Part 7 - Questions (and your Answers)
Acme Corporation isn't just a few .py files. If you want to grow in your career here, you'll have to answer the following:

Q1: What, in your opinion, is an important part of code reviews? That is, what is something you pay attention to when you review code, and that you appreciate when others do the same for your code?

As it has been said: Code is written once but read many times. Readability, maintainability, modifiability (without being broken), and compatibility are among the many important reasons for code reviews. It is possibly that underlying code review is the topic of best practice. What habits and methods, what standards and policies, produce "good code" most of the time? From variable naming to checking pep8 compliance, these could be afterthoughts added to code when the code is done, but, like debugging, waiting until the end, while ~possible, is strongly against best practice though it may produce a product that appears to reflect good practice. In this way, a code review should peer deeper than the finished product in a given case and look at how that was produced. How many people looked it over? How was it checked? etc.

Q2: We have an awful lot of computers here, and it gets pretty confusing with slightly different things running on all of them. How could containers help us improve this situation?

A: Google's' Colab may provide an accessible and tangible example of the advantages offered by containers. Containers such as Docker allow an environment of both processing and packages and dependencies that can be accessed over a network (unlike local virtual machines which are only locally available). This online-access allows people with a variety of front-end-connecting-devices (from phones to tablets (perhaps watches and wearables) to various operating systems of netbooks, laptops desktops, servers, mini's, mainframes etc.) to utilize the environment as if it were, often thinking that it is, simply a program running locally for themselves (it is hard not to fret over how much local RAM is being use by colab when it begins a costly task). Managing a container environment is very feasible, whereas managing all user environments is frighteningly prohibitive. Depending on how many "an awful lot of" means, Docker may not be the right solution and something such as kubernetes may be a better fit. Containers work by having a configurable operating system entirely contained in the accessible environment, so needs of specific uses can often be accomodated.

Geoffrey Gordon Ashbrook DS8