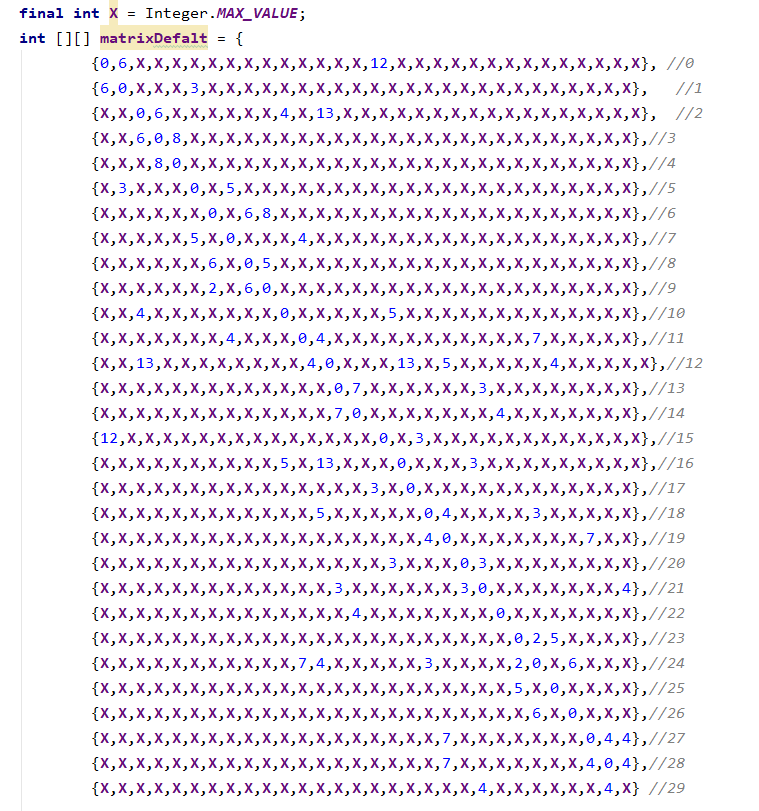
 

1. **地图选择：**

从图中选取了30个点，标号为0-29，用红色标出，黑色直线代表两点可直接到达，上面的数字是步行用时。

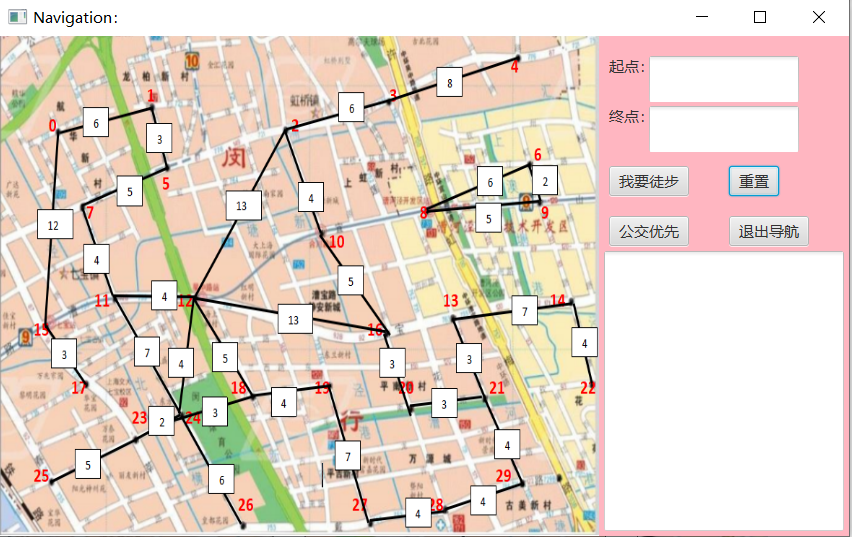


根据地图得到的邻接矩阵为：

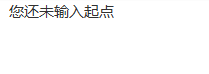
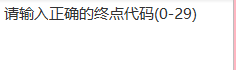


1. **界面实现**

利用javafx实现了简单的UI界面，左边是地图，右边是按钮和输入输出框，用户可以输入起点和终点的对应代码，然后点击我要徒步按钮，就会在下方的输出框中输出最短路径。



如果用户没有输入起点终点或者输入的代码不对，都会在下方的输出框里提示相应内容：

当用户点击重置时，会清空所有输入框里面的内容，点击退出导航则可以退出程序。

示例：

