BIG IDEAS

Decomposition and abstraction improve our ability to understand and solve problems.	Algorithms describe the process of solving computational problems.	Programming is a tool that allows us to implement computational thinking.	Data representation allows us to understand and efficiently solve problems.

Learning Standards

Curricular Competencies	Content
Students are expected to do the following: **Reasoning and analyzing** • Use **reasoning and logic** to analyze and apply mathematical ideas • **Estimate** algorithmic correctness • Demonstrate **fluent and flexible thinking** • Use **tools** or technology to analyze relationships and test conjectures • **Model** mathematics in contextualized experiences **Understanding and solving** • Develop, demonstrate, and apply **conceptual understanding** of mathematical ideas • **Visualize** to explore and illustrate mathematical concepts and relationships • Apply **flexible strategies** to solve problems in both abstract and contextualized situations • Engage in problem-solving experiences that are connected to place, story, cultural **practices**, and perspectives relevant to First Peoples communities, the local community, and other cultures **Communicating and representing** • Communicate mathematical thinking in **many ways** • Use mathematical and computer science vocabulary and language to contribute to **discussions** • **Represent** mathematical ideas in a variety of ways • Explain and justify mathematical and computational ideas	*Students are expected to know the following:* • **ways** to access variables in memory • ways in which **data structures** are organized in memory • **uses** of multidimensional arrays • **algorithms**, including sorting and searching • performance **analysis** by Big-O notation • **recursive problem solving** • **persistent memory** • **encapsulation** of data • ways to model **mathematical problems**

Learning Standards (continued)

Curricular Competencies	Content
Connecting and reflecting • **Reflect** on mathematical and computational thinking • Use mathematics and computer science to support personal choices • Connect mathematical and computer science concepts to each other and to **other areas and personal interests** • **Incorporate** First Peoples worldviews and perspectives to make connections to computer science concepts	

Curricular Competencies – Elaborations	MATHEMATICS – Computer Science Grade 12

- **reasoning and logic:**
  - inductive and deductive reasoning
  - predicting, generalizing, drawing conclusions through experiences and coding
- **Estimate:**
  - avoiding logical errors
  - justifying correctness through test cases
  - estimating run-time complexity
- **fluent and flexible thinking:**
  - understanding the efficiency of different algorithms in solving the same problem
- **Tools**
  - using integrated development environments (IDE)
  - using the IDE debugger to inspect memory at run-time
  - importing third-party libraries
  - using visual diff tools to view code differences
  - using memory analyzers to discover memory leaks
  - using version control systems to share source code among team members (e.g., cvs, svn, git)
- **Model:**
  - using concrete materials, dynamic interactive technology
  - representing a situation graphically and/or symbolically
  - using technology to explore and create patterns, simulations, and relationships and to test conjectures

Curricular Competencies – Elaborations

- **conceptual understanding:**
  - developed through playing with ideas, inquiry, and problem solving
- **Visualize:**
  - generating simulations and models through computing
- **flexible strategies:**
  - designing algorithms that solve a class of problems rather than a single problem
- **practices:**
  - including context, strategies and approaches, language across cultures
  - Learning takes patience and time.
  - Code Talkers (cryptography) (https://en.wikipedia.org/wiki/Code_talker)
- **many ways:**
  - including oral, written, pictures, use of technology
- **discussions:**
  - developing a mathematical community in the classroom through discourse — partner talks, small-group discussions, teacher-student conferences
- **Represent:**
  - concretely (http://csunplugged.org), pictorially, symbolically, including using models, tables, flow charts, words, numbers, symbols
- **Reflect:**
  - sharing the mathematical and computational thinking of self and others, including evaluating strategies and solutions, extending, posing new problems and questions
- **other areas and personal interests:**
  - to develop a sense of how computer science helps us understand the world around us (e.g., daily activities, local and traditional practices, the environment, popular media and news events, social justice, cross-curricular integration)
- **Incorporate:**
  - http://www.fnesc.ca/resources/math-first-peoples/

Content – Elaborations

- **ways:**
  - pass by value versus by reference, or mutable/immutable data types
- **data structures:**
  - vectors, lists, queues, dictionaries, maps, trees, stacks
- **uses:**
  - board games, image manipulation, representing tabular data or matrices
- **algorithms:**
  - sorting: bubble, insertion, selection, quick, merge; searching: binary search, data structure traversal
- **analysis:**
  - analyzing algorithms to predict and compare run-time complexity
- **recursive problem solving:**
  - Fibonacci sequence, exponents, factorials, palindromes, combinations, greatest common factor, fractals
- **persistent memory:**
  - read from/write to a file
- **encapsulation:**
  - creating your own data type, class, or structure as well as public, private, static/class variables
- **mathematical problems**:
  - estimate theoretical probability through simulation, sequences, and series; solve a system of linear equations, exponential growth/decay; solve a polynomial equation; calculate statistical values, such as frequency, central tendencies, standard deviation, of large data set, or greatest common factor/least common multiples