Pīkau Name: First steps in programming: CT PO1 (EMP05)

Video Name: Decomposition and programming

Presenter: Tim Bell

Progress outcome 1 says that students use “decomposition skills”. We’ll have a look at how that comes up in the exercise we’ve been looking at.

At a simple level, the students have taken a very high level task I’ve given them - get the bot and to move over to where the pineapple is - and they’ve decomposed it or converted it into a whole lot of small steps: move forward, turn left, or turn right. I gave them some very limited instructions they could work with. That decomposition, breaking down what you generally want to do into very specific steps, is a very fundamental skill in programming.

So to look at decomposition from a different angle I’m going to use this small grid here to demonstrate some of the ideas. Normally you’d do this on a large grid with children, but I’m just using this small one to demonstrate it to you.

Here I have my turtle trying to get to the pineapple, but there are these slightly intimidating ducks in the way. The turtle needs to avoid the ducks to get to there. Now I could write a program that would be quite long to get it there, using forward, left, and right, but just to make life easier for myself I’m going to decompose that program into two parts by putting this imaginary way-point along the way and say, well, let’s write a program to get the turtle to the ‘x’, and then from the ‘x’ to the pineapple. So there are two shorter programs to write. I will work on the program to get to the ‘x’ first. Okay, I think my program’s written to get to the yellow square, to the ‘x’. I will test it. We go forward, right, forward, forward, forward, forward, whoops! Now my program has a bug, I’ve actually run it off the square. When I tested it I noticed that that ‘forward’ shouldn’t be there. Now, I’ve modified my program, I’ve debugged it. Let’s see if I’ve got it right this time.

Okay, I’ve got this part of the program going, now I’m going to write the part of the program that gets from here to the pineapple. I will test that part of the program: forward, left, forward, right, forward, forward, left, forward. I’ve got to the pineapple.

I’ve decomposed my problem to get from here to the pineapple. I’ve got this shorter program, and this shorter program here. Now, I’m going to put them together as one big program and I’m

fairly confident it will work. And my complete program works. The good thing is I've decomposed it into two shorter programs that were easier to write and easier to test.

Decomposition is something that happens in practice with programming a lot. First of all, at a general level, you are decomposing a human problem that someone has brought you into the very small steps that a computer program can actually do - into commands that are available to you in that programming language. But also, a lot of programs that people develop are very long, they might be thousands of lines of code, or even millions of lines of code. One person can't write that in a reasonable amount of time. What happens is you have a team of people, or even many teams, working on a large software system. They decompose it into different parts. Someone might work on the input, someone might work on some of the calculations, but you break it up into parts, and each team, or each individual within a team, can work on those parts. They put it all back together and hopefully it works. There's quite a bit of testing involved in making sure it works, and of course a lot of communication involved to make sure that what they go off to do is going to fit together when they put it back together. You can see that this relatively simple exercise that we did of just moving things around, especially if you did it physically with kids moving around, the ideas they are encountering there, are actually the fundamental ideas that are going to serve them well as they look at larger programs, and as they look at programming on devices.