```java
/*
    Time complexity: O(N^2)
    Space complexity: O(N^2)
    where N is the number of vertex in the graph
*/


public class Solution {


        public static int numConnected(int[][] edges, int n) {
                boolean[] visited = new boolean[n];
                int count = 0;
                for (int i = 0; i < n; i++) {
                        if (visited[i] == false) {
                                dfs(edges, i, visited, n);
                                count++;
                        }
                }
                return count;
        }

        private static void dfs(int[][] edges, int v1, boolean[] visited, int n) {
                visited[v1] = true;

                for (int i = 0; i < n; i++) {
                        if (visited[i] == false && edges[v1][i] == 1) {
                                dfs(edges, i, visited, n);
                        }
                }
        }

}
```