Dijkstra算法

迪杰斯特拉(Dijkstra)算法是典型的最短路径算法，用于计算一个节点到其他节点的最短路径。它是以起始点为中心向外层层扩展（广度优先搜索），直到扩展到终点为止。

**基本思想**

通过Dijkstra计算图G中的最短路径时，需要指定起始点，此外，为了便于理解，引入两个集合S和U。S的作用是记录以求出最短路径的顶点（以及相应的最短路径长度），而U则是记录还未求出最短路径的顶点。

初始时，S中只有起点s；U中是除s之外的顶点，并且U中顶点的路径是“起点s到该顶点的路径”。然后，从U中找出路径最短的顶点，并将其加入到S中；接着，更新U中的顶点和顶点对应的路径。然后，再从U中找出路径最短的顶点，并将其加入到S中；接着，更新U中的顶点和顶点对应的路径。重复该操作，直到遍历完所有顶点。

**操作步骤**

* 1）初始时，S中只包含起点s; U包含除s外的其他顶点，且U中顶点的距离为“起点s到该顶点的距离”【例如，U中的顶点v的距离为(s,v)的长度，然后s和v不相邻，则v的距离为】。
* 2）从U中选出“距离最短的顶点k”，并将顶点k加入到S中；同时，从U中移除顶点k。
* 3）更新U中各个顶点到起点s的距离。之所以更新U中顶点的距离，是由于上一步中确定了k是求出最短路径的顶点，从而可以利用k来更新其他顶点的距离，例如，(s,v)的距离可能大于(s,k)+(k,v)的距离。
* 4）重复 步骤2和步骤3，直到遍历完所有顶点。

**Dijkstra算法图解**

如下图是一个距离图：

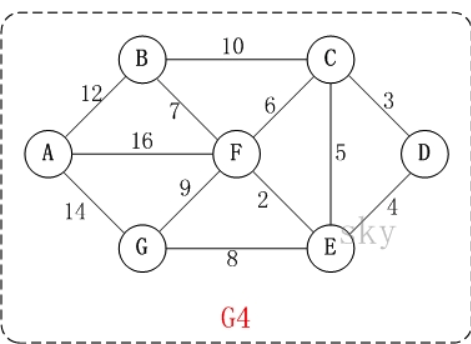
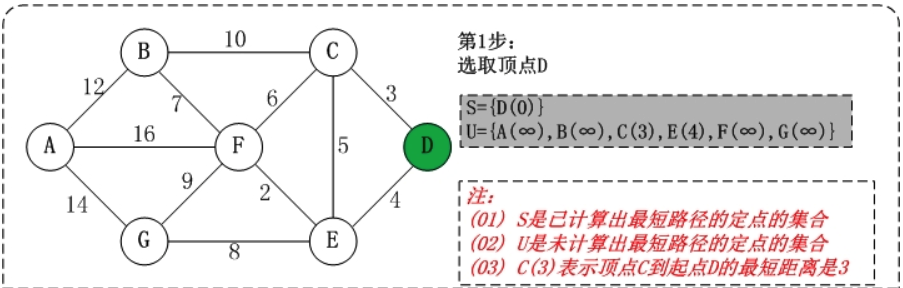


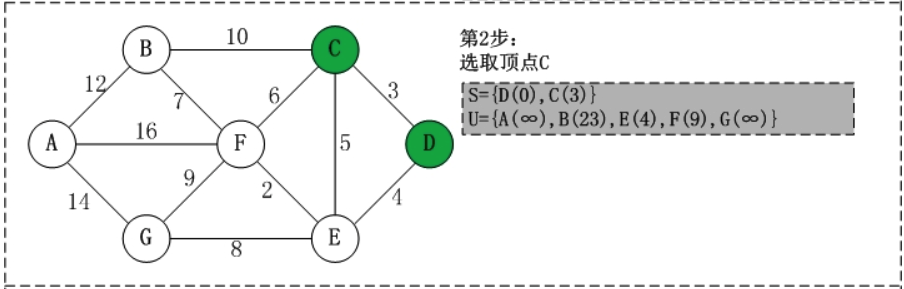
图1：距离图

我们以D为起点，来对Dijkstra算法的具体流程进行详解。

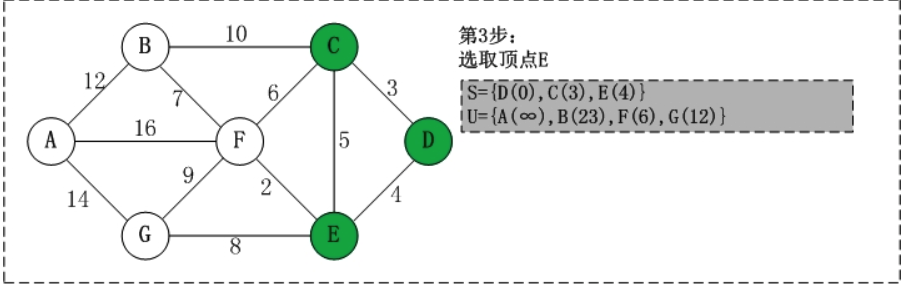
**第一步**：



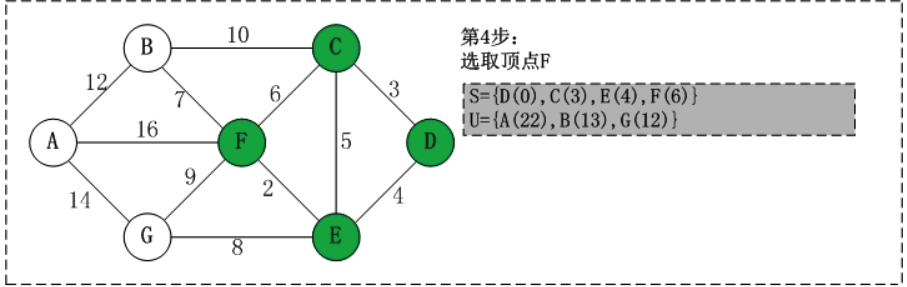
**第二步**[这里的U中的B(23)，应该为B(13)下面一样]：



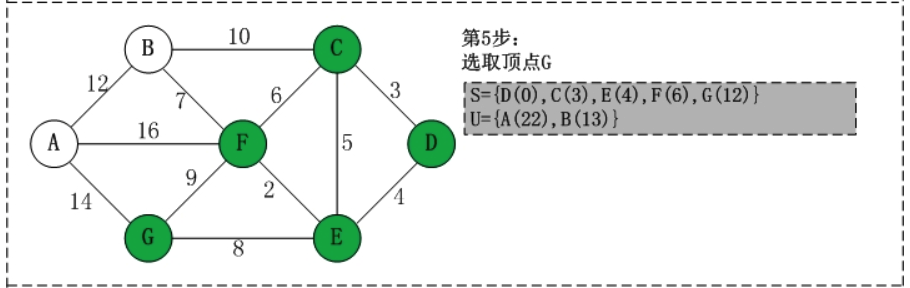
**第三步：**

****

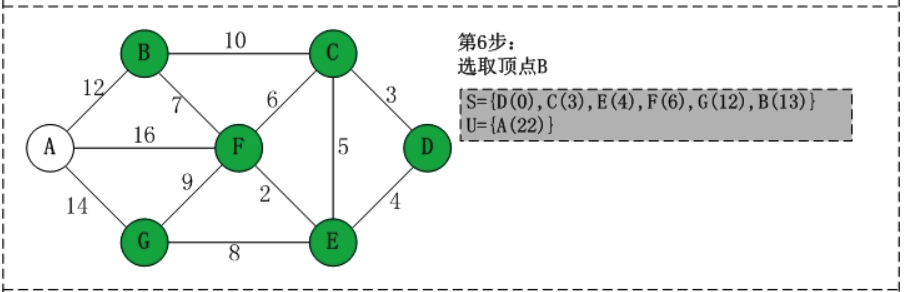
**第四步：**

****

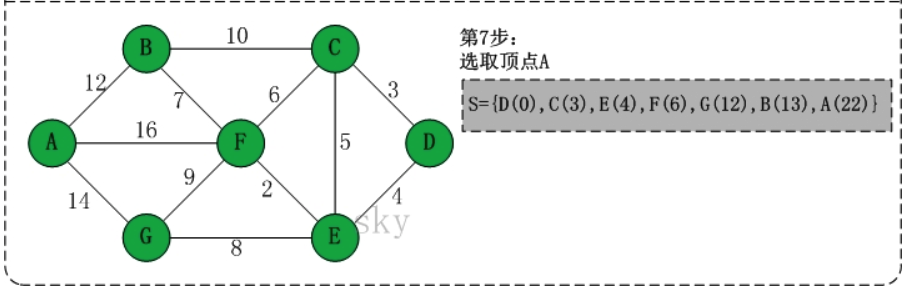
**第五步：**

****

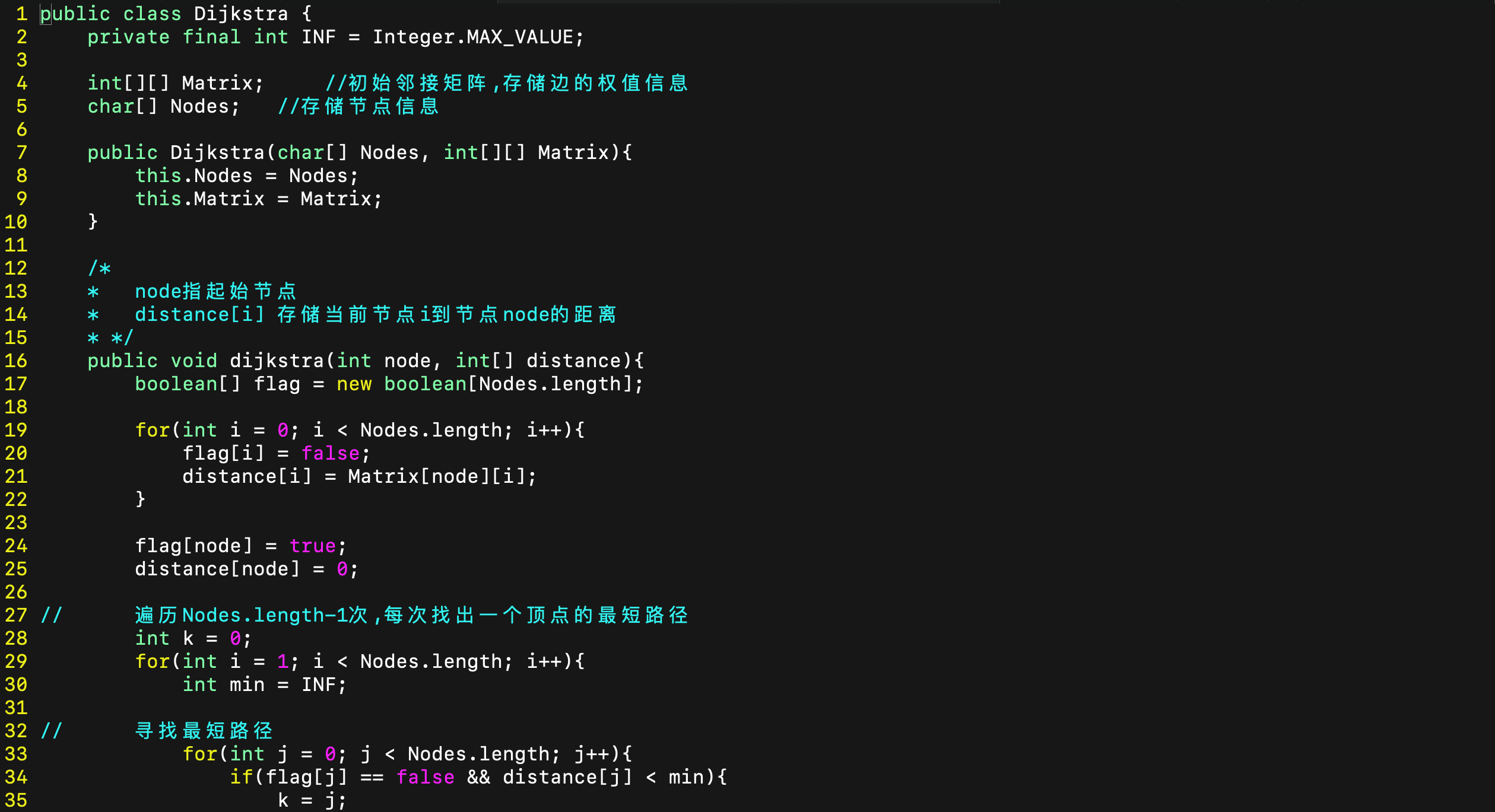
**第六步：**

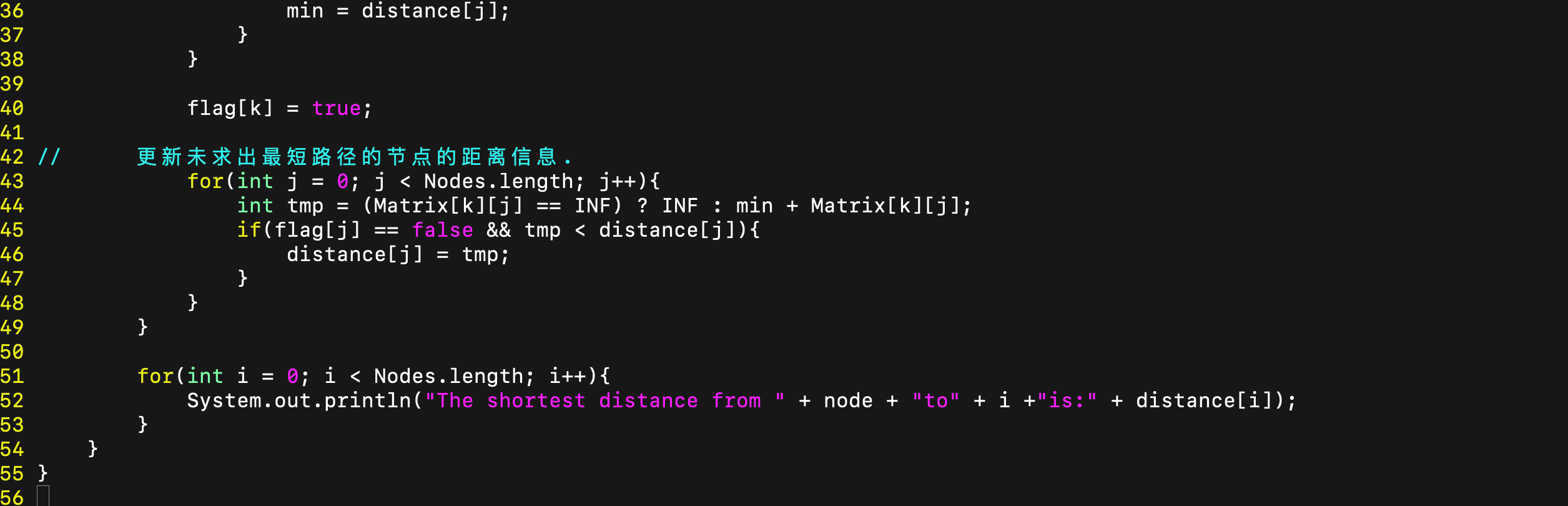
****

**第七步：**

****

**Dijkstra代码：**

****

****