基站维修工程师

时间限制：1s 空间限制：256MB 限定语言：不限

题目描述：

小王是一名基站维护工程师，负责某区域的基站维护。

某地方有n个基站（1<n<10），已知各基站之间的距离s（0<s<500），并且基站x到基站y的距离，与基站y到基站x的距离并不一定会相同。

小王从基站1出发，途径每个基站1次，然后返回基站1，需要请你为他选择一条距离最短的路线。

输入描述：

站点数n和各站点之间的距离（均为整数）。如：
3 {站点数}
0 2 1 {站点1到各站点的路程}
1 0 2 {站点2到各站点的路程}
2 1 0 {站点3到各站点的路程}

输出描述：

3
最短路程的数值

示例1

输入：

```
3
0 2 1
1 0 2
2 1 0
```

输出：

```
3
```

解题思路：

1. 将各基站路程转化为二维数组
2. 通过回溯的方法将所有可能的路线模拟一遍，找到最短的路程

```
public class Main{

    public static int min = Integer.MAX_VALUE;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        sc.nextLine();

        int[][] distance = new int[n][n];     //路程转化为二维数组
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                distance[i][j] = sc.nextInt();
            }
        }

        for(int i=1; i<distance.length; i++){
            List<Integer> stepList = new ArrayList<>();
            stepList.add(i);       //从基站 2 开始走
            handle( distance, i, stepList, distance[0][i]);
        }

        System.out.println(min);
    }

    /**
     *
     * @param ints          各基站路程的二维数组
     * @param index         达到的基站
     * @param step          路过的基站
     * @param sum           走过的路程
     */
    public static void handle(int[][] ints, int index, List<Integer> step, int sum){

        if(step.size() + 1 == ints.length){        //走完所有的基站，准备返回基站 1
            min = Math.min( min, sum + ints[index][0]);        //记得加上基站 1 的路程
        }else {
            for(int i=1; i<ints.length; i++){
                if(step.contains(i)){     //已经走过的基站不需要再走
                    continue;
```

```
                    }
                    step.add(i);       //走过的基站
                    handle(ints, i, step, sum + ints[index][i]);       //index 到达的基站，i 下次去的基站
                    step.remove(step.size()-1);         //刚走过的基站需要剔除以便进行下一个循环
                }
            }
        }
    }
```