```
 /**
 * Excerpt taken from
 * Network.java
 * http://www.ludowaltman.nl/slm/modularity_optimizer_source.zip
 *
 * @author Ludo Waltman
 * @author Nees Jan van Eck
 * @version 1.3.1, 08/30/15
 */
import java.io.Serializable;

/**
 * Excerpt taken from
 * Network.java
 * http://www.ludowaltman.nl/slm/modularity_optimizer_source.zip
 *
 * @author Ludo Waltman
 * @author Nees Jan van Eck
 * @version 1.3.1, 08/30/15
 */

public class Network implements Serializable
{
    private static final long serialVersionUID = 1;

    protected int nNodes;
    protected int nEdges;
    protected double[] nodeWeight;
    protected int[] firstNeighborIndex;
    protected int[] neighbor;
    protected double[] edgeWeight;
    protected double totalEdgeWeightSelfLinks;

    public Network(int nNodes, double[] nodeWeight, int[][] edge, double[] edgeWeight)
    {
        double[] edgeWeight2;
        int i, j;
        int[] neighbor;

        this.nNodes = nNodes;

        nEdges = 0;
        firstNeighborIndex = new int[nNodes + 1];
        neighbor = new int[edge[0].length];
        edgeWeight2 = new double[edge[0].length];
        totalEdgeWeightSelfLinks = 0;
        i = 1;

        for (j = 0; j < edge[0].length; j++)
            if (edge[0][j] != edge[1][j])
            {
                if (edge[0][j] >= i)
                    for (; i <= edge[0][j]; i++)
                        firstNeighborIndex[i] = nEdges;
                neighbor[nEdges] = edge[1][j];
                edgeWeight2[nEdges] = (edgeWeight != null) ? edgeWeight[j] : 1;
                nEdges++;
        }
        else
            totalEdgeWeightSelfLinks += (edgeWeight != null) ? edgeWeight[j] : 1;
        for (; i <= nNodes; i++)
            firstNeighborIndex[i] = nEdges;

        this.neighbor = Arrays.copyOfRange(neighbor, 0, nEdges);
        this.edgeWeight = Arrays.copyOfRange(edgeWeight2, 0, nEdges);

        this.nodeWeight =
            (nodeWeight != null) ? (double[])nodeWeight.clone() : getTotalEdgeWeightPerNode();
    }

}
```