For each of the two problems below, you should may reuse code from the textbook. You may not copy code from the Internet or other sources.

1. **Social network connectivity.** Given a social network containing ***n*** members and a log file containing ***m*** timestamps at which times pairs of members formed friendships, write a java program to determine the earliest time at which all members are connected (i.e., every member is a friend of a friend of a friend ... of a friend). Assume that the log file is sorted by timestamp and that friendship is an equivalence relation. The running time of the algorithm in your program should be ***m*log *n*** or better and use extra space proportional to ***n***. (If you wish, you may read the data from the user instead of a file but it will be easier to test if you use a file)
2. **Union-find with special element**. Add a method find() to the union-find data type so that find(i) returns the largest element in the connected component containing i. The operations, union(), connected(), and find() should all take logarithmic time or better. For example, if one of the connected components is {1,2,6,9}, then the find() method should return 9 for each of the four elements in the component.

Is the following implementation of union() for quick-find correct? If not, give an example where it is not correct.

public void union(int p, int q)

{

if (connected(p, q))

return;

for (int i = 0; i < id.length; i++)

if (id[i] == id[p])

id[i] = id[q];

count--;

}