Project-based Interview: Infection

The goal of this project is to give you an opportunity to code on the sort of problem that we would actually tackle here at Khan Academy.

Feel free to put up the project code wherever you wish. If you already have private github repos available to you, feel free to share with us that way and give access to the user **ka-interview**. You can also use a public repo. Sending a zip file is fine too. Please under no circumstances pay for a new private repo just to send us your work!

Don't spend more than 12 hours on this project - you do not need to complete everything. Just do what you can in that time and if you have thoughts on the parts that you didn't complete, you can share those in a README or in our follow-up conversation.

If you ever have any questions, please feel free to send a message and we'll try to help you as best as we can!

See this blog post for more background on our project-based interviews.

Total Infection

When rolling out big new features, we like to enable them slowly, starting with just the KA team, then a handful of users, then some more users, and so on, until we've ramped up to 100%. This insulates the majority of users from bad bugs that crop up early in the life of a feature.

This strategy can cause problems somewhat unique to our user base. It's not uncommon for a classroom of students to be using the site together, so it would be confusing to show half of them a completely different version of the site. Children are not software engineers and often have a poor understanding of deployment and a/b testing, so inconsistent colors, layout, and interactions effectively mean the site is broken.

Ideally we would like every user in any given classroom to be using the same version of the site. Enter "infections". We can use the heuristic that each teacher-student pair should be on the same version of the site. So if A coaches B and we want to give A a new feature, then B should also get the new feature. Note that infections are transitive - if B coaches C, then C should get the new feature as well. Also, infections are transferred by both the "coaches" and "is coached by" relations.

First, model users (one attribute of a user is the version of the site they see) and the coaching relations between them. A user can coach any number of *other* users. You don't need to worry about handling self-coaching relationships.

Now implement the infection algorithm. Starting from any given user, the entire connected component of the coaching graph containing that user should become infected.

Limited Infection

The problem with infection is lack of control over the number of users that eventually become infected. Starting an infection could cause only that person to become infected or at the opposite (unrealistic) extreme it could cause all users to become infected.

We would like to be able to infect close to a given number of users. Ideally we'd like a coach and all of their students to either have a feature or not. However, that might not always be possible.

Implement a procedure for limited infection. You will not be penalized for interpreting the specification as you see fit. There are many design choices and tradeoffs, so be prepared to justify your decisions.

Summary

Write `total_infection` and `limited_infection` in your preferred language. Provide tests and instructions for running them.

Optionally do one of the following:

- create a visualization of the relations between users and the infection's spread
- write a version of `limited_infection` that infects *exactly* the number of users specified and fails if that's not possible (this can be (really) slow)
- make up your own enhancement!