Pīkau Name: Computational Thinking - the International Perspective

Video Name: What does "thinking like a computer scientist" mean?

Presenter: Professor Tim Bell

Hopefully with the example with the phone book I've convinced you that we could search a million pages by just looking at twenty of them. That's a fast algorithm for doing quite a big task, and of course there are companies that make a lot of money with that algorithm or with related algorithms, searching billions of web pages in a very short time. A search engine does depend on those very fast algorithms; in fact, they search billions of pages very, very quickly - in a fraction of a second - because it's not humans actually doing the processing, it's a computer. But it's not just a matter of having a fast algorithm, because you want a website that people like to use. It has to have a good interface, and if you think of a search engine it's not how *complicated* the interface is, sometimes it's how *simple* it is. A search engine is often just one box where you type in what you are searching for, and a button where you ask it to do the search. A very simple interface, and yet the most popular website in the world is basically that interface. We know how to design the fast algorithm - very important because humans don't like slow algorithms; you don't want to wait for two weeks for it to come back with a result – but it's the simple interface that means you just go to the site and it's kind of obvious what to do; but then there's other aspects to it. It needs to be secure; people don't want other people to know what they have been searching for. Imagine if all the searches that you did in the last couple of weeks were put up on the staff notice board! You probably don't mind, but people like their privacy and so how do you keep it secure to stop other people from finding out? Or in particular, to stop other people from manipulating what you get to see because they know what you like to search for? And then of course it's even better if you don't have to type in what you're searching for - the search engine predicts it, and most search engines do this. You start typing and it goes 'Oh that word that you started typing, everyone's been searching for that this week so I'll predict that's what you're going to be typing in'. Or something more specific to you; if you're starting to type something, it goes 'well I've started to understand what you are interested in so I think you're probably looking for this phrase here'. You try that out, try searching for different things and see if it's suggesting things specific to you. That is starting to get into artificial intelligence, trying to understand the user and model what they do, and make useful predictions. That's another part of computer science.

One other issue that we need to worry about when you are designing something that people are going to use is scalability. Suppose it works for a thousand people using it at the same time today - that's great, but what if it goes viral and suddenly a million people hit your software or your website or whatever it is you've designed? You want that to work, otherwise those million people who have a bad experience will never come back again. Scalability is another thing that we have to worry about.

We are starting to see that computer science covers a number of different ideas. Keeping things secure, using artificial intelligence to help the user, having fast algorithms – and that's just scraping the surface of the many things that we worry about in order to make sure that whatever you've produced, whatever the software is, it's really good for the user. Once you've got these ideas about how to do it, we need to program them up. Programming is just a tool for implementing ideas. It's not *just* a tool, it's a very powerful tool, and it requires quite a bit of skill to use it really well, but programming on its own isn't enough to create software that people love to use. Computer science is the area that gives programmers the inside knowledge to make their software fast, efficient, reliable, secure, usable, intelligent, scalable and even delightful! We need to be able to look at all of those areas so that as students learn to program they're actually programming something that people will want to use.