[SQUEAKING]

[RUSTLING]

[CLICKING]

**PROFESSOR:** Hi, OpenCourseWare learners. We've gotten to the part of the first lecture where the technical difficulties made it a little too hard to follow what was happening there. So I'm going to rerecord what we all did as the opening of the class. To begin, go ahead and go to the URL on the screen right now. This is a project from Google called Colab that allows you to run Python scripts, including Music21, in your own browser, without installing anything on your computer. So go ahead and click this Play button. You're going to see a warning that Google didn't write this notebook. I did.

I think it's pretty safe. What it does is it installs the latest version of Music21, and MuseScore, and some other software needed to run some of the tools we're going to be using, without installing Python on your computer. Google already has a copy of Music21, but this just brings you to the latest version. And go ahead and pause, and we'll come back after it's all installed.

Welcome back. I hope that these cells aren't running anymore. So we can start coming in. If you've never used Jupyter Notebook or another online collaborative environment, the important thing that you're going to need is to be clicking into these cells. The gray ones are where you type. And the ones with the white or black background are the ones where output's coming out. In order to run a cell, you just type Shift and Return or Shift and Enter. So what we're going to do is I'm just going to click into this cell and hit Shift Enter. And this will make sure that Music21 actually works.

Sometimes the first time you show a score it doesn't work as well. And on some browsers, this will work. You can listen to a score. And let's see if this works or not.

[TONE]

There we go. And now we're ready to actually begin. We're not going to generate with AI today. We'll write our own scores. So if you're with me, we're going to start anything where you're doing a Python session by importing the code that's useful. And for this one, it's the Music21 toolkit. So we're going to import everything from it. If you're somebody who's done a lot of Python and cares about things like polluting your namespace, you can do individual modules. But I'm always going to do it as if we have all the parts of Music21 at your fingertips.

So we're going to load a chorale by Johann Sebastian Bach. It's his shortest chorale, called BWV66.6. So first, I'll create a variable, call it Bach. And we'll use the corpus tool from Music21 and its parse function to load in bach/bwv66.6. And I'll hit Shift Enter. And it's assigned to Bach. And we can call the method show on it to see the score of that. Yep, looks like a chorale by Bach. And you can see that has soprano, alto, tenor, bass.

And we can get the alto part by saying bach.parts, square brackets, open, alto. And we'll just show that. Yes. Looks like the alto part. Now let's get the soprano part and assign it to a variable sopr for soprano. And we'll just do that. Instead, this time we'll get it by doing the part number. And so it's the first part. And so in Python, that means it's part 0, parts 0. And one of the things that we do in Music21 is when something is in a programmer's rule, we follow the conventions of programming, where everything starts with 0.

And when we're in musician space, such as talking about measures, we don't subtract 1 from the measure number or anything like that. So if we wanted to just see measures 3 through 5, inclusive of the soprano part, we could do something like that. Oops. Or we could type the word measures properly. And then we'll get that measure. Just gets a single measure. This one got that. Great.

And in fact, we'll use the measure thing now. And we'll get in measure five, the zeroth, first, second note, the one with the fermata on it. And so we'll call it C because it's a C. Actually, it's a C sharp. But we'll see why not. And so we'll get notes number 2. If we called it C sharp, then everything after this would be a comment. So that doesn't work. So we'll just call it C. And now what we can do is manipulate it, change its style, its color to red. And now when I show the soprano part, because they're all linked to each other, we should see-- yep, there's a note that's in red.

So that's how we can manipulate a score. And now let's see how we can do some analytical functions on things. So let's do Bach to analyze. And we'll analyze the key of the piece. And it says it returns a key object that says it's in F sharp minor. And one of the things I really want to emphasize is always go back and do the smell test. Is this true, or does it look like baloney? And we can see, well, begins on A, but it does have three sharps. And at the end, yep, there is an F sharp in the bass. Though, it looks kind of like F major, but-- OK, here some F minor.

OK, it kind of passes the smell tests. And we'll see why that's so important in just a little bit. But it did seem like it was beginning in A. So we can just analyze a portion of that. So we'll do bach measures 0, because we want the pickup measure, and 2. And we'll analyze the key of just that passage. Ah, does seem to start in A major.

Let's do something that seems pretty simple. Let's count the number of notes in the piece. So we can just do len bach.notes. That just gets all the notes of the piece. In another language, it would be bach.notes.length. I don't know how many notes are in the piece, but I'm pretty sure it's more than 0. And the reason for this is that Music21 and most ways of conceiving music on the computer don't think of the score as having notes. The score has parts. The parts have measures. And the measures have notes.

So what we could do is get the length of bach parts 1, measure 2, dot notes, and see, OK, that passes the smell test. So what we can do instead is we can use something called recurse. So we'll get the bach.recurse. That's a function, or a method. And we get the notes on that. And what recurse says is, open up every box. You get to a part. Open it up. Look inside for measures. Open it up and keep going until you find notes. And 165 notes. That seems to pass the smell test to me.

Right now we've been working with this as a series of parts that are not connected to each other. But of course, there are chords in there also. But if we do bach.-- and here's another way you can say recurse. I want to recurse and get everything that's a chord.

There are no chords in this piece. So we have to create some chords using the chordify method. So we'll say chor, for chords, equals bach.chordify. And now if we show the chordify method-- And we can see that some of these notes are way too low to be read properly. And they have duplicated notes and things. And over the course of the semester, we'll find ways that we can make this a lot more representable, including things like getting rid of passing tones and other things that disturb looking at the chords. But for now, we can look at chor for the chords, the number of chords in that, and see that, yep, that looks like a reasonable number.

So I hope that gives you a taste of some of what we're going to be doing over the course of the semester of this class. Now, we've gone very fast through a bunch of things, but don't worry, because we're going to lock all those things up. Choridfy is going to be locked up. And now Analyze Key is going to be locked up. And what does that mean? That means that it won't be available to be used in the rest of the class until it's unlocked again.

The way this class is going to work is all of these analytical tools, all these ways of conceiving a score, will begin by being locked. And then you'll do an assignment either as a problem set or as an in-class assignment, where we'll figure out, How does this work? What are the ways of doing this? And then once you've designed your own algorithm for doing these things, you can either continue to use your own algorithm, or you will have unlocked the Music21 version of the algorithm. And from then on, you can use that instead. So we're going to just be building up knowledge little by little until we have everything.

One of the things that as an OpenCourseWare learner you won't have access to is the extremely valuable things that happened outside of class. Yes, coming to office hours, but mostly with students helping each other, giving each other inspiration for projects they could do, and helping them through some of the more difficult parts of the class. So I encourage you, if you can find somebody else online who's also taking the course, or somebody who has taken it in the past, and get to know them, that will give you a lot more of the experience that the students had at MIT when they were taking the class in person. I'm so glad that you've chosen to take your time to learn about the fascinating world of music theory and music analysis with a computer. And I look forward to seeing you again from time to time in this course.