BIG IDEAS

Decomposition and **abstraction** help us to solve difficult problems by managing complexity.	**Algorithms** are essential in solving problems computationally.	Programming is a tool that allows us to implement **computational thinking**.	**Solving problems** is a creative process.	**Data representation** allows us to understand and solve problems efficiently.

Learning Standards

Curricular Competencies	Content
Students are expected to do the following:	*Students are expected to know the following:*
Reasoning and modelling • Develop **fluent, flexible, and strategic thinking** to analyze and create algorithms • Explore, **analyze**, and apply mathematical ideas and computer science concepts using **reason**, **technology,** and **other tools** • **Model** with mathematics in **situational contexts** • **Think creatively** and with **curiosity and wonder** when exploring problems **Understanding and solving** • Develop, demonstrate, and apply conceptual understanding through experimentation, **inquiry**, and problem solving • **Visualize** to explore and illustrate computer science concepts and relationships • Apply **flexible and strategic approaches** to **solve problems** • Solve problems with **persistence and a positive disposition** • Engage in problem-solving experiences **connected** with place, story, cultural practices, and perspectives relevant to local First Peoples communities, the local community, and other cultures	• **access variables** in memory • ways in which **data structures** are organized in memory • **uses** of multidimensional arrays • classical algorithms, including **sorting and searching** • use of Big-O notation to help predict run-time **performance** • **recursive problem solving** • **persistent memory** • **encapsulation** of data • ways to **model mathematical problems**

Learning Standards (continued)

Curricular Competencies	Content
Communicating and representing • **Explain and justify** computer science ideas and **decisions** in **many ways** • **Represent** computer science ideas in concrete, pictorial, and symbolic forms • Use computer science and mathematical vocabulary and language to contribute to **discussions** in the classroom • Take risks when offering ideas in classroom **discourse** **Connecting and reflecting** • **Reflect** on mathematical and computational thinking • **Connect mathematical and computer science concepts** with each other, other areas, and personal interests • Use **mistakes** as **opportunities to advance learning** • **Incorporate** First Peoples worldviews, perspectives, **knowledge**, and **practices** to make connections with computer science concepts	

Big Ideas – Elaborations

- **abstraction:**
  - reducing complexity by representing essential features without including the background details or explanations

  Sample questions to support inquiry with students:
  - How do we decide when an object should be abstracted?
  - How do we choose public features?
  - How do we choose which features are advertised?
  - How does hiding background detail simplify the problem-solving process?
- **Algorithms:**

  Sample questions to support inquiry with students:
  - When comparing algorithms, how do we determine which one is most efficient?
  - Can an elegant algorithm be efficient?
  - How is an algorithm formulated?
  - What makes one algorithm better than another algorithm?
  - What is the relationship between elegant algorithms and efficient algorithms?
  - Can all problems be solved through a series of predefined steps?
- **computational thinking:**
  - a thought process that uses pattern recognition and decomposition to describe an algorithm in a way that a computer can execute

  Sample questions to support inquiry with students:
  - How do we decide which programming language to use in solving a specific problem?
  - Why is code readability important?
  - What factors affect code readability?
  - How much source code documentation is enough?
  - Are there patterns in the solution that can be generalized?
  - How do we recognize patterns?
- **Solving problems:**

  Sample questions to support inquiry with students:
  - How many different ways can this problem be solved?
  - How do we determine which solution is better?
  - How do we approach solving a problem in different ways?
  - Without knowing a solution, how do we start to solve a problem?

Big Ideas – Elaborations

- **Data representation:**
  - a method of storing and organizing information in a container

  Sample questions to support inquiry with students:
  - When should we create our own data type?
  - How do computers use electricity to represent data?
  - How can we organize our data types more efficiently?
  - How do we decide which data types to use?

MATHEMATICS – Computer Science
Grade 12

Curricular Competencies – Elaborations

- **fluent, flexible, and strategic thinking:**
  - understanding the efficiency of different algorithms in solving the same problem, balancing performance and elegance
- **analyze:**
  - examine the structure of and connections between mathematical ideas (e.g., big-O analysis)
- **reason:**
  - inductive and deductive reasoning
  - predictions, generalizations, conclusions drawn from experiences (e.g., with coding)
- **technology:**
  - graphing technology, dynamic geometry, calculators, virtual manipulatives, concept-based apps
  - can be used for a wide variety of purposes, including:
    - exploring and demonstrating mathematical relationships
    - organizing and displaying data
    - generating and testing inductive conjectures
    - mathematical modelling

- **other tools:**
  - integrated development environments (IDE)
  - IDE debugger to inspect memory at run-time
  - third-party libraries
  - visual code comparison tools to view code differences (e.g., Meld)
  - memory analyzers to discover memory leaks
  - version control systems to share source code among team members (e.g., git)
- **Model:**
  - use mathematical concepts and tools to solve problems and make decisions (e.g., in real-life and/or abstract scenarios)
  - take a complex, essentially non-mathematical scenario and figure out what mathematical concepts and tools are needed to make sense of it
- **situational contexts:**
  - including real-life scenarios and open-ended challenges that connect mathematics with everyday life
- **Think creatively:**
  - by being open to trying different strategies
  - refers to creative and innovative mathematical thinking rather than to representing math in a creative way, such as through art or music
- **curiosity and wonder:**
  - asking questions to further understanding or to open other avenues of investigation
- **inquiry:**
  - includes structured, guided, and open inquiry
  - noticing and wondering
  - determining what is needed to make sense of and solve problems
- **Visualize:**
  - visualize data structures pictorially
  - use flow charts
  - use code visualization tools or websites (e.g., http://pythontutor.com/)
- **flexible and strategic approaches:**
  - using different algorithms to solve the same problem
  - designing algorithms that solve a class of problems rather than a single problem
  - deciding which programming patterns and well-known algorithms to use to solve a problem
  - choosing an effective strategy to solve a problem (e.g., guess and check, model, solve a simpler problem, use a chart, use diagrams, role-play)

- **solve problems:**
  - interpret a situation to identify a problem
  - apply mathematics to solve the problem
  - analyze and evaluate the solution in terms of the initial context
  - repeat this cycle until a solution makes sense
- **persistence and a positive disposition:**
  - not giving up when facing a challenge
  - problem solving with vigour and determination
- **connected:**
  - through daily activities, local and traditional practices, popular media and news events, cross-curricular integration
  - by posing and solving problems or asking questions about place, stories, and cultural practices
- **Explain and justify:**
  - use mathematical arguments to convince
  - includes anticipating consequences
- **decisions:**
  - Have students explore which of two scenarios they would choose and then defend their choice.
- **many ways:**
  - including oral, written, pseudocode, pictures, use of technology
  - communicating effectively according to what is being communicated and to whom
- **Represent:**
  - using pseudocode (e.g., with models, tables, flow charts, words, numbers, symbols)
  - connecting meanings among various representations
  - using concrete materials and dynamic interactive technology
- **discussions:**
  - partner talks, small-group discussions, teacher-student conferences
- **discourse:**
  - is valuable for deepening understanding of concepts
  - can help clarify students' thinking, even if they are not sure about an idea or have misconceptions
- **Reflect:**
  - share the mathematical and computational thinking of self and others, including evaluating strategies and solutions, extending, posing new problems and questions

Curricular Competencies – Elaborations

- **Connect mathematical and computer science concepts:**
  - to develop a sense of how computer science helps us understand the world around us (e.g., daily activities, local and traditional practices, popular media and news events, social justice, cross-curricular integration)
- **mistakes:**
  - include syntax, semantic, run-time, and logic errors
- **opportunities to advance learning:**
  - by:
    - analyzing errors to discover misunderstandings
    - making adjustments in further attempts (e.g., debugging)
    - identifying not only mistakes but also parts of a solution that are correct
- **Incorporate:**
  - by:
    - collaborating with Elders and knowledge keepers among local First Peoples
    - exploring the First Peoples Principles of Learning (e.g., Learning is holistic, reflexive, reflective, experiential, and relational [focused on connectedness, on reciprocal relationships, and a sense of place]; Learning involves patience and time)
    - making explicit connections with learning mathematics
    - exploring cultural practices and knowledge of local First Peoples and identifying mathematical connections
- **knowledge:**
  - local knowledge and cultural practices that are appropriate to share and that are non-appropriated
- **practices:**
  - Bishop's cultural practices: counting, measuring, locating, designing, playing, explaining
  - Aboriginal Education Resources
  - *Teaching Mathematics in a First Nations Context*, FNESC

Content – Elaborations

- **access variables:**
  - 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
- **sorting and searching:**
  - sorting (e.g., bubble, insertion, selection, quick merge)
  - searching (e.g., binary search, data structure traversal)
- **performance:**
  - analyzing algorithms to predict and compare run-time complexity
  - working with large data sets
- **recursive problem solving:**
  - recognizing recursive problems or patterns
  - 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
- **model mathematical problems:**
  - estimate theoretical probability through simulation
  - represent finite sequences and series
  - solve a system of linear equations, exponential growth/decay
  - solve a polynomial equation
  - calculate statistical values (e.g., frequency, central tendencies, standard deviation) of a large data set

