Pīkau Name: Communicating well when programming (CT P0 5)

Video Name: Summary (EMP10-8)

Presenters: Tim Bell and Joanne Roberts

Now we've looked at most of the elements of Progress Outcome 5. Progress Outcome 5 is a significant point because it's the point that's expected round about the end of Year 10. That's the point at which students are looking at what they might do in NCEA and which direction they might go in. So this is the foundational material.

The first point here as usual is talking about **authentic contexts and taking account of end-users**. By now we hope students have really got to a mind set where digital devices are all about the end-user and creating things for real people in real situations which in turn will sensitise them to the idea that the digital devices they are using should be looked at critically. Do they really work for real end-users?

It says that they can **independently decompose the problems into algorithms**. So independently they are not getting so much support. They can look at what a program is needed for and think about how it might be implemented. Bear in mind that once the curriculum settles in they'll have had several years of programming behind them so they will be developing some independence at this point. Once they've designed what a program is going to do, once they've **designed an algorithm**, they can **create a program**. That program now has the full range of elements that Alan Turing would recognise as being everything a computer can do. That is it has **inputs**, it has **outputs**, it can have **sequences** of commands, it can do **selection** (the 'if' statement choosing between operations). The selection here at this level involves **comparative operations** (<, >, =) but also **logical operators**. They are the 'and' operation, the 'or' and the 'not'. So we can require two things to be true like the 'age must be more than 12 and less than 18' to qualify for a discount or something. The 'and' in there means that they both have to be true. There are different ways of combining the comparative operators with the logical operators to get quite tricky conditions.

It also says that they will deal with **variables of different data types**. We've looked at the main ones which is 'distinction between text and numbers'. Text (often called 'strings') and numbers (often subdivided into typically 'integers' and 'floating point' numbers). Integers are a bit better behaved and easier to work with. Floating point numbers can represent all sorts of different kinds of values. It then says that they can **determine when to use different types of control structures**. That ties in with the independence that there are all these different control structures around ('if' statements and 'while' statements and 'repeat' statements and so on), which one should you use in which situation.

Then we get onto the communication aspect. **Students document their programs**. Bear in mind that that may just be some comments in the programs that explain what they are doing but it also applies to things like choosing very good names for their variables so it's very clear what the program is meant to be doing. They may write some independent documentation explaining it or talking about how they've designed it as well. Then it says that they've used an **organised approach for testing and debugging**. They've tried out many values. They've tried to find bugs, tried to test whether bugs exist and then track those bugs down.

The next part we haven't looked at yet. They try to **understand how computers store more complex types of data using binary digits**. We've talked about how there is text and there are numbers and so on. We will have a separate pīkau explaining how that relates to the binary digits. In the end the binary digits, that's the thing that makes digital devices digital. It's when we are manipulating those things that we are working with the power of it being digital.

The final part here talks about **human-computer interaction**. It talks about how students can develop programs that take that into account and in particular these things called **heuristics.** We haven't covered that yet but just as a heads-up heuristics are just 'rules of thumb' for how to tell if you've written a program that is easy to use or not. That again is for another day but at this point you can see that students looking at the whole progress outcome have got a pretty deep understanding of what this thing called computer programming, what these digital devices are capable of. Also they've been empowered to actually create programs that control these devices. Programs that bring them to life.