ERIK DEMAINE: I mean, there are a couple of different approaches to teaching this class. One is to make it fun. I really like to make material as engaging and as accessible as possible. So, especially if you have relatively little experience in algorithms-- you've just taken 6006-- and some of these algorithms get really complicated. And they're hard to understand and it makes you wonder, how did they ever come up with that?

So I like to talk about that. In some of the lectures, I'll kind of give an idea of how they might have gotten there. It's like, oh we could start with this really simple step. It's really inefficient, but at least it solves a problem. But then we have this extra thing and this extra thing, and eventually you get to the really fast solution, so it's less of the mystery of how you got there, or how the original researchers got there.

And I also try to remind students, especially in the harder topics, that this is not easy stuff and you should think about it a little more. Do the problem sets and you'll learn it-- figure it out at a deeper level. But a lot of this material is hard, and I think being honest about that makes it more reasonable. Students don't feel bad for thinking that it's hard, because the first time you learn it, it is hard.

But the whole algorithmic way of thinking of setting up a problem-- what should the input and the output be-- and then thinking about all different solutions for how to fill in the middle I think is a really powerful perspective, and it can shape pretty much your entire life, but especially if you're doing computer science. It's a really useful way of thinking, and I've gotten lots of letters later from students where they're doing some job, some computer thing and they ran this problem and then they discover-- and then they looked back at their notes and said, oh, this is the right data structure I should use now. And they solve it, and they tell me about it. And it's kind of exciting to see years later the way of thinking and lots of the solutions and techniques you see in this class really are useful out there in the world, so it's a useful class to take.

It's a relatively hard class. The exams are challenging. We try to make them not super challenging, but it's hard, especially whenever creativity is involved. It's hard to do that live in an exam. So students may not perform super well my class, but still they get the idea that this is a way to approach problems, and they understand some of the techniques.

And over time, they can fill in whatever holes they left behind, and say, you know, it seems like

a problem where dynamic programming would be really useful. It's time I really learned dynamic programming. Dynamic programming is one that we actually teach both in 006 and 046, because it's really conceptually weird. It's actually really easy once you understand it, but getting to that point of understanding is really tough.

So we spend a lot of time on it so that the ideas-- and more exposure you get to it, the clearer it will become. I think by the end of 6046, they have a pretty good idea. But if not, years later, if they come back it, I'm sure it'll be a lot easier this third time around. So a lot of these concepts are-- I mean, the ideas are actually really simple. They're just hard to shift your mind into that kind of mindset, but once you do, things become really clear. So hope is, by the end of the semester, they get there.