Computational thinking for digital technologies

Year	Use of computers (Digital and non-digital contexts)	Algorithmic thinking (Decomposing problems and creating algorithms)	Programming concepts: *Programming* (Developing computer programs)	Programming concepts: *Testing & debugging* (Identifying and correcting bugs in algorithms & programs)	Principles of Computer Science (Binary numbers, data representation, algorithm efficiency, error detection & human-computer interaction)
	In authentic contexts and taking account of end users… (Technology strands underpin the PO's)				
1–3 (PO1)	Non-computerised	Break down simple tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking). Give the instructions.		Identify any errors in algorithms as they are followed, and correct them (simple debugging).	
5–6 (PO2)	Computerised (age-appropriate) & non-computerised	Give and follow simple algorithms	Create simple programs using algorithms they have developed using outputs & sequence	Debug simple algorithms/programs	
7–8 (PO3)	Computerised (age-appropriate) & non-computerised	Decompose problems into step-by-step instructions to create algorithms for computer programs. Use logical thinking to predict the behaviour of the programs. Understand that there can be more than one algorithm for the same problem.	Develop simple programs that use inputs, outputs, sequence and iteration (repeating part of the algorithm with a loop).	Debug their simple programs.	Understand that digital devices store data using just two states represented by binary digits (bits).
9–10 (PO4)	Computerised (age-appropriate) & non-computerised	Decompose problems to create simple algorithms using the three building blocks: sequence, selection, and iteration.	Implement their algorithms by creating programs that use inputs, outputs, sequence, basic selection using comparative operators, and iteration.	Debug simple algorithms and programs by identifying when and explaining why things go wrong with their instructions, and explaining how they fixed them.	Understand: data representation using binary digits & error detection in data storage & transmission. Evaluate efficiency of: algorithms (search/sort data); interfaces (incl usability).
10–11 (PO5)	Computerised (age-appropriate) & non-computerised	Independently decompose problems into algorithms.	Use their algorithms to create programs with inputs, outputs, sequence, selection using comparative and logical operators and variables of different data types, and iteration. Determine when to use different types of control structures.	Document their programs, using an organised approach for testing and debugging.	Understand how computers store more complex types of data using binary digits, and they develop programs considering human-computer interaction (HCI) heuristics.
11 (PO6)					

Kia Takatū ā-Matihiko
Digital Readiness