[SQUEAKING]

[RUSTLING]

[CLICKING]

MICHAEL SCOTT ASATO CUTHBERT: Hello, I want to give a sense of how some problem sets are going to work or the majority of the time. Generally speaking, you'll be given some instructions on what to do like, write a routine that takes a string representing an enharmonic with minus sign for flats-- we'll talk about that in a bit --and number sign for sharps and return a new string that is an enharmonic-- you're not allowed to change it, but I can here. That is an enharmonic of the pitch. You will be given valid input only. I will always tell if you have to check for invalid input. If there's more than one valid enharmonic, you can give any enharmonic up to double flats or sharps.

Then below, you're going to be given the name of the routine that you're going to need to write, along with usually the name of a variable that you should use and the type, if you haven't looked at Python typing, this can be really exciting, and what's output.

At this point, you need to define the routine. Quite often, you can take some of the language from above but make it in your own words and to see what happens. But put it in the active voice, takes in a string representing a pitch-- sorry, I should say representing a pitch earlier --and returns a valid enharmonic for it.

If there's specific things that constrain what the valid input might be, I'll tend to write it here, or you should be writing it in. We write for humans to read, not just for computers to process. So it's always going to be important that your code be documented and that your variable names be things that humans can understand.

OK, so then you can go ahead and write your code. And if you're a not-so-great programmer, maybe you'll start with if enharmonis_in is E flat-- we're going to use minus signs for flat --return D sharp, otherwise, return C sharp. And at least maybe it'll get it right sometimes, great.

Now, I will sometimes give you one or two test cases. So maybe I'll say, assert flip_enharmonic E flat equals G sharp. And you see that that works. Maybe, if I'm very nice, I'll give you two of them. Up, and you can see that doesn't work.

I am not going to give you all of the interesting and fun test cases. Part of the thought process of this class, part of why you get Humanities credit, is that you need to be thinking about, well, what are the weird cases? OK, I can put in if it's this, it's E flat. And if it's D sharp, it's E flat, and so on. Oh, but maybe I need to make sure that this works. Or maybe I need to make sure that that works, or at least that it's a valued answer, huh.

And so part of what you need to do, is be thinking about how do we write test code, what are the test cases that Professor Cuthbert is going to be putting in, such that it's going to challenge how I write? You'll probably need to rewrite your code as you go. Great, let's get started.