Floyd算法

Floyd算法与Dijkstra算法一样都是用于寻找给定的加权图中顶点间的最短路径。区别则是，Floyd最后求出的则是每个顶点间的最短距离，而Dijkstra算法则是求出到一个顶点的最短距离。

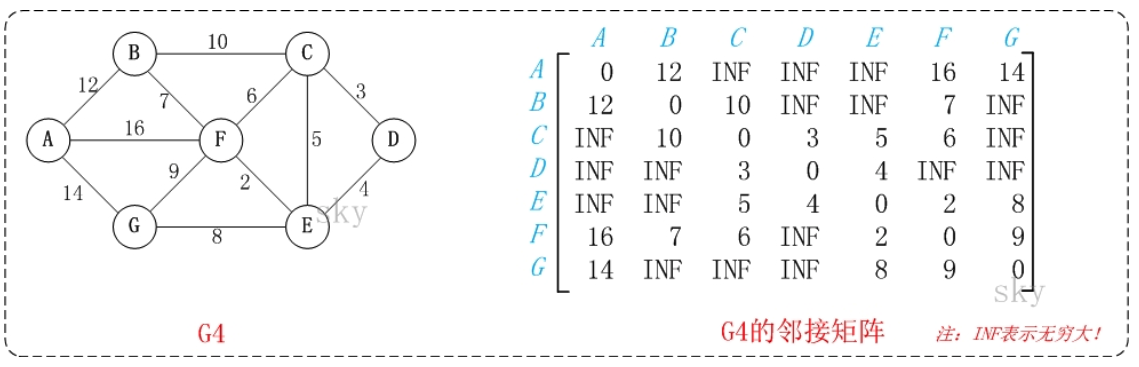
**基本思想**

通过Floyd算法计算图G=(V,E)中各个顶点的最短路径是，需要引入一个矩阵S，矩阵S中的元素S[i][j]表示顶点i，j之间的距离。

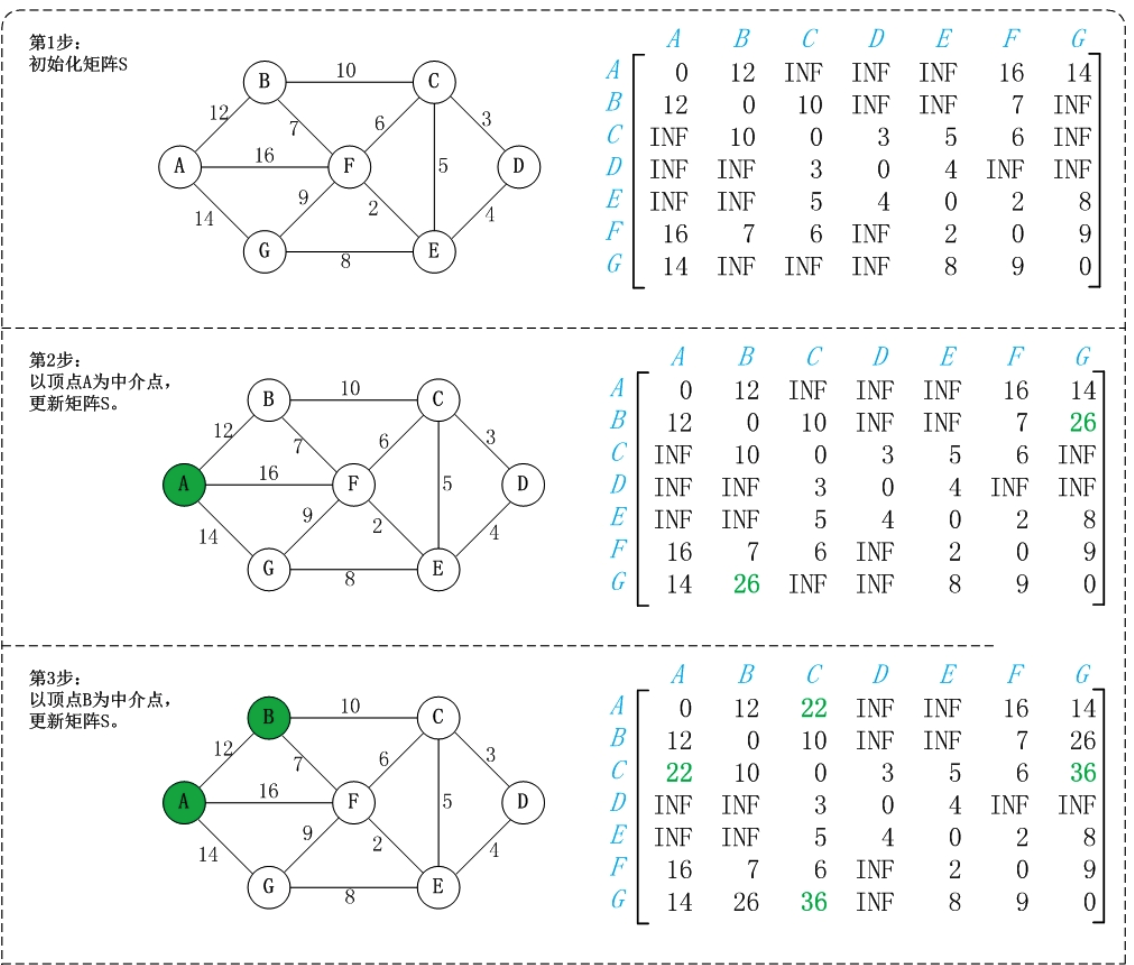
假设图G中顶点的个数为N，则需要对矩阵S进行N次更新。初始时，矩阵S中顶点S[i][j]为顶点i到顶点j的权值；如果i和j不相邻，则S[i][j] = 。接下来开始，对矩阵S进行N次更新。第一次更新时，如果S[i][j] > S[i][0] + S[0][j] (S[i][0] + S[0][j 表示i与j之间经过第1个顶点的距离)，则更新S[i][j]为 S[i][0] + S[0][j]。同理，第k次更新时，如果S[i][j] > S[i][k] + S[k][j]，则更新为S[i][k] + S[k][j]。更新N次后，操作完成。

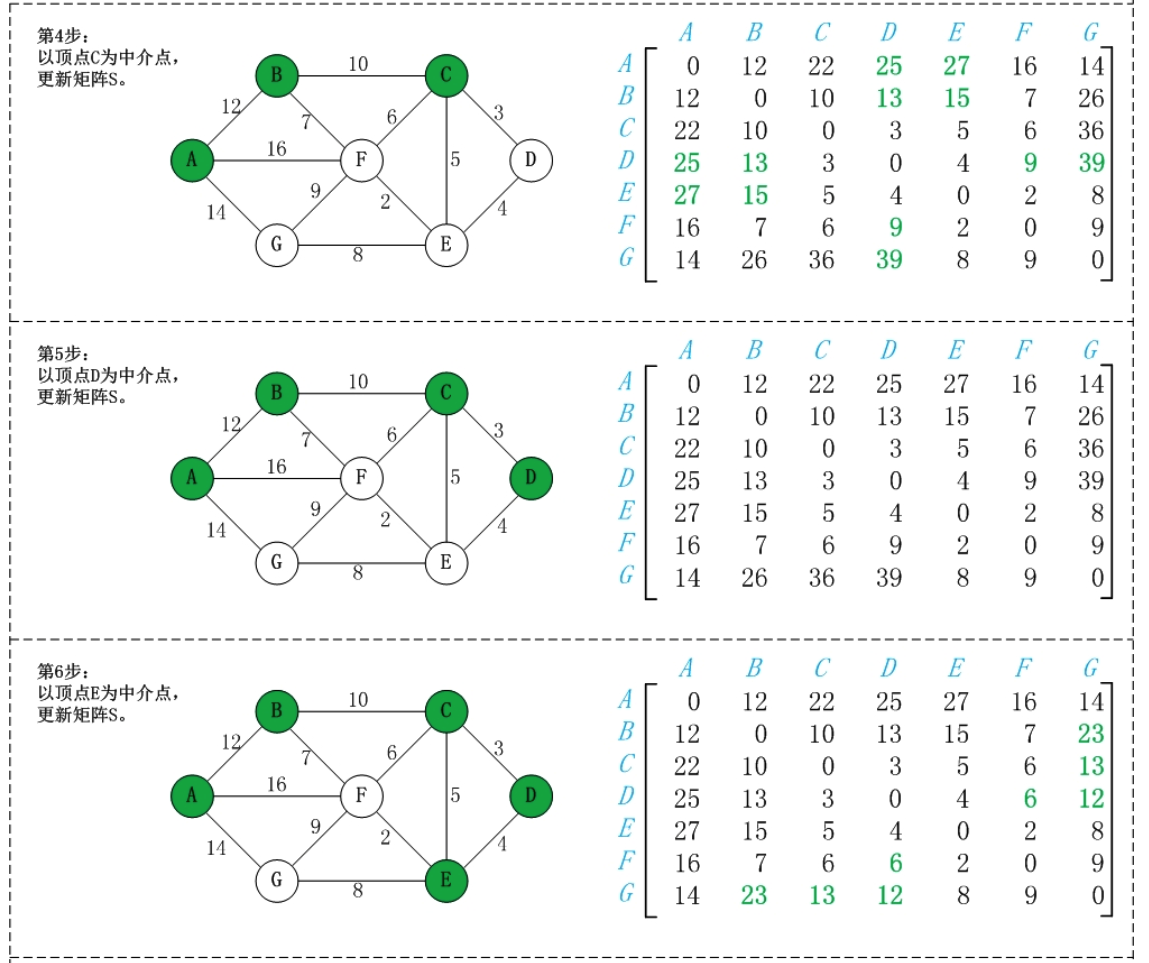
下面通过实例来对算法进行说明。

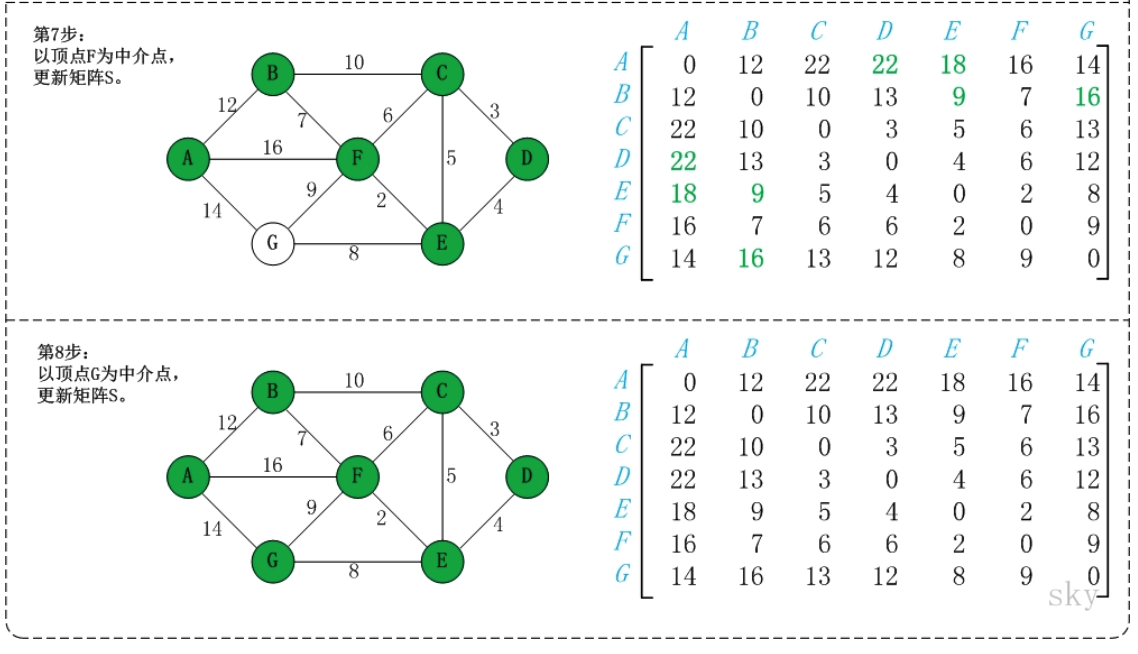
**Floyd算法图解**



上图的G4是一个邻接矩阵的实例，我们以其为例进行算法演示：

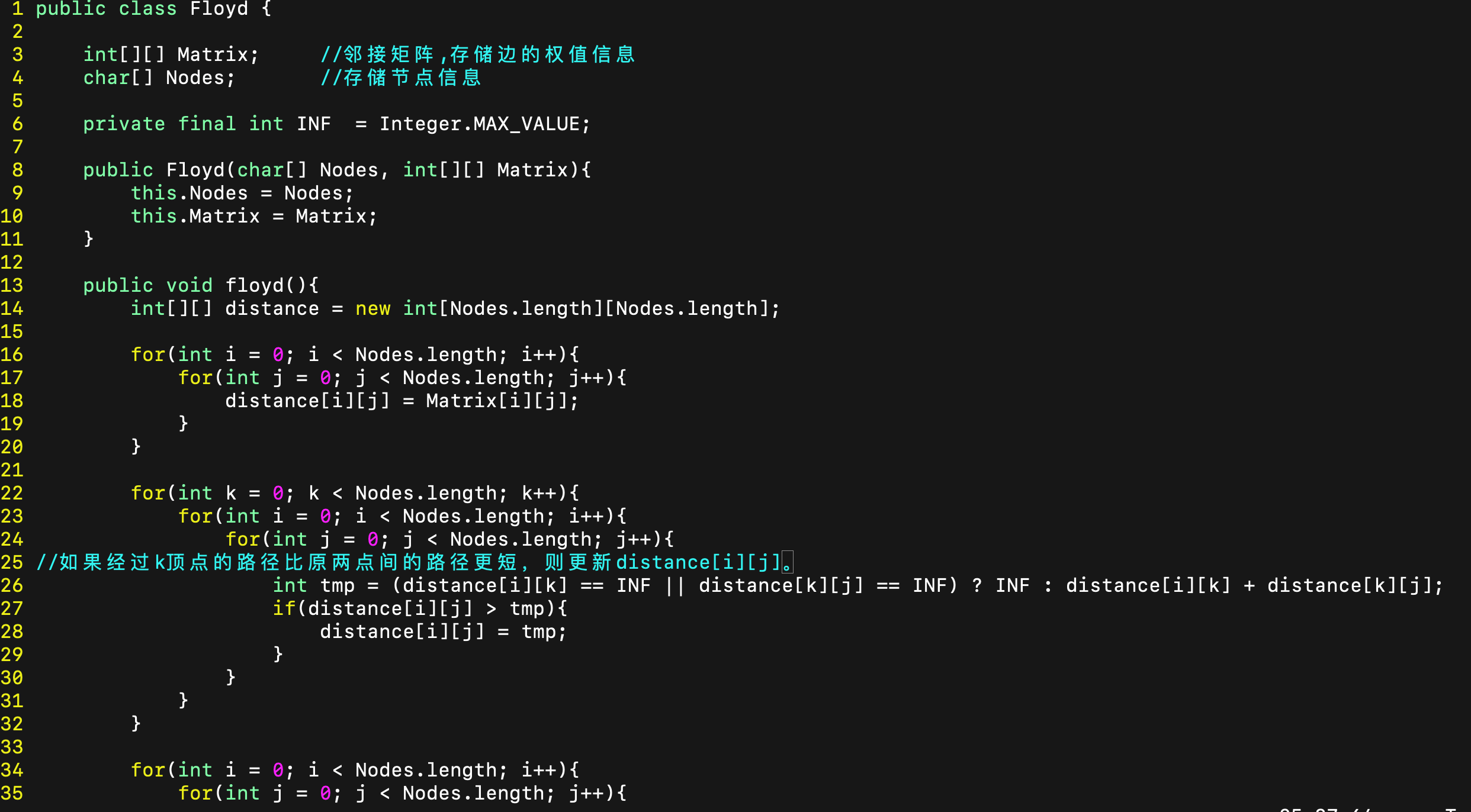


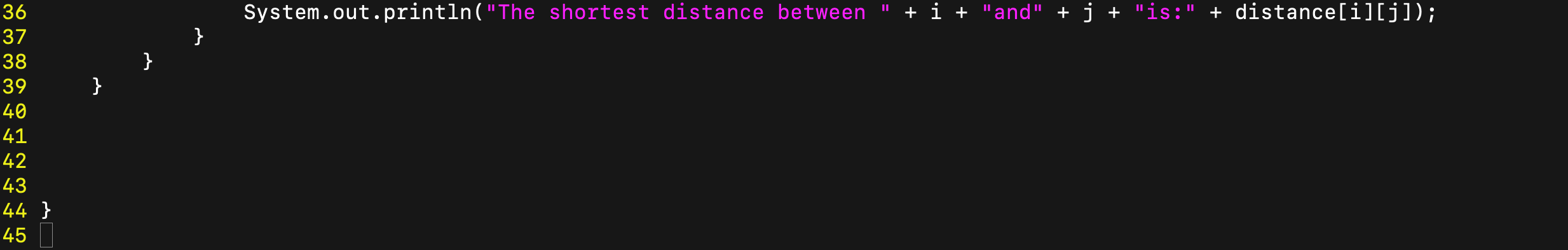




如上是，按照节点一个个进行更新邻接矩阵，更新N次后的邻接矩阵中存储的就是节点之间的最短距离。

**代码**

****

****