**Preface**

Before getting into anything, it’s important to note that you are not expected to know how to do this off the top of your head. Let Google be your guide! No one we have interviewed could do this whole challenge without consulting external sources. Not even us.

Here’s a hint: there is a basic, foundational algorithm that can be used to solve this problem. You should not invent anything new as this algorithm has been widely used for over 50 years (though not intended for student gossip analysis). Pseudocode can be found on Wikipedia – where you can also learn that the algorithm’s late author had a deep dislike of the GOTO statement. You can undoubtedly find actual source code on the web. If you get stuck, we can let you know the specific name of this algorithm.

Everything here should be useful knowledge for you to have going forward, so we hope that you regard this as a learning experience for your own professional development.



**Background**

This challenge is about gossip. 662 high school students are spreading juicy bits of social news among each other. Your challenge is to determine how long it will take, in seconds, for the “Victim” of each rumor to hear the news after it is started by the “Gossiper.”

A few factors add complexity to this challenge:

· Each student (“Talker”) only talks to their closest and most trusted friends (“Listener”). Talkers can have anywhere between 1 and 10 trusted Listeners with whom they share their secrets. Thus, in order for a rumor to reach the Victim, the rumor will most likely have to propagate across the network of friend pairs (the Victim will probably not hear it directly from the Gossiper).

· Every Talker/Listener pair has conversations of different lengths. The shortest conversation is only 8 seconds and the longest is over 45 minutes.

· Conversation lengths between the same two friends differ depending on which one is the Talker and which is the Listener.

**Data**

The attached file contains two tabs:

· Gossip Chain – list of Talker/Listener pairs and their Chat Time in seconds

· Inputs & Outputs – List of rumors for which to determine the travel time based on who is specified as the Gossiper and who is the Victim

**Answer Format**

Record your responses on the Inputs & Outputs tab. Assume this program will be implemented as a web application. It accepts and sends JSON strings as its inputs and outputs. Thus, each rumor should be encoded as a JSON string (using the specifications in cell H4) and answers should follow the JSON specification in cell H5.

We’ll also review your code submission.

**Code Guidelines**

Your program should:

1. accept a set of JSON strings as inputs and produce a set of JSON strings as outputs

2. process the set of inputs on multiple concurrent threads (how you divide up the work among threads is up to you)

3. use a test suite (jUnit, for example, if using Java) to ensure that the program behaves as desired (include your tests with your codesubmission)

In addition, provide a *brief* plan for deployment of this application to a secure, cloud-based environment. Assume Git is being used for version control and the repository can be cloned to both local and remote environments, but that we do not have a cloud server running yet – setting that up would be your job. Include enough detail that we’re confident you could execute this task if needed. Names of specific platforms, for instance, would be useful. But don’t worry about writing a step-by-step tutorial on the subject.

**Judging**

You will be scored on:

1. accuracy of results

2. readability of code

3. efficiency of algorithms and data structures used

4. thoroughness of unit tests

5. precision and validity of proposed deployment plan

**Good Luck!**