Digital Technology Curriculum

PLD - 2019

Who am I?

- Victoria Macann
- Work for Learning Architects
- MOE accredited
- Science & Biology teacher
- MEd in Digital Education
- Apple Professional Learning Specialist

How will this PLD run?

- 80 hours
- Allocated time to have out of class support
- Learning content and setting inquiry linked to school-wide goals
- Followed with in-class support

Questionnaire

https://docs.google.com/forms/d/e/1FAIpQLSfqUGpJ4M8Ra1LbZn6fR1cm-MpSc1nx4ehMkF60G2M18U-UHg/viewform?usp=sf_link

Curriculum Document

In shared Google drive folder

WHY do we need the new curriculum?

Some Ideas about that

Reality - what's here.

World is built around it - now it is necessary

Engineering/robotics/Design/programmers/wifi/infrastructure.

WHY do we need Computational Thinking?

Students need to be prepared for a fast-moving digital world where they have the confidence and skills to not only use digital technologies, but to **design and create** digital systems.

There are two trends from Bocconi et al. (2016) research as to why educators should include computational thinking (CT) into their curriculum that support the NZ Ministry of Education's digital technology curriculum ethos:

1. CT skills in young people enable them to think in different ways, express ideas through various mediums, solve real-world problems and look at everyday issues from a wider perspective.
2. CT is needed to fill job vacancies in ICT, boost economies and prepare people for future employment.

Why do we need Computational Thinking?

Computational Thinking:

Decomposition: breaking down, looking at something closer - each part, isolating parts

Abstraction: taking out, open for interpretation, different perspective/view, summary of something written - key points, thinking about other people's ideas. Focused.

Logic/pattern recognition: if I do this, what will happen? Cause/effect. Prior knowledge.

Algorithm: equation, steps, sequences, rules, can trip you up

WHAT is Computational Thinking?

WHAT is Computational Thinking?

Decomposition – breaking down the problem into smaller, more manageable parts

Abstraction – focusing on the important information only, ignoring irrelevant details

Algorithms – developing a step-by-step solution to the problem

Logic/pattern recognition - looking for similarities among and within problems

Computational Thinking Progress Outcome 1

Students break down a simple non-computerised task into a set of precise, unambiguous, step by step instructions (algorithmic thinking).

They are able to give these instructions, and identify if they have gone wrong and correct them (simple debugging).

By doing this they show that they can use their decomposition skills to take a task and break it down into its smallest steps.