```java
/* Floyd-Warshall Algorithm to find the shortest paths in weighted-directed graphs
 * n-cubed algorithm
 * Also use for Single Source Shortest Path problems when n<100
 *
 * */
import java.util.*;

class Floyd_Warshall_Alg{
    static int[][] AdjMat;
    static int[][] paths;

    public static void initFW(int n){
        for(int i = 0; i<n; i++){
            for(int j = 0; j<n; j++){
                AdjMat[i][j]=Integer.MAX_VALUE;
                if(i==j)
                    AdjMat[i][j]=0;
                paths[i][j] = -1;
            }
        }
    }
    public static void allPairsFW(int n){
        for(int k = 0; k<n; k++){
            for(int i = 0; i<n; i++){
                for(int j = 0; j<n; j++){
                    if(AdjMat[i][k]!=Integer.MAX_VALUE && AdjMat[k][j]!=Integer.MAX_VALUE){
                        if(AdjMat[i][j]> AdjMat[i][k] + AdjMat[k][j]){
                            paths[i][j] = paths[k][j];
                        }
                        AdjMat[i][j] = Math.min(AdjMat[i][j], AdjMat[i][k] + AdjMat[k][j]);
                    }
                }
            }
        }
    }
```

```
36      public static void printArr2D(int[][] arr){
37          for(int i =0; i<arr.length; i++){
38              for(int j = 0; j<arr.length; j++){
39                  System.out.print(arr[i][j] + "\t");
40              }
41              System.out.println();
42          }
43      }
44
45      public static void main(String[] args){
46          Scanner scan = new Scanner(System.in);
47          int K = scan.nextInt();
48          int V = 0;
49          int E = 0;
50          for(int i = 0; i<K; i++){
51              V = scan.nextInt();
52              E = scan.nextInt();
53              AdjMat = new int[V][V];
54              paths = new int[V][V];
55              initFW(V);
56
57              for(int j = 0; j<E; j++){
58                  int r = scan.nextInt();
59                  int c = scan.nextInt();
60                  AdjMat[r][c] = scan.nextInt();
61                  paths[r][c] = r;
62              }
63              allPairsFW(V);
64              printArr2D(AdjMat);
65              System.out.println();
66              printArr2D(paths);
67              System.out.println();
68          }
69          scan.close();
70      }
71
72  }
73
```