BRITISH
COLUMBIA
Ministry of Education

Area of Learning: MATHEMATICS — Computer Science Grade 11

BIG IDEAS

Decomposition and abstraction improve our ability to understand, reduce complexity, 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

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

Content

Students are expected to know the following:

- **ways** to represent **basic data types**
- **basic programming concepts**
- variable **scope**
- ways to construct and evaluate **logical statements**
- use of **control flow** to manipulate program execution
- **development of algorithms** to solve problems in multiple ways
- techniques for **operations** on and **searching** of arrays and lists
- problem decomposition through **modularity**
- **uses** of computing for **financial analysis**
- 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 11

- **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)
  - importing third-party libraries
  - using visual diff tools to view code differences
  - using memory analyzers to discover memory leaks
- **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:**
  - using different algorithms to solve the same problem
  - designing algorithms that solve a class of problems rather than a single problem
- **practices:**
  - including context, strategies and approaches, language across cultures
  - http://www.behavioradvisor.com/CircleOfCourage.html
  - 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/
  - http://www.behavioradvisor.com/CircleOfCourage.html

Content – Elaborations

- **ways:**
  - number systems (e.g., binary, hexadecimal)
- **basic data types:**
  - strings, integers, characters, floating point
- **basic programming concepts:**
  - variables, constants, mathematical operations, input/output
- **scope:**
  - local versus global
- **logical statements:**
  - logical operators (AND, OR, NOT), relational operators (<, >, <=, >=, ==, != or <>), and logical equivalences (e.g., De Morgan's Law), simplification of logical statements, truth tables
- **control flow:**
  - loops (for, while, nested loop) and decision structures (if-then-else)
- **development of algorithms:**
  - step-wise refinement, pseudocode or flowcharts, translating between pseudo-code and code (and vice versa)
- **operations:**
  - append, remove, insert, delete, indices
- **searching:**
  - searching algorithms such as linear and binary searches
- **modularity:**
  - use of methods/functions to reduce complexity, reuse code and use function parameters, return values
- **uses:**
  - for example, time value of money, appreciation/depreciation, mortgage amortization
- **financial analysis:**
  - modify the variables of a financial scenario to run "what-if" analysis on them (e.g., compare different monthly payments, term lengths, interest rates)
- **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