Dijkstra 算法实现 sk 导航系统

一、任务及要求：

选用合适的数据结构及算法实现苏州科技大学石湖校区导航系统,使得当客人无论从 1,2 号校门进入学校及在学校的某个教学楼、图书馆、办公楼、塔影阁、东宿舍区、西宿舍区、快递点等地点都能最快游览到其它地点，并能顺利离开学校。

二、设计与实现说明：主要说明你的设计及实现思路,采用的数据结构及类和方法的定义、功能、特点。

1、图是由顶点集 V 和边集 E 组成，记为 G=(V,E)。V 是有穷非空集合，称为顶点集，v∈V 称为顶点。E 是有穷集合，称为边集，e∈E 称为边。

2、全部由有向边构成的图称为有向图。

3、戴克斯特拉算法：通过 Dijkstra 计算图 G 中的最短路径时，需要指定起点 vs(即从顶点 vs 开始计算)。此外，引进两个集合 S 和 U。S 的作用是记录已求出最短路径的顶点，而 U 则是记录还未求出最短路径的顶点(以及该顶点到起点 vs 的距离)。初始时，S 中只有起点 vs；U 中是除 vs 之外的顶点，并且 U 中顶点的路径是"起点 vs 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。

4、createGraph1(int index):图的创建

5、dijkstra(int vs):戴克特斯拉算法。（vs -- 起始顶点(start vertex) 即，统计图中"顶点 vs"到其它各个顶点的最短路径）

步骤一：初始时，S 中只有起点 vs；U 中是除 vs 之外的顶点，并且 U 中顶点的路径是"起点 vs 到该顶点的路径"。

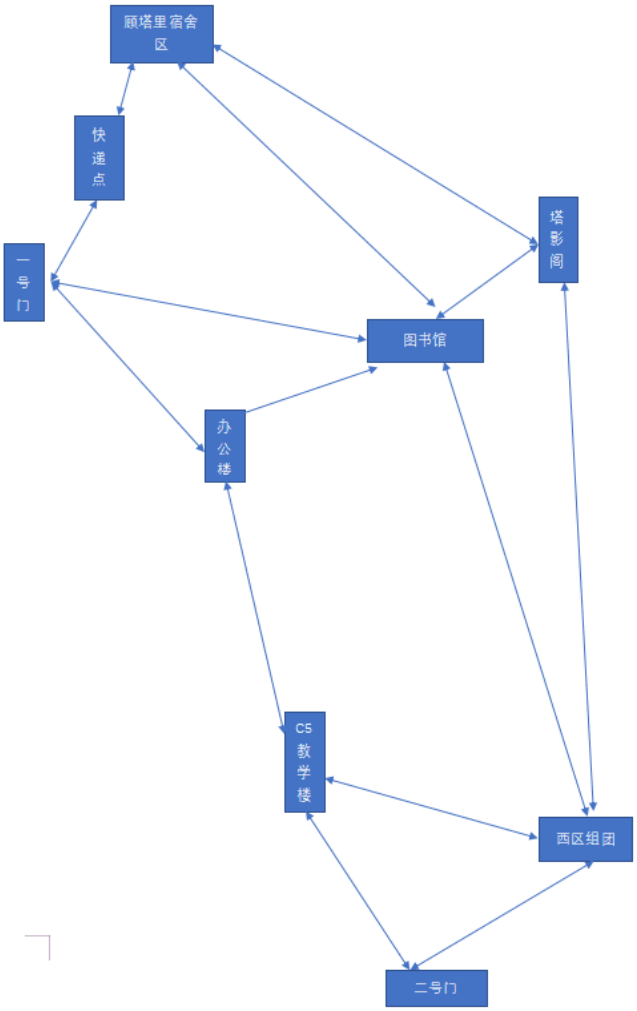
步骤二：从 U 中找出路径最短的顶点，并将其加入到 S 中。

步骤三：更新 U 中的顶点和顶点对应的路径。

步骤四：重复步骤二三，直到遍历完所有顶点。

6、图可分两部分：一个是创建顶点信息，可用一个一维数组存储；另一个是创建弧的信息，包括弧的相关顶点和权值，可存储到二维数组里，其中，二维数组的下标分别表示两个顶点的弧尾和弧头编号，权值存放在对应的数组中。

7、有向图



8、邻接矩阵

矩阵	一号门	塔影阁	C5 教学楼	图书馆	办公楼	顾塔里宿舍区	西区组团	快递点	二号门
一号门	0	∞	∞	∞	4	∞	∞	2	∞
塔影阁	∞	0	∞	1	∞	7	9	∞	∞
C5 教学楼	∞	∞	0	∞	6	∞	2	∞	3
图书馆	∞	1	∞	0	2	6	8	∞	∞
办公楼	4	∞	6	2	0	∞	∞	∞	∞
顾塔里宿舍区	∞	7	∞	6	∞	0	∞	2	∞
西区组团	∞	9	2	8	∞	∞	0	∞	2
快递点	2	∞	∞	∞	∞	2	∞	0	∞
二号门	∞	∞	3	∞	∞	∞	2	∞	0

三、测试结果：包括输入和输出 (粘贴输入输出界面，实现结果界面等)。

```
**************************欢迎来到苏州科技大学石湖校区导航系统**************************
苏科导航已为您规划最佳游览路线，请您查收，仔细阅读
起始顶点：1号门(苏州科技大学石湖校区)
最短路径（1号门,1号门):0  1号门
最短路径（1号门,塔影阁):7  1号门->办公楼->图书馆->塔影阁
最短路径（1号门,C5教学楼):10  1号门->办公楼->C5教学楼
最短路径（1号门,图书馆):6  1号门->办公楼->图书馆
最短路径（1号门,办公楼):4  1号门->办公楼
最短路径（1号门,顾塔里宿舍区):4  1号门->快递点->顾塔里宿舍区
最短路径（1号门,西区组团):12  1号门->办公楼->C5教学楼->西区组团
最短路径（1号门,快递点):2  1号门->快递点
最短路径（1号门,2号门):13  1号门->办公楼->C5教学楼->2号门
顶点放入S中的顺序：
1号门-->快递点-->办公楼-->顾塔里宿舍区-->图书馆-->塔影阁-->C5教学楼-->西区组团-->2号门
起始顶点：2号门(苏州科技大学石湖校区)
最短路径（2号门,2号门):0  2号门
最短路径（2号门,塔影阁):11  2号门->西区组团->塔影阁
最短路径（2号门,C5教学楼):3  2号门->C5教学楼
最短路径（2号门,图书馆):10  2号门->西区组团->图书馆
最短路径（2号门,办公楼):9  2号门->C5教学楼->办公楼
最短路径（2号门,顾塔里宿舍区):16  2号门->西区组团->图书馆->顾塔里宿舍区
最短路径（2号门,西区组团):2  2号门->西区组团
最短路径（2号门,快递点):15  2号门->C5教学楼->办公楼->1号门->快递点
最短路径（2号门,1号门):13  2号门->C5教学楼->办公楼->1号门
顶点放入S中的顺序：
2号门-->西区组团-->C5教学楼-->办公楼-->图书馆-->塔影阁-->1号门-->快递点-->顾塔里宿舍区
起始顶点：塔影阁(苏州科技大学石湖校区)
最短路径（塔影阁,塔影阁):0  塔影阁
最短路径（塔影阁,2号门):11  塔影阁->西区组团->2号门
最短路径（塔影阁,C5教学楼):9  塔影阁->图书馆->办公楼->C5教学楼
最短路径（塔影阁,图书馆):1  塔影阁->图书馆
最短路径（塔影阁,办公楼):3  塔影阁->图书馆->办公楼
最短路径（塔影阁,顾塔里宿舍区):7  塔影阁->顾塔里宿舍区
最短路径（塔影阁,西区组团):9  塔影阁->西区组团
最短路径（塔影阁,快递点):9  塔影阁->顾塔里宿舍区->快递点
最短路径（塔影阁,1号门):7  塔影阁->图书馆->办公楼->1号门
顶点放入S中的顺序：
塔影阁-->图书馆-->办公楼-->顾塔里宿舍区-->1号门-->C5教学楼-->西区组团-->快递点-->2号门
起始顶点：C5教学楼(苏州科技大学石湖校区)
最短路径（C5教学楼,C5教学楼):0  C5教学楼
最短路径（C5教学楼,塔影阁):9  C5教学楼->办公楼->图书馆->塔影阁
最短路径（C5教学楼,2号门):3  C5教学楼->2号门
```

<已终止> ShortestPathDijkstra [Java 应用程序] D:\java\Eclispe\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.ful
最短路径（C5教学楼,西区组团):2 C5教学楼->西区组团
最短路径（C5教学楼,快递点):12 C5教学楼->办公楼->1号门->快递点
最短路径（C5教学楼,1号门):10 C5教学楼->办公楼->1号门
顶点放入S中的顺序：
C5教学楼-->西区组团-->2号门-->办公楼-->图书馆-->塔影阁-->1号门-->快递点-->顾塔里宿舍区
起始顶点：图书馆(苏州科技大学石湖校区)
最短路径（图书馆,图书馆):0 图书馆
最短路径（图书馆,塔影阁):1 图书馆->塔影阁
最短路径（图书馆,C5教学楼):8 图书馆->办公楼->C5教学楼
最短路径（图书馆,2号门):10 图书馆->西区组团->2号门
最短路径（图书馆,办公楼):2 图书馆->办公楼
最短路径（图书馆,顾塔里宿舍区):6 图书馆->顾塔里宿舍区
最短路径（图书馆,西区组团):8 图书馆->西区组团
最短路径（图书馆,快递点):8 图书馆->顾塔里宿舍区->快递点
最短路径（图书馆,1号门):6 图书馆->办公楼->1号门
顶点放入S中的顺序：
图书馆-->塔影阁-->办公楼-->顾塔里宿舍区-->1号门-->C5教学楼-->西区组团-->快递点-->2号门
起始顶点：办公楼(苏州科技大学石湖校区)
最短路径（办公楼,办公楼):0 办公楼
最短路径（办公楼,塔影阁):3 办公楼->图书馆->塔影阁
最短路径（办公楼,C5教学楼):6 办公楼->C5教学楼
最短路径（办公楼,图书馆):2 办公楼->图书馆
最短路径（办公楼,2号门):9 办公楼->C5教学楼->2号门
最短路径（办公楼,顾塔里宿舍区):8 办公楼->图书馆->顾塔里宿舍区
最短路径（办公楼,西区组团):8 办公楼->C5教学楼->西区组团
最短路径（办公楼,快递点):6 办公楼->1号门->快递点
最短路径（办公楼,1号门):4 办公楼->1号门
顶点放入S中的顺序：
办公楼-->图书馆-->塔影阁-->1号门-->C5教学楼-->快递点-->顾塔里宿舍区-->西区组团-->2号门
起始顶点：顾塔里宿舍区(苏州科技大学石湖校区)
最短路径（顾塔里宿舍区,顾塔里宿舍区):0 顾塔里宿舍区
最短路径（顾塔里宿舍区,塔影阁):2 顾塔里宿舍区->图书馆->塔影阁
最短路径（顾塔里宿舍区,C5教学楼):9 顾塔里宿舍区->图书馆->2号门->C5教学楼
最短路径（顾塔里宿舍区,图书馆):1 顾塔里宿舍区->图书馆
最短路径（顾塔里宿舍区,2号门):3 顾塔里宿舍区->图书馆->2号门
最短路径（顾塔里宿舍区,顾塔里宿舍区):11 顾塔里宿舍区->图书馆->西区组团->顾塔里宿舍区
最短路径（顾塔里宿舍区,西区组团):9 顾塔里宿舍区->图书馆->西区组团
最短路径（顾塔里宿舍区,快递点):2 顾塔里宿舍区->快递点
最短路径（顾塔里宿舍区,1号门):4 顾塔里宿舍区->快递点->1号门
顶点放入S中的顺序：
顾塔里宿舍区-->图书馆-->塔影阁-->快递点-->2号门-->1号门-->C5教学楼-->西区组团-->顾塔里宿舍区
起始顶点：西区组团(苏州科技大学石湖校区)
最短路径（西区组团,西区组团):0 西区组团
顶点放入S中的顺序：
顾塔里宿舍区-->图书馆-->塔影阁-->快递点-->2号门-->1号门-->C5教学楼-->西区组团-->顾塔里宿舍区
起始顶点：西区组团(苏州科技大学石湖校区)
最短路径（西区组团,西区组团):0 西区组团
最短路径（西区组团,塔影阁):9 西区组团->塔影阁
最短路径（西区组团,C5教学楼):2 西区组团->C5教学楼
最短路径（西区组团,图书馆):8 西区组团->图书馆
最短路径（西区组团,办公楼):8 西区组团->C5教学楼->办公楼
最短路径（西区组团,顾塔里宿舍区):14 西区组团->图书馆->顾塔里宿舍区
最短路径（西区组团,2号门):2 西区组团->2号门
最短路径（西区组团,快递点):14 西区组团->C5教学楼->办公楼->1号门->快递点
最短路径（西区组团,1号门):12 西区组团->C5教学楼->办公楼->1号门
顶点放入S中的顺序：
西区组团-->C5教学楼-->2号门-->图书馆-->办公楼-->塔影阁-->1号门-->顾塔里宿舍区-->快递点
起始顶点：快递点(苏州科技大学石湖校区)
最短路径（快递点,快递点):0 快递点
最短路径（快递点,塔影阁):9 快递点->顾塔里宿舍区->塔影阁
最短路径（快递点,C5教学楼):12 快递点->1号门->办公楼->C5教学楼
最短路径（快递点,图书馆):8 快递点->顾塔里宿舍区->图书馆
最短路径（快递点,办公楼):6 快递点->1号门->办公楼
最短路径（快递点,顾塔里宿舍区):2 快递点->顾塔里宿舍区
最短路径（快递点,西区组团):14 快递点->1号门->办公楼->C5教学楼->西区组团
最短路径（快递点,2号门):15 快递点->1号门->办公楼->C5教学楼->2号门
最短路径（快递点,1号门):2 快递点->1号门
顶点放入S中的顺序：
快递点-->顾塔里宿舍区-->1号门-->办公楼-->图书馆-->塔影阁-->C5教学楼-->西区组团-->2号门
导航结束，谢谢使用

四、带注释的源代码及相关说明：

```
package sj;
import java.util.*;
public class ShortestPathDijkstra {
    /** 邻接矩阵 */
    private int[][] matrix;
    /** 表示正无穷 */
    private int MAX_WEIGHT = Integer.MAX_VALUE;
    /** 顶点集合 */
    private String[] vertexes;

    /**
     * 创建图
     */
    private void createGraph(int index){
        matrix = new int[index][index];
        vertexes = new String[index];

        int[] v0 = { 0, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 4,
MAX_WEIGHT, MAX_WEIGHT, 2, MAX_WEIGHT };
        int[] v1 = {MAX_WEIGHT,0, MAX_WEIGHT, 1, MAX_WEIGHT,7,
9, MAX_WEIGHT, MAX_WEIGHT };
        int[] v2 = { MAX_WEIGHT, MAX_WEIGHT, 0,MAX_WEIGHT, 6,
MAX_WEIGHT, 2, MAX_WEIGHT, 3 };
        int[] v3 = { MAX_WEIGHT, 1, MAX_WEIGHT, 0, 2, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
        int[] v4 = {4, MAX_WEIGHT, 6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT };
        int[] v5 = { MAX_WEIGHT, 7, MAX_WEIGHT, 6, MAX_WEIGHT, 0,
MAX_WEIGHT,2, MAX_WEIGHT };
        int[] v6 = { MAX_WEIGHT, 9, 2, 8,
MAX_WEIGHT,MAX_WEIGHT,0,MAX_WEIGHT,2};
        int[] v7 = { 2, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, 0, MAX_WEIGHT };
        int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, 3, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 2,MAX_WEIGHT, 0 };

        matrix[0] = v0;
        matrix[1] = v1;
        matrix[2] = v2;
        matrix[3] = v3;
        matrix[4] = v4;
        matrix[5] = v5;
```

```
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "1 号门";
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "2 号门";
        }
        private void createGraph1(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,3,MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v1 = {MAX_WEIGHT,0,MAX_WEIGHT, 1, MAX_WEIGHT, 7,
9, MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { 2, MAX_WEIGHT, 0,MAX_WEIGHT, 6, MAX_WEIGHT,
2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { MAX_WEIGHT, 1, MAX_WEIGHT, 0, 2, 6,9,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT,6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { MAX_WEIGHT, 7, MAX_WEIGHT,6, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = { 2, 9,2,8, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 2, MAX_WEIGHT, 0, 2};
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
```

```
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "2 号门";
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph2(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,MAX_WEIGHT,1, MAX_WEIGHT, 7,
9, MAX_WEIGHT, MAX_WEIGHT };
            int[] v1 = {MAX_WEIGHT,0,3, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { MAX_WEIGHT, 3, 0,MAX_WEIGHT, 6, MAX_WEIGHT,
2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { 1, MAX_WEIGHT, MAX_WEIGHT, 0, 2, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT, 6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { 7, MAX_WEIGHT, MAX_WEIGHT, 6, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = {9, 2, 2, 8, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 2, MAX_WEIGHT, 0, 2 };
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };
            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;
```

```
            vertexes[0] = "塔影阁";
            vertexes[1] = "2 号门";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph3(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,3,MAX_WEIGHT,6,MAX_WEIGHT, 2,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v1 = {MAX_WEIGHT,0,MAX_WEIGHT, 1, MAX_WEIGHT,7, 9,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { 3, MAX_WEIGHT, 0,MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { MAX_WEIGHT, 1, MAX_WEIGHT, 0, 2, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { 6, MAX_WEIGHT, MAX_WEIGHT,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { MAX_WEIGHT, 7, MAX_WEIGHT, 6, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = { 2, 9, 2, 8, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT,2, MAX_WEIGHT, 0,2 };
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT,4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };
            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "C5 教学楼";
```

```
            vertexes[1] = "塔影阁";
            vertexes[2] = "2 号门";
            vertexes[3] = "图书馆";
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph4(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, 1,MAX_WEIGHT,MAX_WEIGHT, 2, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v1 = {1,0,MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 7,9,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { MAX_WEIGHT, MAX_WEIGHT, 0,3, 6, MAX_WEIGHT,
2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { MAX_WEIGHT, MAX_WEIGHT, 3, 0, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { 2, MAX_WEIGHT, 6,MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { 6, 7, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = { 8, 9, 2, 2, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT };
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "图书馆";
```

```
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "2 号门";
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph5(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,6,2, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 4 };
            int[] v1 = {MAX_WEIGHT,0,MAX_WEIGHT, 1, MAX_WEIGHT, 7,
9, MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { 6, MAX_WEIGHT, 0,MAX_WEIGHT, 3, MAX_WEIGHT,
2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { 2,1, MAX_WEIGHT, 0, MAX_WEIGHT, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT, 3,MAX_WEIGHT, 0,
MAX_WEIGHT,2, MAX_WEIGHT, MAX_WEIGHT};
            int[] v5 = { MAX_WEIGHT, 7, MAX_WEIGHT, 6, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = { MAX_WEIGHT, 9, 2, 8, 2, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 2, MAX_WEIGHT, 0, 2 };
            int[] v8 = {4, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,2, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "办公楼";
            vertexes[1] = "塔影阁";
```

```
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
            vertexes[4] = "2 号门";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph6(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, 7,MAX_WEIGHT,1, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v1 = {7,0,MAX_WEIGHT, 1, MAX_WEIGHT, MAX_WEIGHT,
9, MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { MAX_WEIGHT, MAX_WEIGHT, 0,MAX_WEIGHT, 6, 3,
2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { 1, 1, MAX_WEIGHT, 0,2, MAX_WEIGHT, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT, 6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT,4};
            int[] v5 = { MAX_WEIGHT, MAX_WEIGHT, 3, MAX_WEIGHT,
MAX_WEIGHT, 0, 2, MAX_WEIGHT, MAX_WEIGHT };
            int[] v6 = { MAX_WEIGHT, 9, 2, 8, MAX_WEIGHT, 2, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { 2, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0, 2};
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "顾塔里宿舍区";
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
```

```
            vertexes[3] = "图书馆";
            vertexes[4] = "2 号门";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph7(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, 9,2,8, MAX_WEIGHT, MAX_WEIGHT, 2,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v1 = {9,0,MAX_WEIGHT, 1, MAX_WEIGHT,7, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { 2, MAX_WEIGHT, 0,MAX_WEIGHT, 6, MAX_WEIGHT,
3, MAX_WEIGHT, MAX_WEIGHT };
            int[] v3 = { 8, 1, MAX_WEIGHT, 0, 2, 6, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT,6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { MAX_WEIGHT,7, MAX_WEIGHT,6, MAX_WEIGHT, 0,
MAX_WEIGHT, 2, MAX_WEIGHT };
            int[] v6 = { 2, MAX_WEIGHT,3, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 0, MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT,2, MAX_WEIGHT, 0,2 };
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, 4, MAX_WEIGHT, MAX_WEIGHT,2, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "西区组团";
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
```

```
            vertexes[4] = "办公楼";
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "2 号门";
            vertexes[7] = "快递点";
            vertexes[8] = "1 号门";
        }
        private void createGraph8(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,MAX_WEIGHT,MAX_WEIGHT,
MAX_WEIGHT, 2, MAX_WEIGHT, MAX_WEIGHT, 2 };
            int[] v1 = {MAX_WEIGHT,0,MAX_WEIGHT, 1, MAX_WEIGHT,7, 9,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v2 = { MAX_WEIGHT, MAX_WEIGHT, 0,MAX_WEIGHT, 6,
MAX_WEIGHT, 2, 3, MAX_WEIGHT };
            int[] v3 = { MAX_WEIGHT, 1, MAX_WEIGHT, 0,2, 6, 8,
MAX_WEIGHT, MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT, 6,2, 0,
MAX_WEIGHT,MAX_WEIGHT, MAX_WEIGHT, 4};
            int[] v5 = { 2, 7, MAX_WEIGHT, 6, MAX_WEIGHT, 0,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT };
            int[] v6 = { MAX_WEIGHT, 9, 2, 8, MAX_WEIGHT, MAX_WEIGHT,
0, 2,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, 3, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 2, 0, MAX_WEIGHT };
            int[] v8 = { 2, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 4,
MAX_WEIGHT, MAX_WEIGHT,MAX_WEIGHT, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
            matrix[7] = v7;
            matrix[8] = v8;

            vertexes[0] = "快递点";
            vertexes[1] = "塔影阁";
            vertexes[2] = "C5 教学楼";
            vertexes[3] = "图书馆";
            vertexes[4] = "办公楼";
```

```
            vertexes[5] = "顾塔里宿舍区";
            vertexes[6] = "西区组团";
            vertexes[7] = "2 号门";
            vertexes[8] = "1 号门";
        }
        private void createGraph9(int index){
            matrix = new int[index][index];
            vertexes = new String[index];

            int[] v0 = { 0, MAX_WEIGHT,MAX_WEIGHT,MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT };
            int[] v1 = {MAX_WEIGHT,0,MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT };
            int[] v2 = { MAX_WEIGHT, MAX_WEIGHT, 0,MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT };
            int[] v3 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT };
            int[] v4 = { MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT,MAX_WEIGHT, 0, MAX_WEIGHT,MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT};
            int[] v5 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, 0, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT };
            int[] v6 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT,MAX_WEIGHT};
            int[] v7 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT, 0,
MAX_WEIGHT };
            int[] v8 = { MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT, MAX_WEIGHT, MAX_WEIGHT,
MAX_WEIGHT,MAX_WEIGHT, 0 };

            matrix[0] = v0;
            matrix[1] = v1;
            matrix[2] = v2;
            matrix[3] = v3;
            matrix[4] = v4;
            matrix[5] = v5;
            matrix[6] = v6;
```

```
        matrix[7] = v7;
        matrix[8] = v8;

        vertexes[0] = "2 号门";
        vertexes[1] = "塔影阁";
        vertexes[2] = "C5 教学楼";
        vertexes[3] = "图书馆";
        vertexes[4] = "办公楼";
        vertexes[5] = "顾塔里宿舍区";
        vertexes[6] = "西区组团";
        vertexes[7] = "快递点";
        vertexes[8] = "1 号门";
    }
    /**
     * Dijkstra 最短路径。
     *
     * vs -- 起始顶点(start vertex) 即，统计图中"顶点 vs"到其它各个顶点的最短路径。
     */
    public void dijkstra(int vs) {
        // flag[i]=true 表示"顶点 vs"到"顶点 i"的最短路径已成功获取
        boolean[] flag = new boolean[vertexes.length];
        // U 则是记录还未求出最短路径的顶点(以及该顶点到起点 s 的距离)，与 flag 配合使用,flag[i] == true 表示 U 中 i 顶点已被移除
        int[] U = new int[vertexes.length];
        // 前驱顶点数组,即，prev[i]的值是"顶点 vs"到"顶点 i"的最短路径所经历的全部顶点中，位于"顶点 i"之前的那个顶点。
        int[] prev = new int[vertexes.length];
        // S 的作用是记录已求出最短路径的顶点
        String[] S = new String[vertexes.length];

        // 步骤一：初始时，S 中只有起点 vs；U 中是除 vs 之外的顶点，并且 U 中顶点的路径是"起点 vs 到该顶点的路径"。
        for (int i = 0; i < vertexes.length; i++) {
            flag[i] = false; // 顶点 i 的最短路径还没获取到。
            U[i] = matrix[vs][i]; // 顶点 i 与顶点 vs 的初始距离为"顶点 vs"到"顶点 i"的权。也就是邻接矩阵 vs 行的数据。

            prev[i] = 0; //顶点 i 的前驱顶点为 0
        }

        // 将 vs 从 U 中“移除”（U 与 flag 配合使用）
        flag[vs] = true;
        U[vs] = 0;
```

```
        // 将 vs 顶点加入 S
        S[0] = vertexes[vs];
        // 步骤一结束
        //步骤四：重复步骤二三，直到遍历完所有顶点。
        // 遍历 vertexes.length-1 次；每次找出一个顶点的最短路径。
        int k = 0;
        for (int i = 1; i < vertexes.length; i++) {
            // 步骤二：从 U 中找出路径最短的顶点，并将其加入到 S 中（如果 vs 顶点到 x 顶点还有更短的路径的话，那么
            // 必然会有一个 y 顶点到 vs 顶点的路径比前者更短且没有加入 S 中
            // 所以，U 中路径最短顶点的路径就是该顶点的最短路径）
            // 即，在未获取最短路径的顶点中，找到离 vs 最近的顶点(k)。
            int min = MAX_WEIGHT;
            for (int j = 0; j < vertexes.length; j++) {
                if (flag[j] == false && U[j] < min) {
                    min = U[j];
                    k = j;
                }
            }

            //将 k 放入 S 中
            S[i] = vertexes[k];

            //步骤二结束


            //步骤三：更新 U 中的顶点和顶点对应的路径
            //标记"顶点 k"为已经获取到最短路径（更新 U 中的顶点，即将 k 顶点对应的 flag 标记为 true）
            flag[k] = true;

            //修正当前最短路径和前驱顶点（更新 U 中剩余顶点对应的路径）
            //即，当已经"顶点 k 的最短路径"之后，更新"未获取最短路径的顶点的最短路径和前驱顶点"。
            for (int j = 0; j < vertexes.length; j++) {
                //以 k 顶点所在位置连线其他顶点，判断其他顶点经过最短路径顶点 k 到达 vs 顶点是否小于目前的最短路径，是，更新入 U，不是，不做处理
                int tmp = (matrix[k][j] == MAX_WEIGHT ? MAX_WEIGHT : (min + matrix[k][j]));
                if (flag[j] == false && (tmp < U[j])) {
                    U[j] = tmp;
                    //更新 j 顶点的最短路径前驱顶点为 k
```

```
                    prev[j] = k;
                }
            }
            //步骤三结束
        }
        //步骤四结束

        // 打印 dijkstra 最短路径的结果
        System.out.println("起始顶点：" + vertexes[vs] +"(苏州科技大学石湖校区)");
        for (int i = 0; i < vertexes.length; i++) {
            System.out.print("最短路径（" + vertexes[vs] + "," + vertexes[i] + "):" + U[i] + "    ");

            List<String> path = new ArrayList<>();
            int j = i;
            while (true) {
                path.add(vertexes[j]);

                if (j == 0)
                    break;

                j = prev[j];
            }

            for (int x = path.size()-1; x >= 0; x--) {
                if (x == 0) {
                    System.out.println(path.get(x));
                } else {
                    System.out.print(path.get(x) + "->");
                }
            }

        }

        System.out.println("顶点放入 S 中的顺序：");
        for (int i = 0; i< vertexes.length; i++) {

            System.out.print(S[i]);

            if (i != vertexes.length-1)
                System.out.print("-->");
        }
```

```
    }

    public static void main(String[] args) {
        System.out.println("**********欢迎来到苏州科技大学石湖校区导航系统**********");
        System.out.println("苏科导航已为您规划最佳游览路线，请您查收，仔细阅读");
            ShortestPathDijkstra dij = new ShortestPathDijkstra();
            dij.createGraph(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph1(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph2(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph3(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph4(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph5(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph6(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph7(9);
            dij.dijkstra(0);
            System.out.println();
            dij.createGraph8(9);
            dij.dijkstra(0);
            System.out.println();
            System.out.println("导航结束，谢谢使用");
    }
}
```

五、心得与体会：

本次实验针对具体问题来进行需求分析、测试计划、概要设计、详细设计、测试分析等具体步骤，从中我们收获了许多。首先，在编写函数时要充分利用各种资源，其次，要更加详细的考虑实际情况，提高程序的实用性，最重要的当然是组员之间的团结协作。

程序设计的过程实际上也是加深对图及 Dijkstra 算法认识的过程，当然后者更是前者的基础，我们在实现 Dijkstra 算法前必须自己独立设计一个图，列出对应矩阵，这样方便我们算法的设计。当然我们可以通过网络学习加强理解，虽然只是看了几个关于 C 语言的数据结构网课，但是它的数组实现方式让我们从另一方面了解了 Dijkstra 算法。

通过本次实验，再次强化了我们函数调用、函数声明、全局变量、局部变量、图以及戴克斯特拉算法的应用。编程本是一件枯燥的事，但是只要我们认真的去对待，总会在过程中收获书本上学不到的东西。