Hi, welcome to this course in Quantitative Formal Modeling and Worst-Case Performance Analysis.
This is the high-tech campus in Eindhoven. And it's also called the Dutch Silicon Valley.
That's because there's a lot of engineering companies all clustered together here.
There's IBM, Intel, Philips of course and just a few kilometers down the road there is ASML.
So basically this area is what we prepare our students for. This is where most of the students end up. And that's why I thought this would be a good place to start the course. Let's have a look around.

My name is Pieter Cuijpers and I'm a teacher at the Eindhoven University of Technology in the Computer Science department. And as a teacher, I'm always wondering what is the best way to prepare students for work at these high-tech companies. Of course, you need some skills, right? You need to know how to do computer programming if you want to work in a software company. You need to know how to do electrical engineer, how to design an electrical circuit if you need to work at a company like that. You need to know strength calculations if you want to be in construction.

But when I talk to my friends who are working here, that's not what they tell me is the most important thing they learned at university. If you ask these companies what competence they are really looking for, and remember that at university, we are preparing students to become system architects or head engineers. Then the most important thing that they're looking for is abstract thinking.

You need to know how to model. You need to know how to separate the important details from the unimportant details. So that is basically what I would like to teach you most. But how? How can I do that? Well, as an example, have you ever thought of why a boxer skips rope? I mean you don't bring a rope to a boxing match so why would you skip rope and jump like that all the time? Of course, that is because rope skipping is an extreme form of footwork, and in that same way I'm looking for an extreme form of abstract thinking, and the most extreme form that I can think of is mathematics.

Mathematics is really the most abstract thing there is around. And the type of mathematics that I like to use is called formal methods that's been developed by computer theoretical scientists. And I think formal methods for preparing you for modelling exercises is the best training that you can get.
The best extreme way of really diving into abstraction and modeling and skip the cutting the unimportant details from the important details.

And I admit, formal method is also a bit like rope skipping in the sense that in a real practical engineering situation, you may not always bring formal methods to the fight. But as I said, it's the best training that you can get and it's not always altogether useless. I also have been in discussion over here where I brought the formal methods in and people get annoyed. But they still like the results in the end. So what can you expect from this course? And what do I expect from you as student? Well, of course, as you may have guessed from the title this is going to be about performance analysis.

So I'm going to teach you a basic skill after all, but I'm going to look at it in an abstract way as possible. And that way you also learn how to model. Performance analysis is about calculating how fast a system is, for example. And we are going to look at an image processing pipeline for an X-ray machine and see how many images can be processed per second. We're going to look at a wireless LAN. And see how long it takes between a message coming in to a wireless LAN chip and an acknowledgement being sent, for example. And we're looking at a printer and see how much space you need for buffering paper in the printer.

Now, in the first week of this course, we're going to do this by just drawing pictures of these kinds of systems. So we'll focus at the actual modeling part. Then, in the second week, that's where the real abstraction begins, we're going to do the same thing as in the first week, but as mathematical as possible. And I'm going to use my own way of doing formal methods, so even if you've had a course in formal methods before, there will be something new in the second week. In the third week of the course, we're going to do actual computations. That's where you can learn how to do computations a bit.
Well, it's windy out here so maybe it's time to get back to the studio.