import java.util.ArrayList;

import java.util.HashSet;

public class Main {

public static void main(String[] args) {

Graph graph = new Graph(4);

graph.addEdge(0, 1);

graph.addEdge(0, 2);

graph.addEdge(0, 3);

graph.addEdge(1, 0);

graph.addEdge(1, 2);

graph.addEdge(2, 0);

graph.addEdge(2, 1);

graph.addEdge(0, 3); // duplicate

graph.addEdge(0, 2); // duplicate

graph.printGraph();

}

}

class Graph {

ArrayList<HashSet<Integer>> graph;

int vertices;

Graph(int vertices) {

this.vertices = vertices;

this.graph = new ArrayList<>();

for (int i = 0; i < vertices; i++) {

graph.add(new HashSet<>());

}

}

void addEdge(int src, int dest) {

if (isValid(src, dest)) {

graph.get(src).add(dest);

graph.get(dest).add(src);

} else {

System.err.println("Invalid source or destination");

}

}

void removeEdge(int src, int dest) {

if (isValid(src, dest)) {

graph.get(src).remove(dest);

graph.get(dest).remove(src);

} else {

System.err.println("Invalid source or destination");

}

}

public HashSet<Integer> neighbour(int node) {

return graph.get(node);

}

boolean isHavingEdge(int src, int dest) {

if (isValid(src, dest)) {

return graph.get(src).contains(dest);

}

System.out.println("Invalid source or destination");

return false;

}

boolean isValid(int src, int dest) {

return src >= 0 && dest >= 0 && src < vertices && dest < vertices && src != dest;

}

void printGraph() {

for (int i = 0; i < vertices; i++) {

System.out.print("Node " + i + " connects to: ");

for (int var : graph.get(i)) {

System.out.print(var + " ");

}

System.out.println();

}

}

}