

FloydWarshall.java

```
1 import java.util.Scanner;
2
3 public class FloydWarshall {
4     private int matrix[][];
5     private int numVertices;
6     public static final int INFINITY = 999;
7
8     public FloydWarshall(int vertices) {
9         matrix = new int[vertices + 1][vertices + 1];
10        this.numVertices = vertices;
11    }
12
13    public void floydwarshall(int mat[][]) {
14        for (int s = 1; s <= numVertices; s++) {
15            for (int d = 1; d <= numVertices; d++) {
16                matrix[s][d] = mat[s][d];
17            }
18        }
19        for (int i = 1; i <= numVertices; i++) { // i = intermediate
20            for (int s = 1; s <= numVertices; s++) {
21                for (int d = 1; d <= numVertices; d++) {
22                    if (matrix[s][i] + matrix[i][d] < matrix[s][d])
23                        matrix[s][d] = matrix[s][i] + matrix[i][d];
24                }
25            }
26        }
27
28        for (int s = 1; s <= numVertices; s++) {
29            System.out.print("\t" + s);
30        }
31        System.out.println();
32        for (int s = 1; s <= numVertices; s++) {
33            System.out.print(s + "\t");
34            for (int d = 1; d <= numVertices; d++) {
35                System.out.print(matrix[s][d] + "\t");
36            }
37            System.out.println();
38        }
39    }
40
41    public static void main(String... arg) {
```

FloydWarshall.java

```
42     int matrix[][];
43     int numVertices;
44     Scanner scan = new Scanner(System.in);
45     System.out.println("Number of vertices:");
46     numVertices = scan.nextInt();
47     matrix = new int[numVertices + 1][numVertices + 1];
48     System.out.println("Weighted matrix");
49     for (int s = 1; s <= numVertices; s++) {
50         for (int d = 1; d <= numVertices; d++) {
51             matrix[s][d] = scan.nextInt();
52             if (s == d) {
53                 matrix[s][d] = 0;
54                 continue;
55             }
56             if (matrix[s][d] == 0) {
57                 matrix[s][d] = INFINITY;
58             }
59         }
60     }
61     System.out.println("The Transitive Closure of the Graph");
62     FloydWarshall floydwarshall = new FloydWarshall
        (numVertices);
63     floydwarshall.floydwarshall(matrix);
64     scan.close();
65 }
66 }
```