1. This lab uses a vector clock (Links to an external site.) algorithm to create unique message IDs based on a sequence number. Could we replace the sequence number with a timestamp? What are the advantages and disadvantages of such an approach?

   You could replace the sequence number with a timestamp. The advantage of that is you would still be able to distinguish one message from another because the times are different between messages and now you have a context of time, a disadvantage is you lose the ability to count how many messages a node has created and received.

2. Are the temperature messages in order? Why or why not? If not, what could you do to fix this?

   The temperature messages were in order of how a node received them. They are this way by default. We could change that and run a sorting algorithm whenever a new message was received.

3. How did you avoid looping (sending messages back to someone who already has it)? Why was the unique ID helpful?

   I avoided looping by determining my neighbor based on whether or not they needed something that I knew (speaking as a node). The unique ID is helpful as it allows us to keep track of subscriptions in the message ID as well as a way to reference the smart tracker and see how many messages from each node a given node has.

4. The propagation algorithm sleeps for `n` seconds between each iteration. What are the trade-offs between a low and high value for `n`.

   Low value means the information propagates faster but it makes your system work harder, especially if there are many nodes. A slower period will propagate data slower but in situations where data doesn't change often it doesn't matter too much.

5. Did new messages eventually end on all the nodes that were connected? Were the messages displayed in the same order on each node? Why or why not?

   Yes, all new messages eventually made it to all nodes that were connected. This is due to each node noticing what their neighbors are missing and sending it off. The messages were not displayed in the same order on each node. It depends upon what messages are being sent in a single 'batch'. Those nodes will always have the same ordering as they are sorted. If messages are sent piecemeal to all nodes they will have the some ordering also except for when a new node is added after messages have already been sent.

6. Why does temporarily disconnecting a node from the network not result in permanent gaps in the messages seen at that node?

   It is because when the node is reconnected it's neighbors are able to see what it is missing and send all the messages. This is due to the fact that we have seen messages and have a missing message comparison.

7. Describe, in a paragraph or two, how you could use the basic scheme implemented here to add failure detection to the system using a reachability table.

   With a graph that is weakly connected using the gossip protocol there is a way for information to travel throughout the entirety of the nodes. This allows us to know that all nodes *should* be receiving all data. If a node recognizes that a neighbor is repeatedly not updating its data even after rumors are being sent, then we can assume that node is failing in some way. Counters could be kept for each neighbor to count how many failed attempts there were to update the neighbors' data. With a given threshold you would be able to notify a parent system or even the same system that a given node was failing.

   Not entirely sure what you want when you say "using a reachability table"