Computer Science 1204

Curriculum Guide 2019

Newfoundland Labrador

Education and Early Childhood Development

Department of Education and Early Childhood Development Mission Statement

The Department of Education and Early Childhood Development will improve provincial early childhood learning and the K-12 education system to further opportunities for the people of Newfoundland and Labrador.

Table of Contents

Acknowledgements v

Section One: Newfoundland and Labrador Curriculum
- Outcomes Based Education 1
- Context for Teaching and Learning 4
  - Inclusive Education 4
  - Literacy 10
  - Learning Skills for Generation Next 12
- Assessment and Evaluation 15

Section Two: Curriculum Design
- Rationale 19
- Curriculum Outcomes Framework 19
- Course Overview 22
- Suggested Yearly Plan 23
- How to use a Four Column Layout 24
- How to use a Strand Overview 26

Section Three: Specific Curriculum Outcomes
- Unit 1: Introduction to Computer Science 27
- Unit 2: Introduction to Programming Concepts (Block-based Techniques) 50
- Unit 3: Manipulate Text Based Code 70
- Unit 4: Coding Interfaces 96
- Unit 5: Innovation Challenge 116

References 137

Acknowledgements

The Department of Education and Early Childhood Development for Newfoundland and Labrador gratefully acknowledges the contribution of the following members of the Computer Science 1204 Curriculum Committee in the completion of this work:

Corey Downey

Newfoundland and Labrador English School District

Garry Walsh

Clarenville High

John Goldsworthy

Newfoundland and Labrador English School District

Matthew Grant

Holy Trinity High

Mike Broadbent

Elwood High

Tim Goodyear

Centre for Distance Learning and Innovation

Tony Hillier

Department of Education and Early Childhood Development

Victoria Byrne

Corner Brook High

Section One: Newfoundland and Labrador Curriculum

Introduction

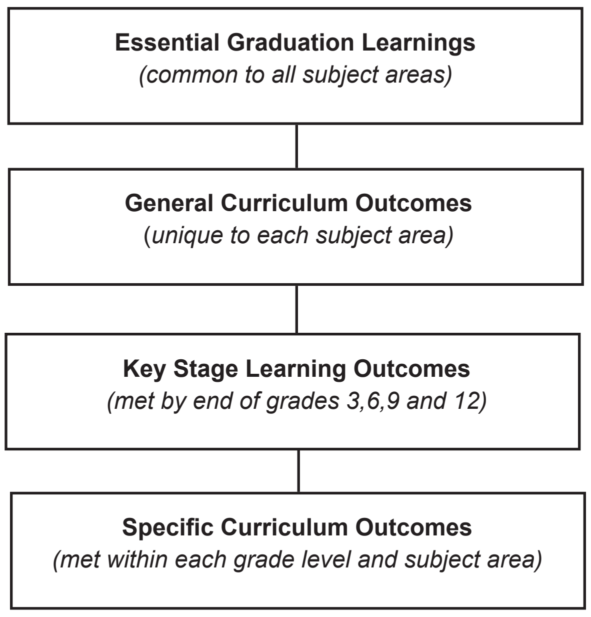
There are multiple factors that impact education: technological developments, increased emphasis on accountability, and globalization. These factors point to the need to consider carefully the education students receive.

The Newfoundland and Labrador Department of Education and Early Childhood Development believes that curriculum design with the following characteristics will help teachers address the needs of students served by the provincially prescribed curriculum:

- Curriculum guides must clearly articulate what students are expected to know and be able to do by the time they graduate from high school.
- There must be purposeful assessment of students' performance in relation to the curriculum outcomes.

Outcomes Based Education

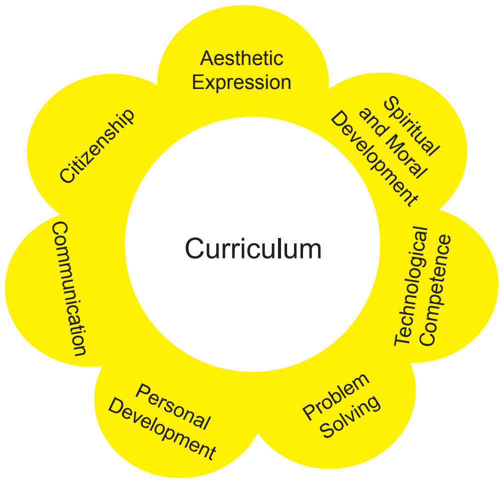
The K-12 curriculum in Newfoundland and Labrador is organized by outcomes and is based on *The Atlantic Canada Framework for Essential Graduation Learning in Schools* (1997). This framework consists of Essential Graduation Learnings (EGLs), General Curriculum Outcomes (GCOs), Key Stage Curriculum Outcomes (KSCOs) and Specific Curriculum Outcomes (SCOs).


Essential Graduation Learnings
(common to all subject areas)
General Curriculum Outcomes
(unique to each subject area)
Key Stage Learning Outcomes
(met by end of grades 3,6,9 and 12)
Specific Curriculum Outcomes
(met within each grade level and subject area)

Essential Graduation Learnings

EGLs provide vision for the development of a coherent and relevant curriculum. They are statements that offer students clear goals and a powerful rationale for education. The EGLs are delineated by general, key stage, and specific curriculum outcomes.

EGLs describe the knowledge, skills, and attitudes expected of all students who graduate from high school. Achievement of the EGLs will prepare students to continue to learn throughout their lives. EGLs describe expectations, not in terms of individual subject areas, but in terms of knowledge, skills, and attitudes developed throughout the K-12 curriculum. They confirm that students need to make connections and develop abilities across subject areas if they are to be ready to meet the shifting and ongoing demands of life, work, and study.



Aesthetic Expression – Graduates will be able to respond with critical awareness to various forms of the arts and be able to express themselves through the arts.

Citizenship – Graduates will be able to assess social, cultural, economic, and environmental interdependence in a local and global context.

Communication – Graduates will be able to use the listening, viewing, speaking, reading and writing modes of language(s), and mathematical and scientific concepts and symbols, to think, learn and communicate effectively.

Problem Solving – Graduates will be able to use the strategies and processes needed to solve a wide variety of problems, including those requiring language, and mathematical and scientific concepts.

Personal Development – Graduates will be able to continue to learn and to pursue an active, healthy lifestyle.

Spiritual and Moral Development – Graduates will demonstrate understanding and appreciation for the place of belief systems in shaping the development of moral values and ethical conduct.

Technological Competence – Graduates will be able to use a variety of technologies, demonstrate an understanding of technological applications, and apply appropriate technologies for solving problems.

Curriculum Outcomes

Curriculum outcomes are statements that articulate what students are expected to know and be able to do in each program area in terms of knowledge, skills, and attitudes.

Curriculum outcomes may be subdivided into General Curriculum Outcomes, Key Stage Curriculum Outcomes, and Specific Curriculum Outcomes.

General Curriculum Outcomes (GCOs)

Each program has a set of GCOs which describe what knowledge, skills, and attitudes students are expected to demonstrate as a result of their cumulative learning experiences within a subject area. GCOs serve as conceptual organizers or frameworks which guide study within a program area. Often, GCOs are further delineated into KSCOs.

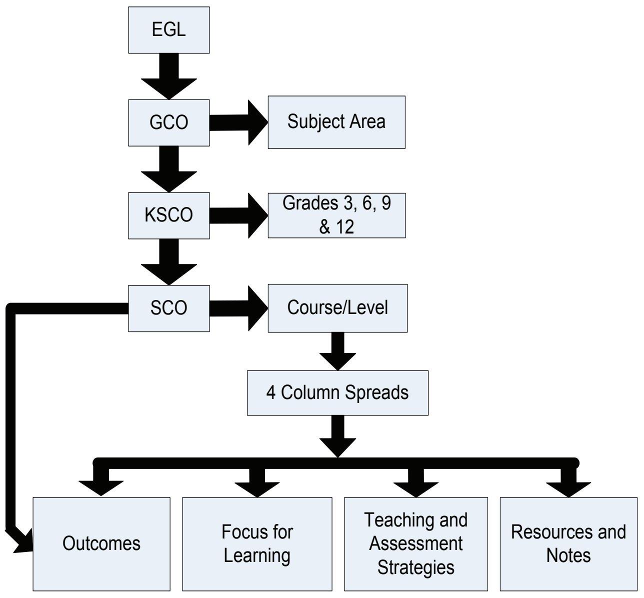
Key Stage Curriculum Outcomes (KSCOs)

Key Stage Curriculum Outcomes (KSCOs) summarize what is expected of students at each of the four key stages of grades three, six, nine, and twelve.

Specific Curriculum Outcomes (SCOs)

SCOs set out what students are expected to know and be able to do as a result of their learning experiences in a course, at a specific grade level. In some program areas, SCOs are further articulated into delineations. *It is expected that all SCOs will be addressed during the course of study covered by the curriculum guide.*

EGLs to Curriculum Guides



Context for Teaching and Learning

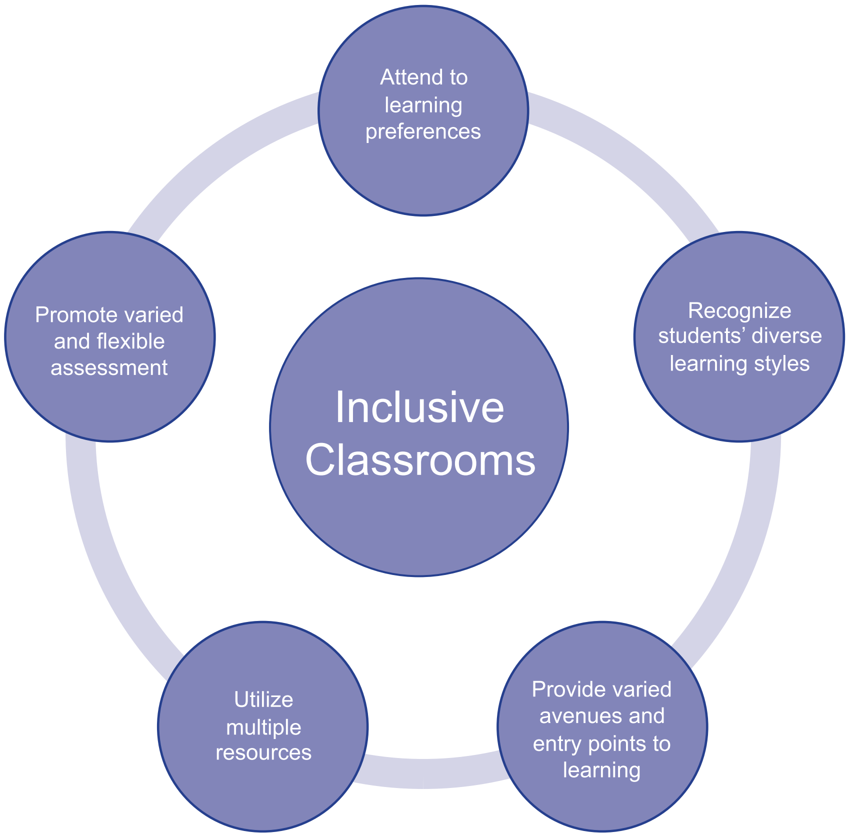
Teachers are responsible to help students achieve outcomes. This responsibility is a constant in a changing world. As programs change over time so does educational context. Several factors make up the educational context in Newfoundland and Labrador today: inclusive education, support for gradual release of responsibility teaching model, focus on literacy and learning skills in all programs, and support for education for sustainable development.

Inclusive Education

Valuing Equity and Diversity

Effective inclusive schools have the following characteristics: supportive environment, positive relationships, feelings of competence, and opportunities to participate. (The Centre for Inclusive Education, 2009)

All students need to see their lives and experiences reflected in their school community. It is important that the curriculum reflect the experiences and values of all genders and that learning resources include and reflect the interests, achievements, and perspectives of all students. An inclusive classroom values the varied experiences and abilities as well as social and ethno-cultural backgrounds of all students while creating opportunities for community building. Inclusive policies and practices promote mutual respect, positive interdependencies, and diverse perspectives. Learning resources should include a range of materials that allow students to consider many viewpoints and to celebrate the diverse aspects of the school community.


Inclusive Classrooms
Attend to learning preferences
Recognize students' diverse learning styles
Provide varied avenues and entry points to learning
Utilize multiple resources
Promote varied and flexible assessment

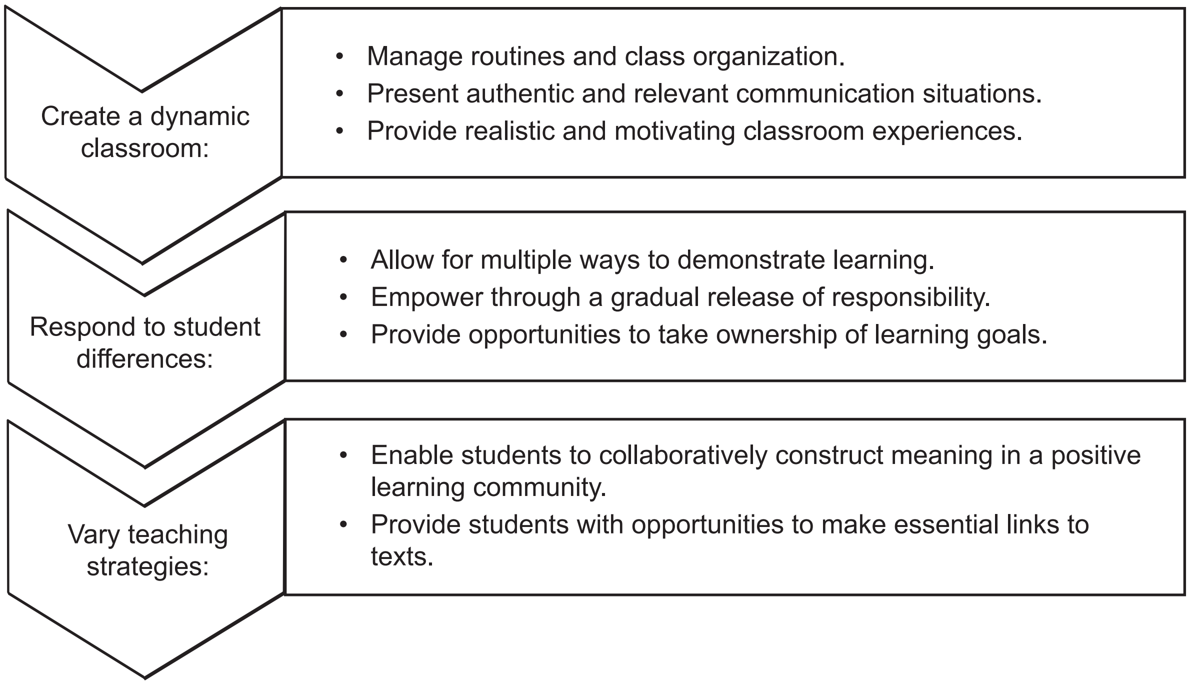
Differentiated Instruction

Differentiated instruction is a teaching philosophy based on the premise that teachers should adapt instruction to student differences. Rather than marching students through the curriculum lockstep, teachers should modify their instruction to meet students' varying readiness levels, learning preferences, and interests. Therefore, the teacher proactively plans a variety of ways to 'get it' and express learning. (Carol Ann Tomlinson, 2008)

Curriculum is designed and implemented to provide learning opportunities for all students according to abilities, needs, and interests. Teachers must be aware of and responsive to the diverse range of learners in their classes. Differentiated instruction is a useful tool in addressing this diversity.

Differentiated instruction responds to different readiness levels, abilities, and learning profiles of students. It involves actively planning so that the process by which content is delivered, the way the resource is used, and the products students create are in response to the teacher's knowledge of whom he or she is interacting with. Learning environments should be flexible to accommodate various learning preferences of the students. Teachers continually make decisions about selecting teaching strategies and structuring learning activities that provide all students with a safe and supportive place to learn and succeed.

Planning for Differentiation



Create a dynamic classroom:

- Manage routines and class organization.
- Present authentic and relevant communication situations.
- Provide realistic and motivating classroom experiences.

Respond to student differences:

- Allow for multiple ways to demonstrate learning.
- Empower through a gradual release of responsibility.
- Provide opportunities to take ownership of learning goals.

Vary teaching strategies:

- Enable students to collaboratively construct meaning in a positive learning community.
- Provide students with opportunities to make essential links to texts.

Differentiating the Content

Differentiating content requires teachers to pre-assess students to identify those who require prerequisite instruction, as well as those who have already mastered the concept and may therefore apply strategies learned to new situations. Another way to differentiate content is to permit students to adjust the pace at which they progress through the material. Some students may require additional time while others will move through at an increased pace and thus create opportunities for enrichment or more indepth consideration of a topic of particular interest.

Teachers should consider the following examples of differentiating content:

- Meet with small groups to reteach an idea or skill or to extend the thinking or skills.
- Present ideas through auditory, visual, and tactile means.
- Use reading materials such as novels, websites, and other reference materials at varying reading levels.

Differentiating the Process

Differentiating the process involves varying learning activities or strategies to provide appropriate methods for students to explore and make sense of concepts. A teacher might assign all students the same product (e.g., presenting to peers) but the process students use to create the presentation may differ. Some students could work in groups while others meet with the teacher individually. The same assessment criteria can be used for all students.

Teachers should consider flexible grouping of students such as whole class, small group, or individual instruction. Students can be grouped according to their learning styles, readiness levels, interest areas, and/or the requirements of the content or activity presented. Groups should be formed for specific purposes and be flexible in composition and short-term in duration.

Teachers should consider the following examples of differentiating the process:

- Offer hands-on activities for students.
- Provide activities and resources that encourage students to further explore a topic of particular interest.
- Use activities in which all learners work with the same learning outcomes but proceed with different levels of support, challenge, or complexity.

Differentiating the Product

Differentiating the product involves varying the complexity and type of product that students create to demonstrate learning outcomes. Teachers provide a variety of opportunities for students to demonstrate and show evidence of what they have learned.

Teachers should give students options to demonstrate their learning (e.g., create an online presentation, write a letter, or develop a mural). This will lead to an increase in student engagement.

Differentiating the Learning Environment

The learning environment includes the physical and the affective tone or atmosphere in which teaching and learning take place, and can include the noise level in the room, whether student activities are static or mobile, or how the room is furnished and arranged. Classrooms may include tables of different shapes and sizes, space for quiet individual work, and areas for collaboration.

Teachers can divide the classroom into sections, create learning centres, or have students work both independently and in groups. The structure should allow students to move from whole group, to small group, pairs, and individual learning experiences and support a variety of ways to engage in learning. Teachers should be sensitive and alert to ways in which the classroom environment supports their ability to interact with students.

Teachers should consider the following examples of differentiating the learning environment:

- Develop routines that allow students to seek help when teachers are with other students and cannot provide immediate attention.
- Ensure there are places in the room for students to work quietly and without distraction, as well as places that invite student collaboration.
- Establish clear guidelines for independent work that match individual needs.
- Provide materials that reflect diversity of student background, interests, and abilities.

The physical learning environment must be structured in such a way that all students can gain access to information and develop confidence and competence.

Meeting the Needs of Students with Exceptionalities

All students have individual learning needs. Some students, however, have exceptionalities (defined by the Department of Education and Early Childhood Development) which impact their learning. The majority of students with exceptionalities access the prescribed curriculum. For details of these exceptionalities see www.gov.nl.ca/edu/k12/studentsupportservices/exceptionalities.html

Supports for these students may include

1. Accommodations
2. Modified Prescribed Courses
3. Alternate Courses
4. Alternate Programs
5. Alternate Curriculum

For further information, see Service Delivery Model for Students with Exceptionalities at www.cdli.ca/sdm/

Classroom teachers should collaborate with instructional resource teachers to select and develop strategies which target specific learning needs.

Meeting the Needs of Students who are Highly Able
(includes gifted and talented)

Some students begin a course or topic with a vast amount of prior experience and knowledge. They may know a large portion of the material before it is presented to the class or be capable of processing it at a rate much faster than their classmates. All students are expected to move forward from their starting point. Many elements of differentiated instruction are useful in addressing the needs of students who are highly able.

Teachers may

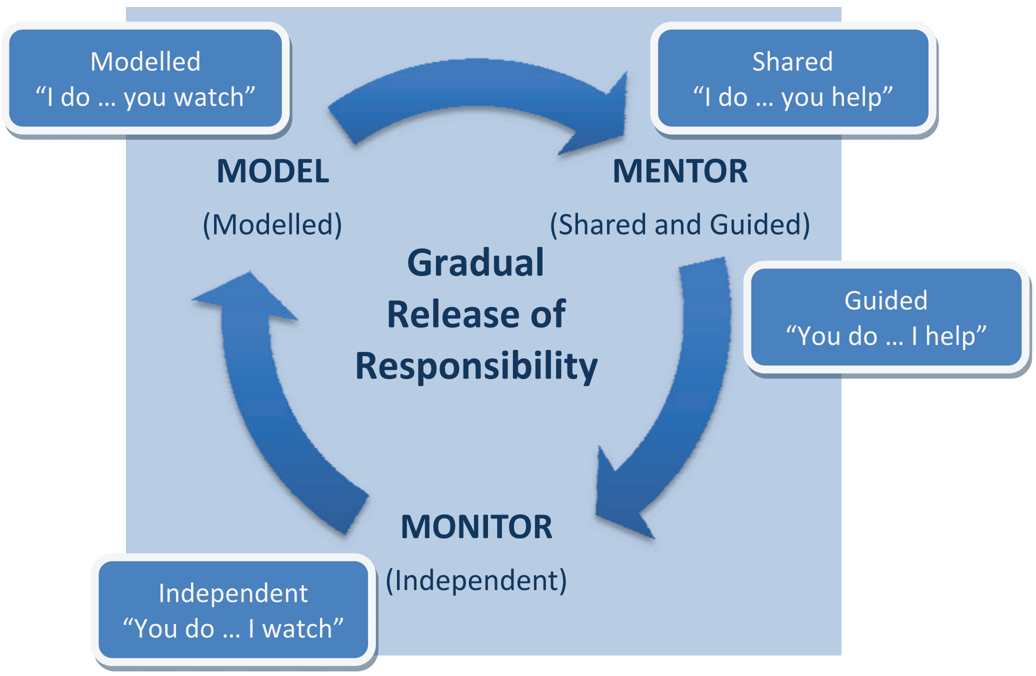
- assign independent study to increase depth of exploration in an area of particular interest;
- compact curriculum to allow for an increased rate of content coverage commensurate with a student's ability or degree of prior knowledge;
- group students with similar abilities to provide the opportunity for students to work with their intellectual peers and elevate discussion and thinking, or delve deeper into a particular topic; and
- tier instruction to pursue a topic to a greater depth or to make connections between various spheres of knowledge.

Highly able students require the opportunity for authentic investigation to become familiar with the tools and practices of the field of study. Authentic audiences and tasks are vital for these learners. Some highly able learners may be identified as gifted and talented in a particular domain. These students may also require supports through the Service Delivery Model for Students with Exceptionalities.

Gradual Release of Responsibility

Teachers must determine when students can work independently and when they require assistance. In an effective learning environment, teachers choose their instructional activities to model and scaffold composition, comprehension, and metacognition that is just beyond the students' independence level. In the gradual release of responsibility approach, students move from a high level of teacher support to independent work. If necessary, the teacher increases the level of support when students need assistance. The goal is to empower students with their own learning strategies, and to know how, when, and why to apply them to support their individual growth. Guided practice supports student independence. As a student demonstrates success, the teacher should gradually decrease his or her support.

Gradual Release of Responsibility Model



Literacy

"Literacy is the ability to identify, understand, interpret, create, communicate and compute, using printed and written materials associated with varying contexts. Literacy involves a continuum of learning in enabling individuals to achieve their goals, to develop their knowledge and potential, and to participate fully in their community and wider society". To be successful, students require a set of interrelated skills, strategies and knowledge in multiple literacies that facilitate their ability to participate fully in a variety of roles and contexts in their lives, in order to explore and interpret the world and communicate meaning. (The Plurality of Literacy and its Implications for Policies and Programmes, 2004, p.13)

Literacy is

- a process of receiving information and making meaning from it; and
- the ability to identify, understand, interpret, communicate, compute, and create text, images, and sounds.

Literacy development is a lifelong learning enterprise beginning at birth that involves many complex concepts and understandings. It is not limited to the ability to read and write; no longer are we exposed only to printed text. It includes the capacity to learn to communicate, read, write, think, explore, and solve problems. Individuals use literacy skills in paper, digital, and live interactions to engage in a variety of activities:

- Analyze critically and solve problems.
- Comprehend and communicate meaning.
- Create a variety of texts.
- Make connections both personally and inter-textually.
- Participate in the socio-cultural world of the community.
- Read and view for enjoyment.
- Respond personally.

These expectations are identified in curriculum documents for specific subject areas as well as in supporting documents, such as *Cross-Curricular Reading Tools* (CAMET).

With modelling, support, and practice, students' thinking and understandings are deepened as they work with engaging content and participate in focused conversations.

Reading in the Content Areas

The focus for reading in the content areas is on teaching strategies for understanding content. Teaching strategies for reading comprehension benefits all students as they develop transferable skills that apply across curriculum areas.

When interacting with different texts, students must read words, view and interpret text features, and navigate through information presented in a variety of ways including, but not limited to

Advertisements	Movies	Poems
Blogs	Music videos	Songs
Books	Online databases	Speeches
Documentaries	Plays	Video games
Magazine articles	Podcasts	Websites

Students should be able to interact with and comprehend different texts at different levels.

There are three levels of text comprehension:

- Independent level – Students are able to read, view, and understand texts without assistance.
- Instructional level – Students are able to read, view, and understand most texts but need assistance to fully comprehend some texts.
- Frustration level – Students are not able to read or view with understanding (i.e., texts may be beyond their current reading level).

Teachers will encounter students working at all reading levels in their classrooms and will need to differentiate instruction to meet their needs. For example, print texts may be presented in audio form, physical movement may be associated with synthesizing new information with prior knowledge, or graphic organizers may be created to present large amounts of print text in a visual manner.

When interacting with information that is unfamiliar to students, it is important for teachers to monitor how effectively students are using strategies to read and view texts:

- Analyze and think critically about information.
- Determine importance to prioritize information.
- Engage in questioning before, during, and after an activity related to a task, text, or problem.
- Make inferences about what is meant but not said.
- Make predictions.
- Synthesize information to create new meaning.
- Visualize ideas and concepts.

Learning Skills for Generation Next

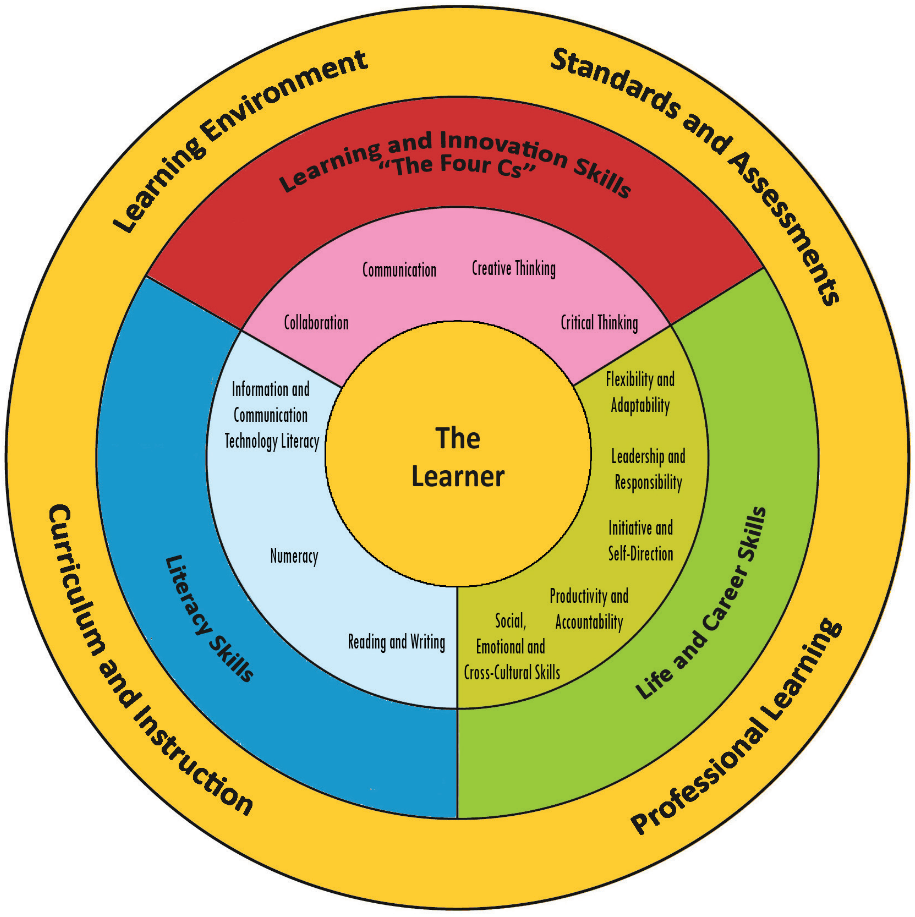
Generation Next is the group of students who have not known a world without personal computers, cell phones, and the Internet. They were born into this technology. They are digital natives.

Students need content and skills to be successful. Education helps students learn content and develop skills needed to be successful in school and in all learning contexts and situations. Effective learning environments and curricula challenge learners to develop and apply key skills within the content areas and across interdisciplinary themes.

Learning Skills for Generation Next encompasses three broad areas:

- Learning and Innovation Skills enhance a person's ability to learn, create new ideas, problem solve, and collaborate.
- Life and Career Skills address leadership, and interpersonal and affective domains.
- Literacy Skills develop reading, writing, and numeracy, and enhance the use of information and communication technology.

The diagram below illustrates the relationship between these areas. A 21st century curriculum employs methods that integrate innovative and research-driven teaching strategies, modern learning technologies, and relevant resources and contexts.


Learning Environment
Standards and Assessments
Curriculum and Instruction
Professional Learning
Learning and Innovation Skills
"The Four Cs"
Communication
Creative Thinking
Collaboration
Critical Thinking
Information and Communication Technology Literacy
Numeracy
Reading and Writing
Literacy Skills
Flexibility and Adaptability
Leadership and Responsibility
Initiative and Self-Direction
Productivity and Accountability
Social, Emotional and Cross-Cultural Skills
Life and Career Skills
The Learner

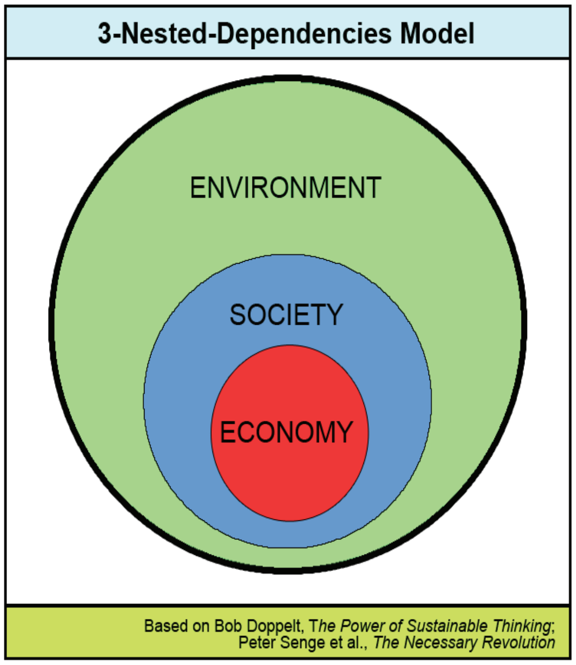
Support for students to develop these abilities and skills is important across curriculum areas and should be integrated into teaching, learning, and assessment strategies. Opportunities for integration of these skills and abilities should be planned with engaging and experiential activities that support the gradual release of responsibility model. For example, lessons in a variety of content areas can be infused with learning skills for Generation Next by using open-ended questioning, role plays, inquiry approaches, self-directed learning, student role rotation, and Internet-based technologies.

All programs have a shared responsibility in developing students' capabilities within all three skill areas.

Education for Sustainable Development

Sustainable development is defined as "development that meets the needs of the present without compromising the ability of future generations to meet their own needs". (Our Common Future, 43)

Sustainable development is comprised of three integrally connected areas: economy, society, and environment.


3-Nested-Dependencies Model
ENVIRONMENT
SOCIETY
ECONOMY
Based on Bob Doppelt, *The Power of Sustainable Thinking*; Peter Senge et al., *The Necessary Revolution*

As conceived by the United Nations Educational, Scientific, and Cultural Organization (UNESCO) the overall goal of Education for Sustainable Development (ESD) is to integrate the knowledge, skills, values, and perspectives of sustainable development into all aspects of education and learning. Changes in human behaviour should create a more sustainable future that supports environmental integrity and economic viability, resulting in a just society for all generations.

ESD involves teaching *for* rather than teaching *about* sustainable development. In this way students develop the skills, attitudes, and perspectives to meet their present needs without compromising the ability of future generations to meet their needs.

Within ESD, the knowledge component spans an understanding of the interconnectedness of our political, economic, environmental, and social worlds, to the role of science and technology in the development of societies and their impact on the environment. The skills necessary include being able to assess bias, analyze consequences of choices, ask questions, and solve problems. ESD values and perspectives include an appreciation for the interdependence of all life forms, the importance of individual responsibility and action, an understanding of global issues as well as local issues in a global context. Students need to be aware that every issue has a history, and that many global issues are linked.

Assessment and Evaluation

Assessment

Assessment is the process of gathering information on student learning.

How learning is assessed and evaluated and how results are communicated send clear messages to students and others about what is valued.

Assessment instruments are used to gather information for evaluation. Information gathered through assessment helps teachers determine students' strengths and needs, and guides future instruction.

Teachers are encouraged to be flexible in assessing student learning and to seek diverse ways students might demonstrate what they know and are able to do.

Evaluation involves the weighing of the assessment information against a standard in order to make a judgement about student achievement.

Assessment can be used for different purposes:

1. Assessment *for* learning guides and informs instruction.
2. Assessment *as* learning focuses on what students are doing well, what they are struggling with, where the areas of challenge are, and what to do next.
3. Assessment *of* learning makes judgements about student performance in relation to curriculum outcomes.

1. Assessment for Learning

Assessment *for* learning involves frequent, interactive assessments designed to make student learning visible. This enables teachers to identify learning needs and adjust teaching accordingly. Assessment *for* learning is not about a score or mark; it is an ongoing process of teaching and learning:

- Pre-assessments provide teachers with information about what students already know and can do.
- Self-assessments allow students to set goals for their own learning.
- Assessment *for* learning provides descriptive and specific feedback to students and parents regarding the next stage of learning.
- Data collected during the learning process from a range of tools enables teachers to learn as much as possible about what a student knows and is able to do.

2. Assessment as Learning

Assessment *as* learning involves students' reflecting on their learning and monitoring their own progress. It focuses on the role of the student in developing metacognition and enhances engagement in their own learning. Students can

- analyze their learning in relation to learning outcomes,
- assess themselves and understand how to improve performance,
- consider how they can continue to improve their learning, and
- use information gathered to make adaptations to their learning processes and to develop new understandings.

3. Assessment of Learning

Assessment *of* learning involves strategies designed to confirm what students know in terms of curriculum outcomes. It also assists teachers in determining student proficiency and future learning needs. Assessment *of* learning occurs at the end of a learning experience and contributes directly to reported results. Traditionally, teachers relied on this type of assessment to make judgements about student performance by measuring learning after the fact and then reporting it to others. Used in conjunction with the other assessment processes previously outlined, assessment *of* learning is strengthened. Teachers can

- confirm what students know and can do;
- report evidence to parents/guardians, and other stakeholders, of student achievement in relation to learning outcomes; and
- report on student learning accurately and fairly using evidence obtained from a variety of contexts and sources.

Involving Students in the Assessment Process

Students should know what they are expected to learn as outlined in the specific curriculum outcomes of a course as well as the criteria that will be used to determine the quality of their achievement. This information allows students to make informed choices about the most effective ways to demonstrate what they know and are able to do.

It is important that students participate actively in assessment by co-creating criteria and standards which can be used to make judgements about their own learning. Students may benefit from examining various scoring criteria, rubrics, and student exemplars.

Students are more likely to perceive learning as its own reward when they have opportunities to assess their own progress. Rather than asking teachers, "What do you want?", students should be asking themselves questions:

- What have I learned?
- What can I do now that I couldn't do before?
- What do I need to learn next?

Assessment must provide opportunities for students to reflect on their own progress, evaluate their learning, and set goals for future learning.

Assessment Tools

In planning assessment, teachers should use a broad range of tools to give students multiple opportunities to demonstrate their knowledge, skills, and attitudes. The different levels of achievement or performance may be expressed as written or oral comments, ratings, categorizations, letters, numbers, or as some combination of these forms.

The grade level and the activity being assessed will inform the types of assessment tools teachers will choose:

Anecdotal Records	Photographic Documentation
Audio/Video Clips	Podcasts
Case Studies	Portfolios
Checklists	Presentations
Conferences	Projects
Debates	Questions
Demonstrations	Quizzes
Exemplars	Role Plays
Graphic Organizers	Rubrics
Journals	Self-assessments
Literacy Profiles	Tests
Observations	Wikis

Assessment Guidelines

Assessments should measure what they intend to measure. It is important that students know the purpose, type, and potential marking scheme of an assessment. The following guidelines should be considered:

- Collect evidence of student learning through a variety of methods; do not rely solely on tests and paper and pencil activities.
- Develop a rationale for using a particular assessment of learning at a specific point in time.
- Provide descriptive and individualized feedback to students.
- Provide students with the opportunity to demonstrate the extent and depth of their learning.
- Set clear targets for student success using learning outcomes and assessment criteria.
- Share assessment criteria with students so that they know the expectations.

Evaluation

Evaluation is the process of analyzing, reflecting upon, and summarizing assessment information, and making judgements or decisions based on the information gathered. Evaluation is conducted within the context of the outcomes, which should be clearly understood by learners before teaching and evaluation take place. Students must understand the basis on which they will be evaluated and what teachers expect of them.

During evaluation, the teacher interprets the assessment information, makes judgements about student progress, and makes decisions about student learning programs.

Section Two: Curriculum Design

Rationale

Technological competence is one of the Essential Graduation Learnings common to all curricular areas in the Newfoundland and Labrador curriculum. The International Society for Technology in Education (ISTE) outlines Empowered Learner, Knowledge Constructor, Innovative Designer and Creative Communicator as four of its seven standards for students. The Conference Board of Canada Employability Skills Profile lists the ability to communicate, manage information, use numbers and think and solve problems as fundamental employability skills. Technology has become increasingly ubiquitous in the day to day lives of students. Learning how to manage information through technology is an essential form of communications.

Curriculum Outcomes Framework

Technology education engages students directly in constructing technological solutions to everyday, real-world problems.Technology Education employs a wide variety of hands-on activities. Students are exposed to a broad range of technological issues, systems, and problem situations in a systemic, systematic fashion. They employ a wide range of technological resources and processes to design, fabricate, and test solutions to familiar and unfamiliar problems. Outcomes, learning experiences, and evaluation of student achievement reflect and are geared towards engagement. Technology Education provides a naturally integrative function that helps students identify contextual relationships between technological activity and principles, and the underlying scientific, mathematical, and other concepts, principles, laws, and theories.

Computer Science 1204

Computer Science is the study of computational systems. While the discipline does deal with some hardware concepts, Computer Science primarily covers programming and software solution development for a vast list of applications including artificial intelligence, networks, and security. Computer Science 1204 introduces students to one aspect of the field of computer science. Its approach is to provide students with the opportunity to build skills in block based and text based programming languages and then use those new skills to program physical interfaces. The culminating activity of the course is an innovation challenge where students use their newly acquired skills to develop a solution to an authentic real-world problem using programming languages and physical interfacing equipment.

Key Stage Curriculum Outcomes

The Key stage curriculum outcomes, based on the general curriculum outcomes, identify what students are expected to know and be able to do at the end of the primary/elementary, intermediate and high school grades in order to meet the essential graduation learnings. Key stage outcomes are identified for each of the dimensions. These key stage curriculum outcomes serve as the basis for the development of specific programs and courses for Technology Education.

Specific Curricular Outcomes

The specific curriculum outcomes are statements that describe what students will know, value, and be able to do as a result of study in a specific course or program at a grade level. These are found in the curriculum guides for each program or course.

General Curriculum Outcomes (GCOs)	Key Stage Curriculum Outcomes (KSCOs) By the end of grade 12, students will be expected to:
GCO 1: Technological Problem Solving Students will be expected to design, develop, evaluate, and articulate technological solutions.	[1.401] articulate problems that may be solved through technological means • assess diverse needs and opportunities • construct detailed design briefs that include design criteria and a work schedule [1.402] conduct design studies to identify a technological solution to a problem • investigate related solutions • document a range of options to solve this problem • determine and justify the best option • determine resource requirements and availability • develop detailed action plans, including technical drawings and sequences of action [1.403] develop (prototype, fabricate, make) technological solutions to problems • match resources and technical processes for specific tasks construct and test models and prototypes as needed • construct the solution with adherence to the design criteria • document activities, decisions, and milestones [1.404] critically evaluate technological solutions and report their findings • develop detailed evaluations of both their own and others' technological solutions, with reference to independently developed criteria • employ a continuous assessment methodology with the purpose of continuous improvement of the design • document and report their changes, the rationale for change,and conclusions

General Curriculum Outcomes (GCOs)	Key Stage Curriculum Outcomes (KSCOs) By the end of grade 12, students will be expected to:
	[1.405] communicate ideas and information about technological solutions through appropriate technical means • accurately present technical information by using a representative sample of analog and digital tools, including, for example, two- and three-dimensional, computer-assisted drafting and modelling tools • create accurately scaled models and prototypes
GCO 2: Technological Systems Students will be expected to operate and manage technological systems.	[2.401] operate, monitor, and adjust technological systems of increasing complexity [2.402] manage technological systems of increasing complexity [2.403] modify programming logic and control systems to optimize the behaviour of systems [2.404] deconstruct complex technological systems into their simpler systems and components [2.405 troubleshoot and maintain systems
GCO 3: History and Evolution of Technology Students will be expected to demonstrate an understanding of the history and evolution of technology, and of its social and cultural implications.	[3.401] evaluate technological systems in the context of convergence where one system has multiple functions, or divergence where multiple systems have the same function [3.402] evaluate the symbiotic roles of technology and science in modern society [3.403] analyse the symbiotic relationship between technology and education, including factors that influence standards for technological literacy and capability, and ways that the community responds [3.404] critically evaluate the effects of accelerating rates of technological change on self and society [3.405] account for effects of cultural diversity on technological solutions • critically examine the effects of cultural diversity onmarket forces and technological products, and viceversa • incorporate knowledge of cultural diversity into development of technological solutions
GCO 4: Technology and Careers Students will be expected to demonstrate an understanding of current and evolving careers and of the influence of technology on the nature of work.	[4.401] assess and evaluate employability profiles for a variety of workplaces and careers and determine the level of technological literacy and capability they would need to achieve for job entry [4.402] employ design and invention as tools to create entrepreneurial activity [4.403] envision their short- and longer-term future and develop a plan for acquiring the technological literacy/capability required to achieve their vision

General Curriculum Outcomes (GCOs)	Key Stage Curriculum Outcomes (KSCOs) By the end of grade 12, students will be expected to:
GCO 5: Technological Responsibility Students will be expected to demonstrate an understanding of the consequences of their technological choices.	[5.401] demonstrate responsible leadership in employing legal and ethical rules and principles. [5.402] demonstrate responsible leadership in employing health and safety rules and standards [5.403] demonstrate responsible leadership in taking proper measures to manage current and future technological risk

Course Overview

Computer Science 1204 deals with software and hardware topics in the area of Computer Science. Students will build skills in both block based and text based programming languages. Students' purpose for programming will be to control physical devices attached to a computer. Students will use their coding and interfacing skills to develop a solution to an authentic problem in an innovation challenge. This course is organized in a linear fashion. The units are intended to be covered sequentially. The skills acquired in each unit will help students attain the outcomes in the next unit. The success of the final culminating activity will depend heavily on students using the newly acquired skills.This course is divided into five units.

Unit 1: Introduction to Computer Science

Unit 2: Programming Concepts

Unit 3: Manipulating Text Based Programming

Unit 4: Interfacing

Unit 5: Innovation Challenge

Suggested Yearly Plan

Computer Science 1204 is a 110 hour two credit course that is intended to be offered over an entire school year from September to June. There are no prerequisites for this course and it is written for students who are beginning to explore the area of Computer Science and programming

Unit one is the introduction to the course. It covers three sub-topics. It is recommended that this unit take up not more than 15 hours of instruction. Suggested time allocations for each subtopic is provided below.

Subtopic	# of classes
Getting Started in Computational Thinking	4
Exploring Computing Devices	8
Careers and Digital Citizenship	3

Suggested Yearly Plan

Unit two is an exploration of the logic and syntax of programming through the use of a block-based programming language. The teacher could alter the amount of time spent on this unit depending on how much experience students have with block based programming from previous grades. This unit has one subtopic and it is recommended that it take up not more than 15 hours of class time.

Subtopic	# of hours
Programming Concepts	15

Unit three is an explortation of the logic and syntax of an industrial standard text based programming language. This unit has three subtopics and it is recommended that this unit take 25 hours of class time.

Subtopic	# of hours
Introduction to Text Based Programming	5
Writing a Text Based Program	10
Developing Solutions	10

Unit four is the point in the course where students use their programming skills to control physical devices This unit is divided into two subtopics. It is recommended that this unit will take 25 hours of class time.

Subtopic	# of hours
Introduction to Interfacing	10
Sensing and Control	15

Unit five represents the culminating activity for the course. Students will use the skills learned in the first four units to work through a major innovation challenged in unit five. It is recommended that this unit take 30 hours of class time.

Subtopic	# of hours
Getting Organized	5
Getting Started	10
Developing the Solution	15

How to Use the Four Column Curriculum Layout

Outcomes

Column one contains specific curriculum outcomes (SCO) and accompanying delineations where appropriate. The delineations provide specificity in relation to key ideas.

Outcomes are numbered in ascending order.

Delineations are indented and numbered as a subset of the originating SCO.

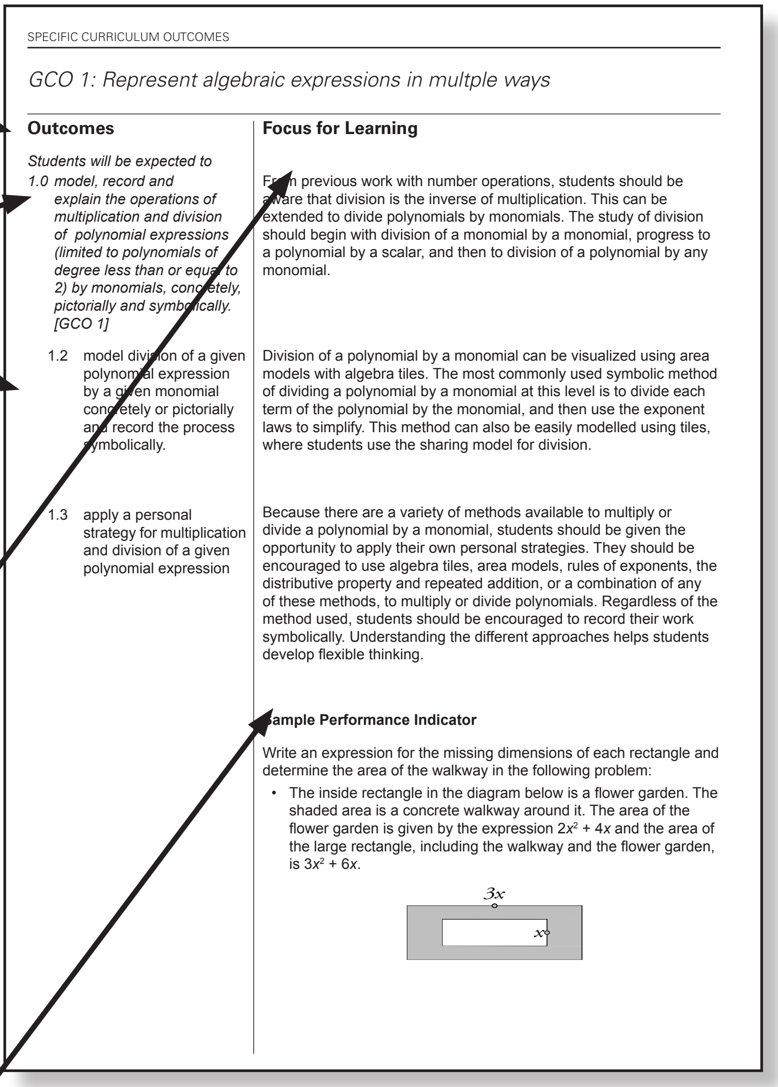
All outcomes are related to general curriculum outcomes.

Focus for Learning

Column two is intended to assist teachers with instructional planning. It also provides context and elaboration of the ideas identified in the first column.

This may include

- cautionary notes
- clarity in terms of scope
- common misconceptions
- depth of treatment
- knowledge required to scaffold and challenge student's learning
- references to prior knowledge



SPECIFIC CURRICULUM OUTCOMES

GCO 1: Represent algebraic expressions in multple ways

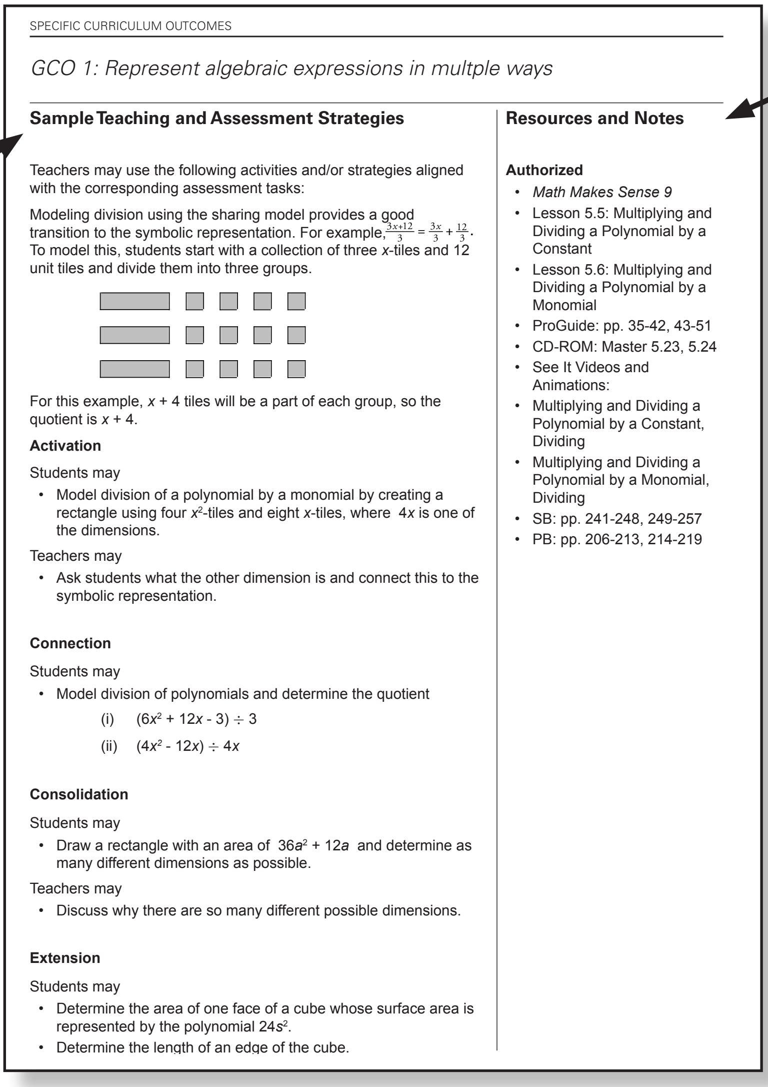
Outcomes	Focus for Learning
Students will be expected to *1.0 model, record and explain the operations of multiplication and division of polynomial expressions (limited to polynomials of degree less than or equal to 2) by monomials, concretely, pictorially and symbolically. [GCO 1]*	From previous work with number operations, students should be aware that division is the inverse of multiplication. This can be extended to divide polynomials by monomials. The study of division should begin with division of a monomial by a monomial, progress to a polynomial by a scalar, and then to division of a polynomial by any monomial.
1.2 model division of a given polynomial expression by a given monomial concretely or pictorially and record the process symbolically.	Division of a polynomial by a monomial can be visualized using area models with algebra tiles. The most commonly used symbolic method of dividing a polynomial by a monomial at this level is to divide each term of the polynomial by the monomial, and then use the exponent laws to simplify. This method can also be easily modelled using tiles, where students use the sharing model for division.
1.3 apply a personal strategy for multiplication and division of a given polynomial expression	Because there are a variety of methods available to multiply or divide a polynomial by a monomial, students should be given the opportunity to apply their own personal strategies. They should be encouraged to use algebra tiles, area models, rules of exponents, the distributive property and repeated addition, or a combination of any of these methods, to multiply or divide polynomials. Regardless of the method used, students should be encouraged to record their work symbolically. Understanding the different approaches helps students develop flexible thinking.
	Sample Performance Indicator Write an expression for the missing dimensions of each rectangle and determine the area of the walkway in the following problem: • The inside rectangle in the diagram below is a flower garden. The shaded area is a concrete walkway around it. The area of the flower garden is given by the expression $2x^2 + 4x$ and the area of the large rectangle, including the walkway and the flower garden, is $3x^2 + 6x$.

Sample Performance Indicator(s)

This provides a summative, higher order activity, where the response would serve as a data source to help teachers assess the degree to which the student has achieved the outcome.

Performance indicators are typically presented as a task, which may include an introduction to establish a context. They would be assigned at the end of the teaching period allocated for the outcome.

Performance indicators would be assigned when students have attained a level of competence, with suggestions for teaching and assessment identified in column three.



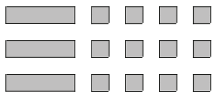
SPECIFIC CURRICULUM OUTCOMES

GCO 1: Represent algebraic expressions in multple ways

Sample Teaching and Assessment Strategies

Teachers may use the following activities and/or strategies aligned with the corresponding assessment tasks:

Modeling division using the sharing model provides a good transition to the symbolic representation. For example, $\frac{3x+12}{3} = \frac{3x}{3} + \frac{12}{3}$. To model this, students start with a collection of three x-tiles and 12 unit tiles and divide them into three groups.



For this example, $x + 4$ tiles will be a part of each group, so the quotient is $x + 4$.

Activation

Students may

- Model division of a polynomial by a monomial by creating a rectangle using four x^2-tiles and eight x-tiles, where $4x$ is one of the dimensions.

Teachers may

- Ask students what the other dimension is and connect this to the symbolic representation.

Connection

Students may

- Model division of polynomials and determine the quotient
  - (i) $(6x^2 + 12x - 3) \div 3$
  - (ii) $(4x^2 - 12x) \div 4x$

Consolidation

Students may

- Draw a rectangle with an area of $36a^2 + 12a$ and determine as many different dimensions as possible.

Teachers may

- Discuss why there are so many different possible dimensions.

Extension

Students may

- Determine the area of one face of a cube whose surface area is represented by the polynomial $24s^2$.
- Determine the length of an edge of the cube.

Resources and Notes

Authorized

- *Math Makes Sense 9*
- Lesson 5.5: Multiplying and Dividing a Polynomial by a Constant
- Lesson 5.6: Multiplying and Dividing a Polynomial by a Monomial
- ProGuide: pp. 35-42, 43-51
- CD-ROM: Master 5.23, 5.24
- See It Videos and Animations:
- Multiplying and Dividing a Polynomial by a Constant, Dividing
- Multiplying and Dividing a Polynomial by a Monomial, Dividing
- SB: pp. 241-248, 249-257
- PB: pp. 206-213, 214-219

Resources and Notes

Column four references supplementary information and possible resources for use by teachers.

These references will provide details of resources suggested in column two and column three.

Suggestions for Teaching and Assessment

This column contains specific sample tasks, activities, and strategies that enable students to meet the goals of the SCOs and be successful with performance indicators. Instructional activities are recognized as possible sources of data for assessment purposes. Frequently, appropriate techniques and instruments for assessment purposes are recommended.

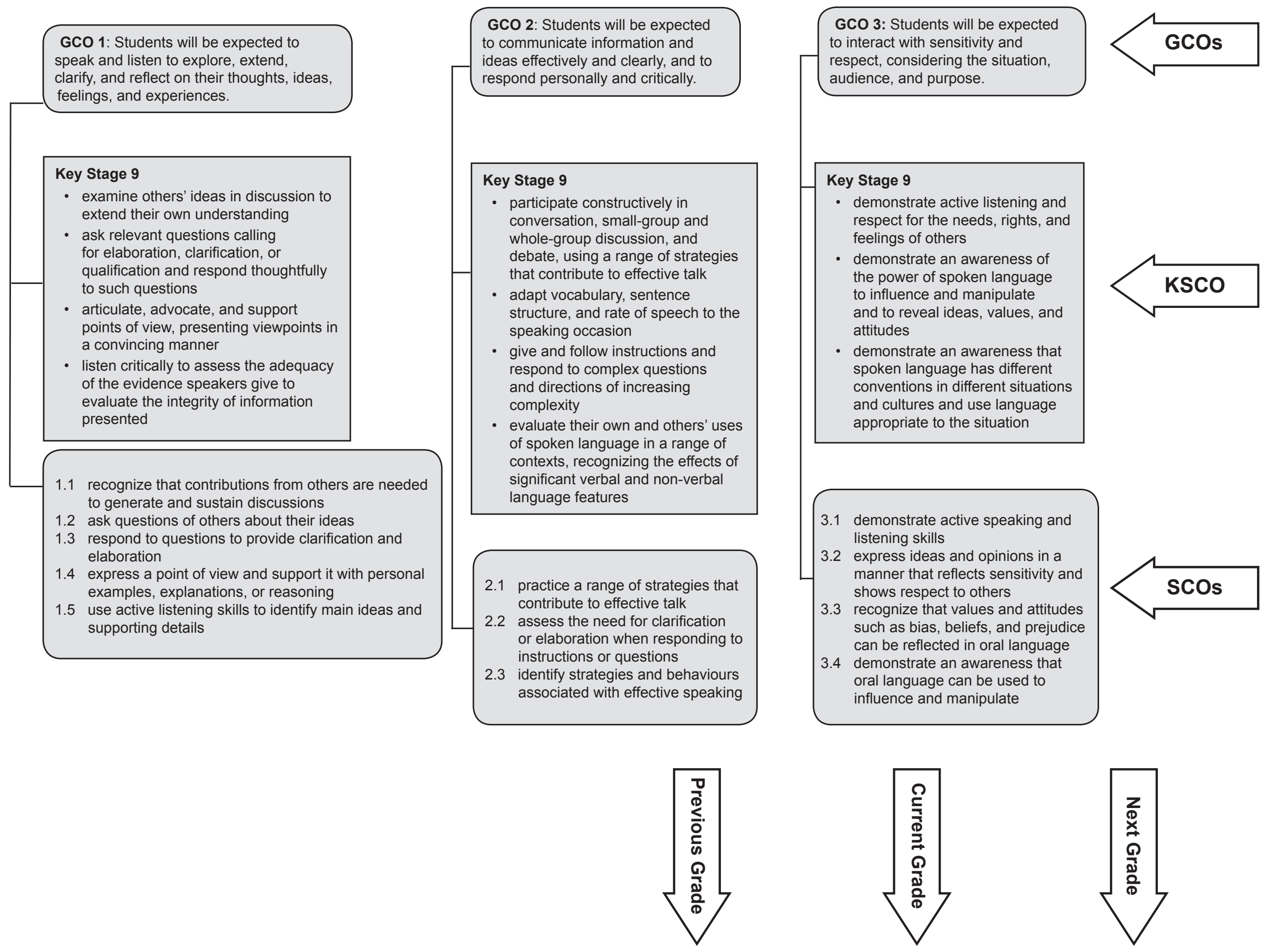
Suggestions for instruction and assessment are organized sequentially:

- Activation – suggestions that may be used to activate prior learning and establish a context for the instruction
- Connection – linking new information and experiences to existing knowledge inside or outside the curriculum area
- Consolidation – synthesizing and making new understandings
- Extension – suggestions that go beyond the scope of the outcome

These suggestions provide opportunities for differentiated learning and assessment.

How to use a Strand overview

At the beginning of each strand grouping there is explanation of the focus for the strand and a flow chart identifying the relevant GCOs, KSCOs and SCOs.



GCO 1: Students will be expected to speak and listen to explore, extend, clarify, and reflect on their thoughts, ideas, feelings, and experiences.

Key Stage 9
- examine others' ideas in discussion to extend their own understanding
- ask relevant questions calling for elaboration, clarification, or qualification and respond thoughtfully to such questions
- articulate, advocate, and support points of view, presenting viewpoints in a convincing manner
- listen critically to assess the adequacy of the evidence speakers give to evaluate the integrity of information presented

1.1 recognize that contributions from others are needed to generate and sustain discussions
1.2 ask questions of others about their ideas
1.3 respond to questions to provide clarification and elaboration
1.4 express a point of view and support it with personal examples, explanations, or reasoning
1.5 use active listening skills to identify main ideas and supporting details

GCO 2: Students will be expected to communicate information and ideas effectively and clearly, and to respond personally and critically.

Key Stage 9
- participate constructively in conversation, small-group and whole-group discussion, and debate, using a range of strategies that contribute to effective talk
- adapt vocabulary, sentence structure, and rate of speech to the speaking occasion
- give and follow instructions and respond to complex questions and directions of increasing complexity
- evaluate their own and others' uses of spoken language in a range of contexts, recognizing the effects of significant verbal and non-verbal language features

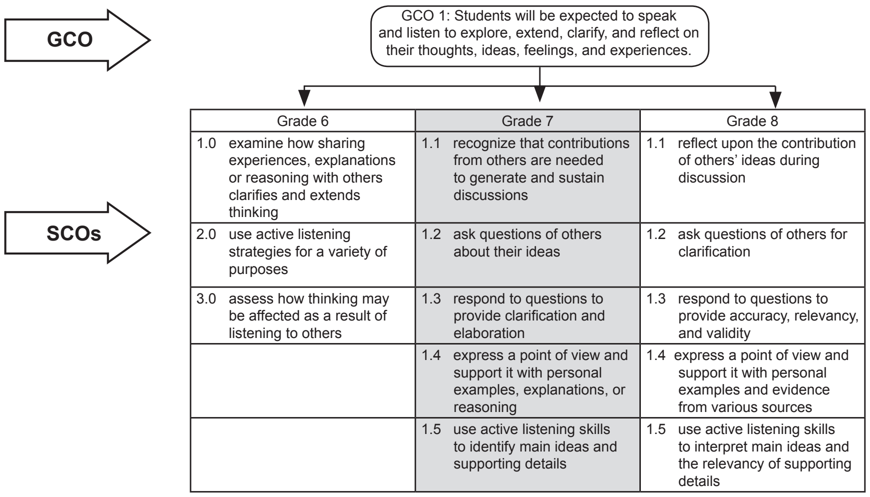
2.1 practice a range of strategies that contribute to effective talk
2.2 assess the need for clarification or elaboration when responding to instructions or questions
2.3 identify strategies and behaviours associated with effective speaking

GCO 3: Students will be expected to interact with sensitivity and respect, considering the situation, audience, and purpose.

Key Stage 9
- demonstrate active listening and respect for the needs, rights, and feelings of others
- demonstrate an awareness of the power of spoken language to influence and manipulate and to reveal ideas, values, and attitudes
- demonstrate an awareness that spoken language has different conventions in different situations and cultures and use language appropriate to the situation

3.1 demonstrate active speaking and listening skills
3.2 express ideas and opinions in a manner that reflects sensitivity and shows respect to others
3.3 recognize that values and attitudes such as bias, beliefs, and prejudice can be reflected in oral language
3.4 demonstrate an awareness that oral language can be used to influence and manipulate

The SCOs Continuum follows the chart to provide context for teaching and assessment for the grade/course in question. The current grade is highlighted in the chart.



GCO 1: Students will be expected to speak and listen to explore, extend, clarify, and reflect on their thoughts, ideas, feelings, and experiences.

Grade 6	Grade 7	Grade 8
1.0 examine how sharing experiences, explanations or reasoning with others clarifies and extends thinking	1.1 recognize that contributions from others are needed to generate and sustain discussions	1.1 reflect upon the contribution of others' ideas during discussion
2.0 use active listening strategies for a variety of purposes	1.2 ask questions of others about their ideas	1.2 ask questions of others for clarification
3.0 assess how thinking may be affected as a result of listening to others	1.3 respond to questions to provide clarification and elaboration	1.3 respond to questions to provide accuracy, relevancy, and validity
	1.4 express a point of view and support it with personal examples, explanations, or reasoning	1.4 express a point of view and support it with personal examples and evidence from various sources
	1.5 use active listening skills to identify main ideas and supporting details	1.5 use active listening skills to interpret main ideas and the relevancy of supporting details

Section Three:

Specific Curriculum Outcomes

Unit 1: Introduction to Computer Science

Focus

In this introductory unit students will explore some of the overarching themes of the course. They will first investigate computational thinking through physical, kinesthetic coding activities. Students will then explore computer systems and how they work to process data. In the last two outcomes of the unit students will explore the fundamentals of digital citizenship and careers associated with the computer science sector of the economy.

Outcomes Framework

GCO 1: Technological Problem Solving: Students will be expected to design, develop, evaluate and articulate technological solutions.

1.0 design a set of instructions to perform a simple task
2.0 perform a set of instructions
3.0 compare kinesthetic coding to concepts in computer programming.
7.0 illustrate how computers compute

GCO 2: Technological Systems: Students will be expected to operate and manage technological systems.

6.0 describe a computer system in terms of input, process, output and feedback
8.0 describe the purpose of operating systems
9.0 evaluate the capabilities of computer hardware
10.0 describe the functions of networking components

GCO 3: History and Evolution of Technology: Students will be expected to demonstrate an understanding of the history and evolution of technology and if its social and cultural implications.

4.0 identify milestones in the evolution of computer science
5.0 discuss the ubiquity of computers

GCO 4: Technology and Careers: Students will be expected to demonstrate an understanding of current and evolving careers and the influence of technology on the nature of work.

12.0 identify career options and trends associated with the computer science field

GCO 5: Technological Responsibility: Students will be expected to demonstrate an understanding of the consequences of their technological choices.

11.0 discuss elements of digital citizenship

SCO Continuum

Unit 1 Introduction to Computer Science

Grade 8 Control Technology	Computer Science 1204
• define the terms system and subsystem and describe examples of each. • trace the evolution of control technologies. • examine the technologies of specific careers and workplaces, including the organizational structures of work environments and the effects of newer technologies.	• describe a computer system in terms of input, process, output and feedback. • evaluate the capabilities of computer hardware. • identify milestones in the evolution of computer science. • identify career options and trends associated with the computer science field.

Suggested Unit Plan

The suggested time for the introductory unit is 15 hours. Approaches to the delivery of the outcomes may vary depending on the teacher's experience and familiarity with the topics. Teachers may want to combine outcomes or treat them in a different order than they are presented. It is important that teachers establish students' prior learning in the area of computer science before beginning this unit.

Getting Started in Computational Thinking

Outcomes

Students will be expected to

1.0 design a set of instructions to perform a simple task [GCO 1]

Focus for Learning

Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a human or machine can effectively carry out. Computational thinking may not have anything to do with computers or digital devices of any kind. This outcome is meant to give students an introduction to computational thinking.

The emphasis here should be on how students can think like a computer. This can be achieved by practicing computational thinking through kinesthetic activities.

Computational thinking is made up of four parts:

- Decomposition: breaking something down, figuring out the parts and how to divide the task
- Pattern recognition: finding similarities and differences, using them to make predictions
- Abstraction: finding general principles that make patterns
- Algorithm design: developing simple steps or rules to solve each of the smaller problems

The teacher should be aware of students' prior knowledge of computational thinking.

Students should complete physical (without electronic computing devices) activities that demonstrate computational thinking. The set of instructions should be documented so as to create an algorithm to complete a specific task.

2.0 perform a set of instructions [GCO 1]

The emphasis of this outcome is on debugging a set of instructions.

Students act as a computer which receives and executes a set of instructions. If there is confusion in the instructions this would be equivalent to a computer error. Students should debug the instructions to fix the error.

Teachers should emphasize the importance of specific and sequential instructions.

Sample Performance Indicator

In pairs, each student creates a set of instructions. Once complete they switch instructions and follow their partners' literally to get to another part of the school and back to the classroom. Once complete, students should debug and re-test each others instructions.

Getting Started in Computational Thinking

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Ask students to design a set of instructions to get them from their desk to the door of the classroom and back.
- Ask students to pick a partner in the class and have the partners discuss their previous experience. Give teams a one minute restriction on discussion.
- Lead discussion relating to students' previous coding experiences.

Connections

Students may

- Complete a journal reflection about their previous coding experiences.
- Take a set of technical instructions for an common appliance and convert the instructions into steps that could be considered an algorithm.

Consolidation

Students may

- Design a set of instructions to perform a task such as:
  - walking to the washroom
  - selecting and sorting one suit from a deck of cards
  - baking muffins
  - singing a song

Resources and Notes

Suggested

EECD Professional Learning Site, Computer Science, Teaching and Learning Strategies, Kinesthetic Coding.

https://www.k12pl.nl.ca/curr/10-12/te/cs1204/teaching-and-learning-strategies/kinesthetic-coding.html

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - CS Fundamentals
  - Computational Thinking

Getting Started in Computational Thinking

Outcomes

Students will be expected to

3.0 compare kinaesthetic coding to concepts in computer programming.

Focus for Learning

Kinaesthetic coding or unplugged activities, refers to any activity that has students physically moving around. In kinesthetic coding students are acting out an algorithm.

This outcome provides an opportunity to introduce the main concepts of computer programming.

- **Sequencing** is the default control structure of a program; instructions are executed one after another unless more advanced features are added to the algorithm.
- **Errors** can be categorized as logic or syntax. In a logic error the instruction may be incorrect or out of sequence for the set of instructions to make sense. In a syntax error, the way the instruction is written may be inaccurate or not descriptive enough.
- **Debugging** is the routine process of locating and removing bugs, errors or abnormalities in the set of instructions.
- **Loops** is a set of instructions that are repeated until some condition is met. This condition stops the loop.
- **Conditionals** are features of a set of instructions that are judged to be true or false based on some condition being met.
- **Function** is the purpose of a specific set of instructions or a subset of instructions in a larger algorithm.

It is important to illustrate to students that computer programs are not magic and cannot think for themselves. They require exact instructions in a logical sequence.

Sample Performance Indicator

Rewrite your list of instructions to get your team to another part of the school and back. Incorporate programming terms and concepts with the goal of making the algorithm more efficient.

Getting Started in Computational Thinking

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide a demonstration to students illustrating how to make instructions more efficient by using a loop instead of writing a set of instructions over several times.
- Reinforce the importance of writing shortcuts into a set of instructions when they are functional.

Connection

Students may

- Complete a journal entry on the problems they encountered when they tried to follow a set of instructions and how the problems were resolved.
- Apply computational thinking elements such as decomposition, pattern recognition, abstraction or algorithm design to organize a set of instructions logically when given a disorganized set of instructions for a task.

Consolidation

Students may

- Work through a debugging process by modifying the instructions.
- Research the connections between logic of kinesthetic coding and how it applies to computer coding. This may cover concepts such as repeating patterns (loops) or decisions (conditional statements).
- Make a poster that displays their set of instructions and highlights key programming concepts.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - Computer science Unplugged
  - Programming concepts
  - Computational Thinking

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 1

Exploring Computing Devices

Outcomes

Students will be expected to

4.0 identify milestones in the evolution of computer science [GCO 3]

Focus for Learning

Computer science is the study of how to manipulate, manage, transform and encode information.

Examples of important milestones may include

- Ada Lovelace creating the first algorithm;
- evolution of the microprocessor;
- development of operating systems;
- evolution of programming languages;
- development of the internet;
- evolution of the Web (Web 1, Web 2, Web 3.0);
- creation of smartphones; and
- evolution of information technology companies such as Google©, IBM©, Microsoft© and Apple©

This outcome is intended to be covered in two to three classes. The focus should be on key events rather than in depth study.

Sample Performance Indicator

Research to identify and describe key milestones in the evolution of computer science.

5.0 discuss the ubiquity of computers [GCO3]

Ubiquity is defined as the fact of appearing everywhere or of being very common. In this regard, ubiquitous computing can occur using any device, in any location, and in any format. A user interacts with the computer, which can exist in many different forms, including laptop computers, tablets and terminals in everyday objects such as a refrigerator or a television.

While reflecting on the ubiquity of computers students should discuss

- global internet connectivity and usage,
- the world wide web,
- smartphones,
- artificial intelligence,
- augmented reality,
- predictive abilities,
- data tracking, and
- the internet of things.

The emphasis should be on classroom discussion and student reflection.

Sample Performance Indicator

Analyze the appliances in your home for the incidence of integrated computing devices.

Exploring Computing Devices

Sample Teaching and Assessment Strategies

Activation

Teachers may
- Provide a list of examples that use computing and/or computers.
- Debate the following statements:
  - Life would be better without the Internet.
  - Data tracking is improving our lives.
  - Internet connectivity is a basic human need.

Connection

Students may
- Identify and describe key milestones in the evolution of computer science.
- Investigate a specific theme and prepare an information display. This could become part of a whole class project connecting all themes. Sharing platforms could include Google Sites© or a traditional wall display.
- Write a journal entry on what their lives would be like without computing devices.

Consolidation

Students may
- In teams prepare multimedia representation of a milestone in the evolution of computer science. In a class presentation of student projects each group teaches the class about a milestone.

Extension

Students may
- Research future trends in computer science.
- Reflect on future developments that could make computing devices even more ubiquitous in your daily life.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html
- Unit 1
  - Computer Science History
  - Ubiquitous Computing

Suggested

Teacher Resource Guide
computerscience@nlesd.ca
- Unit 1

Exploring Computing Devices

Outcomes

Students will be expected to

6.0 describe a computer system in terms of input, process, output and feedback [GCO 2]

Focus for Learning

The systems model of industry includes input, process, output and feedback. Computer hardware refers to the physical parts of a computer system. Computer software refers to the programs that run on the computer system. Software can be grouped as operating systems or applications.

Students should be able to identify the following hardware and software components.

Input:
- keyboard
- mouse
- webcam, camera
- touchpad/touchscreen
- microphone
- sensors (e.g.,temperature, light, switch)

Process:
- random access memory (RAM)
- central processing unit (CPU)
- operating systems such as Windows©, Mac© OSX, iOS, Android©, Linux© and Raspbian©
- application software (i.e., Microsoft Office©, Google Chrome©)

Output:
- monitor
- speakers
- data storage (hard disk drive, solid state drive, SD cards, cloud-based, USB thumb drives)
- printer
- actuators (e.g., DC motors, servo motors, stepper motors, LEDs)

Feedback:
- Was the desired outcome achieved? Adjust if necessary.

This would be a good time to introduce the single board computer such as the Raspberry Pi©. Teachers should differentiate between a single board computer and a traditional computer system.

Sample Performance Indicator

Deconstruct a computing device documenting each component with photographs, and create a report to represent the systems model.

Differentiate between a single board computer and a traditional computer system.

Exploring Computing Devices

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate the components of a computer system and prompt students to identity and describe in the context of the systems model.

Students may

- Dismantle an old computer, documenting components with photos.
- Explain which computer components fit where within the universal systems model of industry.

Students may

- Draw a flow chart illustrating the input, process and output of their favourite app.

Consolidation

Students may

- In teams, devise a role play activity so that each students plays a different device. They can walk around the classroom organizing themselves into input, process and output devices.

Extension

Students may

- Select appropriate hardware or software to match a specific task.
- Create a poster that describes the functions of a single board computer such as a Raspberry Pi.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - How Computers Work

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 1

Exploring Computing Devices

Outcomes

Students will be expected to

7.0 illustrate how computers compute
[GCO 1]

Focus for Learning

The emphasis of this outcome is on data processing and data flow. As an example. One keystroke on the keyboard, for example, creates an electrical pulse. In order to encode this information, it has to be stored in a binary format.

Key terms

- Bits
- Bytes
- Binary

The term “bit” is from the shortening of “binary digit”. It is the basic unit of information for computer systems. A “byte” is eight bits in a single binary word, which allows for up to 256 possible numbers or letters to be stored. One kilobyte (1kB) is 1024 bytes, one megabyte (1MB) is 1024 x 1024 or 1 048 576. One gigabyte (1GB)1024 x 1024 x 1024 or 1 073 741 824 bytes.

This is not meant to be an in-depth examination of digital communications. It is intended to provide a context only.

Sample Performance Indicator

Write your name using the binary number system without a conversion tool.

Exploring Computing Devices

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Discuss with students the significance of the statement, "There are ten (10) kinds of people in the world: those who understand binary, and those who don't."

Connection

Students may

- Use online binary to text conversion tools to translate messages.

Consolidation

Students may

- Use a binary code conversion tool to convert a message into a binary code which other students will decode.
- The message could be conveyed between students by turning a light off and on.

Extension

Students may

- Associate data storage capacities with their everyday use, e.g., the storage capacity of a smartphone and what that means.
- Research the hexadecimal number system and ASCI.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

Unit 1

- Introducing How Computers Work

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

Unit 1

Exploring Computing Devices

Outcomes

Students will be expected to

8.0 describe the purpose of operating systems [GCO 2]

Focus for Learning

An operating system (OS) is computer system software that manages computer hardware and software resources and provides common services for computer programs. The OS contains the drivers that enable the software to recognize and communicate with the hardware. The OS converts input signals to binary and all communication between devices is done as electrical pulses or binary.

The most widely established computer operating systems for traditional computing include

- Apple© Mac OS;
- Google© Chrome OS;
- Linux© (Raspbian, Debian, Ubuntu); and
- Microsoft© Windows

Mobile devices also require an operating system. The most common include

- Apple© iOS,
- Blackberry© OS,
- Google Android, and
- Microsoft© Windows Phone

This is not meant to be an in-depth examination of operating systems. It is intended to illustrate how the OS connects hardware and software.

9.0 evaluate the capabilities of computer hardware [GCO 1]

This outcome is intended to give insight into how hardware components are interconnected within the computer system. Students should understand the purpose and role of each part individually and as a sum of the parts of the system. Various hardware components are more suited for certain tasks than others. For example:

- Small Computer Systems Interface (SCSI) Drive is more robust than a regular hard drive and, therefore, would be more suited to use in network server.
- Solid State Drive (SSD) has no moving parts and is faster than a Hard Disk Drive (HDD).
- Specialized gaming computers require a dedicated video card with upgraded processing capabilities.
- Specialized gaming computers often require upgraded power supply and cooling mechanisms.

There should be opportunity for students to examine computer systems to identify hardware components and troubleshoot problems.

Sample Performance Indicator

Plan out a personal computer by selecting its component capacities. price a computer that is suitable for gaming purposes. Document why you make the decisions you did.

Exploring Computing Devices

Sample Teaching and Assessment Strategies

Connection

Students may

- Create a journal entry describing why they prefer one traditional computer and mobile OS over another. For example, I am a "Mac person because…", I am a "Windows person because ...".
- Research and write a report documenting the evolution and key features of a selected OS.
- Examine advertising flyers or online information from retail stores to identify computer that best matches a specified budget. Students should justify this decision.
- Write a journal entry to describe a time when a personal computing device did not function properly. What was the problem and how was it resolved?

Extension

Students may

- Practice the keyboard shortcuts on traditional and mobile operating systems.
- Compare the features and limitations of various OS.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - Khan Academy
  - How Computers Work

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 1

Exploring Computing Devices

Outcomes

Students will be expected to

10.0 describe the functions of networking components [GCO 2]

Focus for Learning

This outcome is intended to give the student an understanding of basic network components. It is important that students can describe the components of networks they use every day. Even an understanding of the components that bring connectivity to their home is important. The technical knowledge of how information is transmitted through network infrastructure is an important concept in computer science. Common network components include:

- wired and wireless networks,
- wide area network (WAN),
- local area network (LAN),
- web server,
- router,
- switch,
- IP Address,
- web browser,
- web server,
- uniform resource locator (URL), and
- internet service provider (ISP).

It may be a good strategy to break this list of components among pairs of students so that each pair can research and become an expert on that component. With the students acting as the teacher, they can then help their classmates understand the component and how they fit into the system.

Sample Performance Indicator

In teams of two, develop a presentation using a medium of your choice to clearly present the details of a network infrastructure component.

Exploring Computing Devices

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Guide students through a tour of the schools network, highlighting the hardware, software and functionality.

Connection

Students may

- Investigate their home network and create a labeled diagram.

Consolidation

Students may

- Compile a personal glossary of networking concepts which could be linked in some way to the network infrastructure of the school or home.

Extension

Students may

- Using an online website, conduct Internet connectivity / speed tests at school or home.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - Khan Academy

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 1

Careers and Digital Citizenship

Outcomes

Students will be expected to

11.0 discuss elements of digital citizenship [GCO 5]

Focus for Learning

Digital citizenship is the ability to think critically, behave safely, and participate responsibly in the digital world.

The nine main elements of digital citizenship are:

- Communication: using digital devices at the right place and time for the exchange of information
- Law: knowing the rights and restrictions of using digital devices, especially pertaining to copyright law
- Access: ensuring that everyone has access to digital devices and that public services do not assume a level of access that does not exist
- Commerce: ensuring the safe use of e-commerce sites that are legitimate and that protect the personal information of consumers
- Security: the precautions people use to maintain the safety of themselves and others while using networked devices
- Rights and responsibilities: the rights and privileges associated with using digital technologies
- Health and Wellness: the functions of physical and psychological well being related to the use of technology
- Digital Literacy: the capability to use digital technology; select the best tool for the job and use it efficiently to meet needs
- Digital Etiquette: the standards of conduct expected by other digital technology users

Sample Performance Indicator

Create a video “rant” relating to a current issue, such as

- net neutrality,
- regulation of the internet,
- data mining,
- hacking, or
- online security versus freedom.

Careers and Digital Citizenship

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Guide a student discussion on the importance of security while online.
- Discuss recent news events that connect corporations or individuals to unethical online practices.

Connection

Students may

- Keep a journal of the sites they use that may track their data and how it could be used.
- Compile a list of news articles and describe the issue within the context of digital citizenship.

Extension

- Students may experiment with accessing different online retail merchandise to see how personalized advertisements change based on what the Internet deems to be their interests.
- Students may share relevant documentaries or news articles.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - Digital Citizenship
  - Cybercrime

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 1

Careers and Digital Citizenship

Outcomes

Students will be expected to

12.0 identify career options and trends associated with the computer science field [GCO 4]

Focus for Learning

Students should have the opportunity to explore the occupations and labor market trends related to computer science.

Teachers should highlight that there are numerous opportunities for programmers across all sectors of the economy.

The ever changing nature of Information and Communication Technology (ICT) ensures there will be career opportunities for students that do not currently exist. Likewise opportunities for self-employment are constantly evolving in the ICT sector.

Similar to previous outcomes, it is valuable for students to research occupations in the ICT sector and use what they find to provide information to fellow students. This could be considered an efficient way to expose students to as many opportunities as possible. The intent is to spark curiosity in students. It is also recommended that labor market information and trends can be infused into class time throughout the duration of the course especially during the innovation challenge in unit five.

Sample Performance Indicator

Using a career exploration service such as My Blueprint©. Research three careers in the technology cluster. Identify labor market trends that may affect these careers in the future.

Careers and Digital Citizenship

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Discuss how change in technology industries has changed the demand for certain occupations.

Connection

Students may

- Create a graphic to represent the variety of career categories within the ICT industry.
- Research and profile local companies.
- Search for a real computer science related job and create a resume as if to apply for that job.

Consolidation

Students may

- Identify a leading technology and predict jobs that may be created as a result of that technology.

Extension

Students may

- Explore the website of a provincial technology sector organization.
- Interview leaders in the ICT sector.

Resources and Notes

Authorized

My Blueprint

https://myblueprint.ca/

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 1
  - Careers in Tech

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 1

Section Three:

Specific Curriculum Outcomes

Unit 2: Introduction to Programming Concepts (Block-based Techniques)

Focus

In this unit students will explore some introductory concepts in programming by using a block-based programming language. Students will develop best practices in manipulating code through the use of variables, loops and conditional statements. They will learn good planning and troubleshooting skills while approaching programming as an exercise in solution development. Concepts learned in unit two will extend to the development of skills in text-based programming in unit three.

Outcomes Framework

GCO 1: Technological Problem Solving: Students will be expected to design, develop, evaluate and articulate technological solutions.

17.0 write code that uses loops in a block-based programming language
18.0 write code that uses conditional statements in a block-based programming language
19.0 create a solution to an authentic problem using a block-based programming language

GCO 2: Technological Systems: Students will be expected to operate and manage technological systems

14.0 design a concept organizer to describe logical sequences and decision making
15.0 create code that follows a sequential structure using a block-based programming language
16.0 write code that assigns and manipulates variables using a block-based programming language
13.0 employ best practices in a block-based coding environment.
20.0 employ appropriate problem solving skills to identify and correct block-based programming and logic errors

SCO Continuum

Unit 2 Introduction to Programming Concepts

Control Technology 8	Computer Science 1204
• describe the function of specific simple programs • demonstrate an understanding of the importance and usage of flowcharts in program design and development • based on a pre-developed flowchart, write and test a simple program to control a process • using established troubleshooting routines, troubleshoot and test components and systems specific to control technology	• create code that follows a sequential structure using a block-based programming language • design a concept organizer to describe logical sequences and decision making • create a solution to an authentic problem using a block-based programming language • employ appropriate problem solving skills to identify and correct block-based programming and logic errors

Suggested Unit Plan

The suggested time for unit two is 15 hours. Block-based programming is a logical place to begin teaching students the fundamental concepts of programming without getting distracted by learning a new programming language. It is highly recommended that students achieve all outcomes of unit two in the sequence provided; however, as programming becomes more common at the K-6 and 7-9 grades teachers should be more aware of students prior learning. Teachers should assess students' prior learning before deciding on how much time to spend on the outcomes of unit two.

Programming Concepts

Outcomes

Students will be expected to

13.0 employ best practices in a block-based coding environment [GCO 2]

Focus for Learning

Best practices are an integral part of the programming process. Students should adhere to industry standards in coding projects. Professional programmers need to consider the following:

- Planning is necessary so that when programmers begin to write code, they have a clear vision of what they are creating and the general process for building the blocks of code.
- Optimizing code is essential so that it is efficient and requires as few lines as possible.
- Documenting should be practiced within the code to provide details on what the code is doing. Good documentation is essential in all programming projects.
- Simplifying code is very important. Keeping code straight forward means there should be less bugs and it will take less time to debug a piece of code when testing.
- Collaboration between programmers is a common occurrence in the profession. Students should be encouraged to review, comment on and debug each other's code.

Programming Concepts

Sample Teaching and Assessment Strategies

Activitation

Teachers may

- Show students samples of two blocks of code. One that is optimized and one that is not optimized. Discuss the differences between the two.
- Provide students with samples of inefficient code and demonstrate best practices to make it more efficient.
- Discuss code planning in the context of kinesthetic coding activities from unit one.

Consolidation

Students may

- Publish the code they write throughout the unit in a common location so that other students may comment on it.
- Trade completed block based programs with other students so that they can practice optimizing each others code.
- Document their code in designated locations in the block-based programming environment.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Commenting
  - What are the Best Practices

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

14.0 design a concept organizer to describe logical sequences and decision making
[GCO 2]

Focus for Learning

Programmers often use concept organizers, such as flowcharts and/or pseudo-code, to lay out logical ideas, features, processes and sequences that the program should include before writing in an established programming language.

Flowcharts are diagrams that represent algorithms, work-flow or processes. They show the steps in various shapes, connecting them with arrows. The diagram is meant to illustrate a solution to a given problem. Teachers should teach the industry standards for flowcharting. This should include standard flowchart symbols.

Pseudo-code is a detailed, yet readable, description of what a computer program or algorithm must do, expressed in a formally-styled natural language rather than in a programming language. Pseudo-code is sometimes used as a detailed step in the process of developing a program.

Programming is no different from any other project-based process. Students need to develop good practices in planning and documenting simple coding projects.

The teacher should present all activities as an exercise in solution development. It is important that students interpret computer science as primarily an exercise in solution development through many different levels of complexity.

Sample Performance Indicator

Select an everyday task and create a flow chart of steps associated with getting that task successfully completed. Annotations may be used to support your plan.

Programming Concepts

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Identify or plan a computer program and describe its logic using flowcharts.

Connection

Students may

- Create a flowchart illustrating the steps in some common movement or skill (layup in basketball, macarena dance, etc.).

Extension

Students may

- Create a flowchart showing the process of adding two numbers entered by the user.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Concept Maps
  - Flowchart Tutorials
  - Planning

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

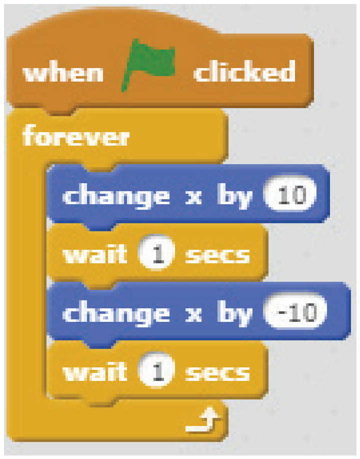
15.0 create code that follows a sequential structure using a block-based programming language [GCO 2]

Focus for Learning

Students should explore and practice the fundamentals of block-based programming.

Block-based programming involves combining blocks of instructions in a drag-and-drop interface. One of the most common block based coding environments is Scratch©

The blocks each contain a small portion of computer code and are usually shaped such that they only fit in a given manner. When the code is run, the groups of instructions are carried out.



```
when ⚑ clicked
forever
    change x by 10
    wait 1 secs
    change x by -10
    wait 1 secs
```

Example from Scratch 2.0

Input at this point can consist of mouse clicks, keyboard inputs or other user interaction on the screen.

Output at this stage can consist of textual or graphical on-screen changes.

Students should explore:

- running a program,
- saving projects,
- events,
- operators,
- inputting text,
- response to input,
- output (text),
- output (graphical),
- parallel programming, and
- editing and modifying code to change the function of the program.

Sample Performance Indicator

Create a program that accepts user input and responds with on-screen output.

Programming Concepts

Sample Teaching and Assessment Strategies

Activitation

Teachers may

- Demonstrate the creation of a simple program in a block-based software.

Connection

Students may

- Complete online skill-builders on the use of a block-based computer programming language.

Consolidation

Students may

- Create a simple animation using a block-based language.
- Create a simple currency converter using a block-based language.

Extension

Students may

- Use an interfacing device with the block-based coding to light an LED.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
    - Rockband Project

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

16.0 write code that assigns and manipulates variables using a block-based programming language
[GCO 2]

Focus for Learning

A variable is a placeholder used to store both numerical and text information. It can change within a program depending on conditions or information passed to the program. Variables can be viewed as containers that simply hold information until it is needed later in a program. Declaring a variable is the process of assigning its type and initial value in the code. There are three common types of variables that programmers will use

- Global variables can be accessed or manipulated by any script, procedure or function within the program. Once they are declared they are available during any point in the run time of the program.
- Local variables are accessed or manipulated by only one script, procedure or function in the computer program. Local variables are used in specific blocks of code and are usually declared at the very beginning of that block.
- Arrays are data structures that contain a group of elements. In most cases these elements are all of the same data type. They could be integers or strings. Arrays are commonly used in computer programs to organize data so that a related set of values can be easily sorted or searched.

A block-based coding environment provides opportunities to discuss variables. Students are not expected to become fluent with the syntax of the code. They only need to select blocks of code and set (declare) the values of the variables. Understanding what variables do will be very important when students code in a text-based programming environment.

Sample Performance Indicator

Create a block of code that stores and manipulates a global or local variable or one that establishes an array.

Programming Concepts

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Discuss why variables are used in a block-based coding environment.
- Discuss and demonstrate different types of variables indicating where or why each would be used in block-based programming language.

Connection

Students may

- Complete skill-building activities on the use of variables in a block-based computer programming language.

Consolidation

Students may

- Create a program that increases or decreases a variable such as a score in a game or counting mouse clicks.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Variables Tutorial
  - Ghostbusters Project

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

17.0 write code that uses loops in a block-based programming language [GCO 1]

Focus for Learning

Students should distinguish between types of loops used in computer programming and be able to select the appropriate loop to accomplish a task.

A loop repeatedly executes code in its body until the loop's conditional statement is met. A loop will continue until some condition either within the code or an event is sensed in a physical interfacing situation. A block of code for example records temperature with a temperature sensor in three minute intervals until the temperature reaches a certain preset level. If the temperature level is never met the condition for stopping the loop will never be met. The program will continue to run. Once the preset temperature value is met, the loop stops running.

A loop is divided into two parts:

- Loop Statement: This defines the value to be true for the continuous loop. The function of this part of the loop depends on the syntax of the conditional statement or the statement that explains the conditions that makes the loop react the way it does.
- Loop Body: This holds the statement's code or instruction; it is is executed with each loop cycle.

Types of loops:

- Definite - A section of code that repeats a set number of times and then stops on its own.
- Indefinite - A section of code that repeats continuously until some condition is met.

Teachers should refer to the kinaesthetic coding activities from unit one for opportunities to illustrate a looping structure.

Sample Performance Indicator

Write a program containing a loop that runs until some condition is met or a certain number of iterations have occurred.

Programming Concepts

Sample Teaching and Assessment Strategies

Activitation

Teachers may

- Demonstrate some common uses of loops in a block-based programming language. This could include everyday activities that use loops to function. Examples could be digital or non-digital in nature.

Connection

Students may

- Reflect on the comparison between loops created in the kinaesthetic coding activities and loops created by a block based programming language.
- Complete skill building activities in the use of loops in a block-based programming language.

Consolidation

Students may

- Create a program counts by multiples of ten using a loop structure.
- Create a program to move a sprite or avatar a specified distance using a loop.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Lost in Space Project
  - Repeat

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

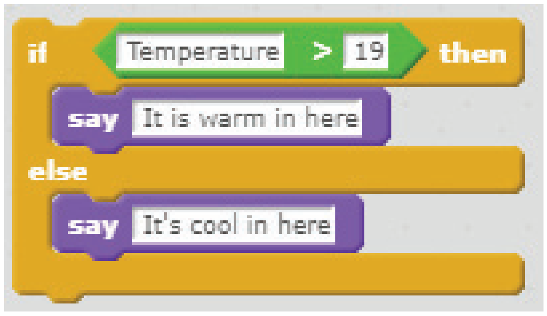
Students will be expected to

18.0 write code that uses conditional statements in a block-based programming language [GCO 1]

Focus for Learning

This outcome is intended to have students alter the output of a program by employing conditional statements in their code.

Conditional statements (if, then, else) are pieces of code that use comparison operators to evaluate whether a given statement is true and then perform some action based on the result. It is typically written in a fashion that checks if one piece of data is greater than, equal to or less than another. If a condition is true, the "then" block is executed. If the condition is not true, the "else" block is executed.



```
if Temperature > 19 then
    say It is warm in here
else
    say It's cool in here
```

Comparison operators (=, <, >) also referred to as relational operators, are used to compare two pieces of data. The data can be number or text strings.

Boolean operators (AND, OR, NOT and XOR), also called logical operators, are used to join two expressions whose outcome will be either True or False.

Sample Performance Indicator

Write a program containing a conditional statement incorporating "if, then" conditions and an "else" result.

Programming Concepts

Sample Teaching and Assessment Strategies

Activitation

Teachers may

- Demonstrate conditional statement using kinesthetic coding from unit one.
- Discuss and demonstrate the use of conditional statements in a block-based programming language.

Connection

Students may

- Reflect on a given conditional statement and explain what the program is doing in that statement.
- Complete skill building activities on conditional statements in a block-based programming language. These activities could be either those embedded in the programming language or those available through resource links.

Consolidation

Students may

- Create a quiz that checks if user input matches the correct answer and gives feedback whether the answer is correct or not.

Students may

- Create a number guessing game using a random number generator. The output will return a response depending on the number entered. i.e., “too high try a lower number, too low try a higher number or “you are correct”.

Extension

Students may

- Create a program with an output such as an alarm sounding if a certain temperature is measured using a physical interface board.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Brain Game Project
  - Scratch
  - Scratch Wiki

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

19.0 create a solution to an authentic problem using a block-based programming language
[GCO 1]

Focus for Learning

Students should create programs that complete a task or solve an authentic problem.

The program should include at least three of the previously discussed programming elements (conditional statements, variables, loops, user input, etc).

This is intended to be a creative project spanning a few classes. Large design challenges should be reserved for unit five.

Computer science is predominantly about solving problems on a variety of scales from declaring a variable in a specific piece of code to digitally controlling the trajectory of a craft reentering the earth's atmosphere. It is very important that teachers and students use the language appropriate to the profession. Students need to be aware that every time they are working on code, they are in actual fact creating a solution. They should consider themselves solution providers.

Teachers should verify the originality of students' code. While borrowing code from other sources is common practice in the coding business, it is important that students also learn to code from scratch.

Sample Performance Indicator

- Create a program that calculates a 15 percent gratuity from the total restaurant bill.
- Create a program that completes some calculation and follows the pattern of input, process and output.

Programming Concepts

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Review an authentic problem with students and brainstorm solutions that can be applied using block-based programming. This could include:
  - calculating Ohm's Law;
  - making an alarm sound when a temperature is reached; or
  - making an LED flash when moisture is sensed.

Connection

Students may

- Duplicate the functionality of an existing/sample block-based program.

Extension

Students may

- Use an interface board to prototype a physical button and a buzzer for an alarm.

Resources and Notes

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 2

Programming Concepts

Outcomes

Students will be expected to

20.0 employ appropriate problem solving skills to identify and correct block-based programming and logic errors
[GCO 2]

Focus for Learning

Students should demonstrate the ability to troubleshoot, identify and correct programming errors. Error handling happens at two points in the coding process

Planning: When the program is being planned, programmers should be keenly aware of how the end user could possibly cause an error when manipulating the interface. Programmers should anticipate user input when selecting structures for coding projects. Troubleshooting is sometimes about minimizing the potential for problems.

Debugging: When the program has been completed and the programmer runs it for the first time the debugging process starts. Errors in code generally fall into the following categories:

- Syntax errors: Errors due to the fact that the syntax of the programming language is not used properly.
- Semantic errors: Errors caused by the improper use of the programming statements.
- Logical errors: Errors due to the fact that blocks of code are not used properly.

This outcome should be integrated with previous outcomes. Good troubleshooting strategies should be employed throughout this course when working with code. Teachers should model good troubleshooting strategies with students.

Sample Performance Indicator

Reflect on past coding activities you have completed. Select one of these activities and indicate how you employed troubleshooting techniques to make a program work properly.

Programming Concepts

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate efficient troubleshooting with code that does not work.

Connection

Students may

- Debug their own code as issues arise.

Consolidation

Students may

- Be provided with a program with errors and debug the code.

Extension

Students may

- Introduce an error into their program and have a classmate attempt to debug it.

Students may

- Analyze a sample program that has more blocks than necessary and work to improve the efficiency or structure using loops or variables.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
    - Debugging Scripts
    - Debugging Challenges

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 2

Section Three: Specific Curriculum Outcomes

Unit 3: Manipulate Text Based Code

Focus

In this unit students will apply skills and knowledge from unit two to develop programs in a text-based programming language. Students will reinforce skills in using best practices and manipulating code through the use of variables, loops and conditional statements. Students will continue to use good planning and troubleshooting skills while approaching programming as an exercise in solution development. Concepts learned in text based programming in unit three will extend to the development of skills in physical interfacing in unit four.

Outcomes Framework

GCO 1: Technological Problem Solving: Students will be expected to design, develop, evaluate and articulate technological solutions.

27.0 write code that uses conditional statements in a text-based coding environment
26.0 write code that uses loops in a text-based coding environment
29.0 employ appropriate debugging skills to identify and correct programming and logic errors in a text-based coding environment
31.0 create a solution to an authentic problem using a text-based programming language

GCO 2: Technological Systems: Students will be expected to operate and manage technological systems.

22.0 describe the relationship between source code, compiler and machine language
23.0 modify code that follows a sequential structure in a text-based coding environment
24.0 create an introductory program in a text-based programming language
25.0 write code that assigns and manipulates variables using text-based programming language.
28.0 employ best practices in a text-based coding environment
30.0 design a concept organizer to plan logical sequences and decision making in a text-based program

SCO Continuum

Unit 3 Manipulate Text Based Code

Control Technology 8	Computer Science 1204
• describe the function of specific simple programs • demonstrate an understanding of the importance and usage of flowcharts in program design and development • based on a pre-developed flowchart, write and test a simple program to control a process • using established troubleshooting routines, troubleshoot and test components and systems specific to control technology	• create a solution to an authentic problem using a text-based programming language • design a concept organizer to plan logical sequences and decision making in a text-based program • employ appropriate debugging skills to identify and correct programming and logic errors in a text-based coding environment

Suggested Unit Plan

The suggested time for unit three is 25 hours. Text-based programming is a logical advancement after students have been introduced to programming in a block based environment in unit two. It is highly recommended that students engage in all the outcomes of unit three in the order they are provided. This unit provides an opportunity for students to learn text based programming skills before they begin programming physical interfaces in unit four.

Introduction to Text Based Programming

Outcomes

Students will be expected to

21.0 Compare strengths and industry uses of common programming languages. [GCO 2]

Focus for Learning

Teachers should begin this unit with a discussion of the limitations of block-based programming. Block-based programming was developed to teach programming concepts to youth but is not powerful enough for many real world applications. The computer science industry uses a variety of text-based programming languages for more advanced applications.

Each programming language has strengths and specific industry applications. Teachers should give an overview of common text-based programming languages:

- Java© is one of the most widely used programming languages in the world. It is versatile across common operating systems, free and powerful. Java is commonly used for the development of Internet content, applications, server software and games.
- Python© is a common programming language that uses syntax similar to English to create commands. Python can be used on multiple operating systems and is quite often used in industry for the development of applications, management of large databases and system scripting.
- C++© is a general purpose object-oriented programming language. It is one of the most popular languages and can be used on a variety of operating systems. It is primarily used in industry for developing application software, coding drivers, and management of client-server applications. C++ is quite often used to introduce coding, and in coding competitions.

Students should understand that it is important to select a programming language to learn the skills of programming. Students need to be immersed in the process of manipulating the specific functionality of a text-based programming language. Once a student learns to be productive in one text-based language, their skills can be applied quite easily to any programming language. For the purposes of this course, Python© is recommended as the text-based programming language.

Sample Performance Indicator

Research and write a comparison of two industry-standard programming languages.

Introduction to Text Based Programming

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide examples of completed programs from three different programming languages.

Connection

Students may

- Research an innovative product and establish which programming language was used to develop that product.

Consolidation

Students may

- Prepare for and engage in a class debate where the subject is the applications and benefits of a variety of programming languages.
- Research one programming language and present strengths and industry uses to the rest of the class using a medium of choice.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - Comparison of Programming Languages
  - 83 Tips for Getting up to Speed
  - 10 Programming Languages

Teacher Resource Guide computerscience@nlesd.ca

- Unit 3

Introduction to Text Based Programming

Outcomes

Students will be expected to

22.0 describe the relationship between source code, compiler and machine language
[GCO 2]

Focus for Learning

Source code refers to the instructions written by the programmer. It could be written in any programming language. Source code is turned into machine language (binary) by a compiler so that a computer can execute the commands.

Teachers should explain that computers do not directly use human language or programming languages. They use binary (1s and 0s) only.

Key terms:

- Source code is the main component of a computer program. It can be read easily with basic English language.
- A compiler is a program that processes source code written in a programming language and turns it into statements that the computer processor can manipulate.
- Machine language is the language of computing devices. Source code is processed through a compiler to create the language needed by the central processing unit. Machine language uses the binary number system.

An integrated development environment (IDE) is a suite of applications that consolidates basic tools for creating, testing and compiling source code.

Sample Performance indicator

Create a graphic organizer to illustrate the relationship between source code, compiler and machine language.

Introduction to Text Based Programming

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate the process of compiling source code into machine language using a text-based programming language.
- Provide students with samples of source code and machine language.

Consolidation

Students May

- Participate in a role play activity where each member of a team takes on the role and acts out the attributes of either source code, compiler or machine code.
- Use multimedia of their choice to create a product that explains the relationship between source code, compiler or machine code.

Extension

Students may

- Write a small program in the text editor of any windows based personal computer.
- Use online text to binary converters to create binary messages and send to other students to decipher.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - How Computers Work
  - Learn To Code With Me

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Introduction to Text Based Programming

Outcomes

Students will be expected to

23.0 modify code that follows a sequential structure in a text-based coding environment [GCO 2]

Focus for Learning

Students should modify existing code to see how changes to the code change the output. The goal is to get students manipulating code quickly by learning the syntax gradually.

Students should begin by experimenting with common programming code. They should change the code of simple programs such as "Hello World." Students should move on to modifying more elaborate programs provided by the teacher.

Error analysis and problem solving are a key part of learning how to code. When students begin coding in a text-based environment they will likely make errors in code syntax or in the misspelling of a variable function name.

Emphasis should be placed on student involvement in text-based programming. For many of the students this will be their first introduction to a text-based programming language.

Introduction to Text Based Programming

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide students with examples of code that have a bug and the same code with the bug fixed. Students can compare the code and discuss the changes

Connection

Students may

- Modify existing code to explore the functionality of the programming language.

Extension

Students may

- Create code without the aid of previously created code.
- Research key terms related to text based programming languages, such as Python.

Resources and Notes

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

24.0 create an introductory program in a text-based programming language [GCO 2]

Focus for Learning

Students should begin writing code in its simplest form. They may use ideas from unit two to help them.

Teachers may introduce text-based programming using programs such as Turtle programming. This is a learning tool that is available for most programming languages.

Key terms:

- Print statement
- String

A print statement tells the machine to output data. Coding 5+3, for example will not return a value of 8 to the user unless the programmer uses a print command.

A string is a word or phrase that is entered into a program. It can be modified but mathematical operations on it will return an error. Most training for programming languages begins by getting the learner to print the string “Hello World.”

Teachers may find that for some students this is review and for other students this is their first experience manipulating source code.

Sample Performance Indicator

In a text-based programming language, create an introductory program that outputs a message to the user.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide examples of simple programs that complete a mathematical equation or print a message to the end user.

Connection

Students may

- Print a string so the compiler outputs a statement of their choosing.
- Develop code that outputs the solution to a simple mathematics equation.

Resources and Notes

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

25.0 write code that assigns and manipulates variables using a text-based programming language [GCO 2]

Focus for Learning

A variable is a placeholder used to store both numerical and text information. It can change within a program depending on conditions or information passed to the program. Variables can be viewed as containers that simply hold information until it is needed later in a program.

Declaring a variable is the process of assigning its type and initial value in the code. There are three common types of variables that programmers will use:

- Global variable can be accessed or manipulated by any script, procedure or function within the program. Once they are declared they are available during any point in the run time of the program.
- Local variables are accessed or manipulated by only one script, procedure or function in the computer program. Local variables are used in a specific block of code and are usually declared at the very beginning of that block.
- Arrays are data structures that contain a group of elements. In most cases these elements are all of the same data type. They could be an integer or string. Arrays are commonly used in computer programs to organize data so that a related set of values can be easily sorted or searched.

The teacher should review variables, global variable and local variables from unit two. The same principles apply; however, in a text based environment the programmer has more control over the variables.

There are a few common guidelines when manipulating variables:

- Use proper naming conventions, such as no numbers in names.
- The variable names should be short but descriptive.
- Variables cannot use software specific command words.

Sample Performance Indicator

In a text-based programming language, create an introductory program that assigns and manipulates variables.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Refer back to the block-based coding in unit two to compare and contrast variable manipulation in a text-based coding environment.
- Provide students with snippets of code that use variables and explain the purpose of each.

Connection

Students may

- Write a program that converts lowercase letters in a string variable to uppercase.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - Python Variables
  - Python Tutorials

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

26.0 write code that uses loops in a text-based coding environment [GCO 1]

Focus for Learning

One of the basic building blocks of computer science is the ability to do things repeatedly or loop. In computer science, this repetitive idea is referred to as iteration. In this section, students should explore some mechanisms for basic iteration.

A loop repeatedly executes code in its body until some condition is met. A loop is divided into two parts:

- Loop Statement: This defines the value to be true for the continuous loop. The function of this part of the loop depends on the syntax of the conditional statement or the statement that explains the conditions that makes the loop react the way it does.
- Loop Body: This holds the statement's code or instruction; it is is executed with each loop cycle.

There are two types of loops:

- Definite - A section of code that repeats a set number of times and then stops on its own.
- Indefinite - A section of code that repeats continuously until some condition is met.

There is an opportunity here to reinforce the discussion of looping from unit two. It is important for students to understand that, while, the fundamentals are the same, looping structures in this unit concentrate on a text-based programming language.

Key terms:

- For...next loop: Repeats a group of statements a specified number of times.
- Do...while loop: This is a control flow statement that executes a piece of code at least once, and then repeatedly executes the code, or not, depending on condition at the end of the block.
- Conditional loop: This is a way to repeat something until a condition is met. This would include sensing something in a physical interfacing environment.

Any operations that are carried out in a loop must be indented so the compiler knows what is and what isn't in the loop.

Sample Performance Indicator

In a text-based programming language, create an introductory program that uses loops.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Refer back to kinesthetic coding activities from unit one. Provide possible examples of loops in every day decision making.
- Provide students with snippets of code that use loops and explain the purpose of each.

Connection

Students may

- Create a program that uses various types of loops at the same time. This could include a video game.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - Python While Loops
  - Python For Loops
  - Python Tutorial

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

27.0 write code that uses conditional statements in a text-based coding environment [GCO 1]

Focus for Learning

Students should alter the output of a program by employing conditional statements in their code.

Conditional statements (if, then, else) are pieces of code that use comparison operators to evaluate whether a given statement is true and then perform some action based on the result. It is usually written in a way that checks if one piece of data is greater than, equal to or less than another. If a condition is true, the "then" block is executed. If the condition is not true, the "else" block is executed.

Conditionals are important in order for a program to make decisions. Without conditionals a program would do the same thing every time it runs. The key terms to control conditional statements are: if, else, and else if.

The discussion of conditional statements from unit two can be revisited here. Students should realize that while the fundamentals are the same, conditional statements in this unit concentrate on a text -based programming language.

Conditional statements require operations to be indented in the block of code.

Sample Performance Indicator

In a text-based programming language create an introductory program that uses conditional statements

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Refer back to kinesthetic coding activities from unit one. Provide possible examples of conditional statements in every day decision making.
- Provide students with snippets of code that use conditionals and explain the purpose of each.

Connection

Students may

- Create a rock paper scissors game or a multiple choice quiz, using a variety of conditional statements.

Extension

Students may

- Create a graphic organizer illustrating how a conditional statement would be useful in a physical interfacing coding challenge.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - Python If....Else
  - Python Course

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

28.0 employ best practices in a text-based coding environment [GCO 2]

Focus for Learning

Best practices are an integral part of the programming process. Every effort should be made to ensure students adhere to industry standards in coding projects. Teachers need to take every opportunity to reinforce attributes of professional programmers. Professional programmers need to consider the following:

- Planning is necessary so that when programmers begin to write code, they have a clear vision of what they are creating and the general process for building the blocks of code.
- Optimizing code is essential so that it is efficient and requires as few lines as possible.
- Documenting should be practiced within the code to provide details on what the code is doing. Good documentation is essential in all programming projects.
- Simplifying code is very important. Keeping code straightforward means there should be less bugs and it will take less time to debug a piece of code when testing.
- Collaboration between programmers is a common occurrence in the profession. Students should be encouraged to review, comment on and debug each others code.

Students were introduced to best practices in unit two. It is important that while the fundamentals are the same, best practices in this unit concentrate on a text-based programming language. Teachers should have high expectations for students for using best practices. Unlike block-based coding, planning, optimizing and simplifying code is an essential process in a text-based environment. Professional programmers are meticulous planners with an eye for efficiency.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate poorly written and efficiently written code.
- Demonstrate the use of creator commenting.

Connection

Students may

- Analyze a piece of poorly written code and make suggestions on how to make it more efficient.
- Practice creating efficient code and writing commenting notes.

Consolidation

Students may

- Write a small program without comments and exchange it with another student. Students can then try to interpret what the program is supposed to accomplish.

Extension

Students may

- Find samples of commenting notes in coding.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - Best of the Best Practices

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

29.0 employ appropriate debugging skills to identify and correct programming and logic errors in a text-based coding environment [GCO 1]

Focus for Learning

A compiler will translate programming language into machine code that the computer processor can understand. If there are errors in the code, the compiler will not understand how to translate it and it will report an error.

A debugging tool will report the line of code where the error exists. It will also give an error code. Though the messages can be somewhat cryptic, students should develop an understanding for the types of errors and their associated types of solutions.

Errors are generally of three types: logic, semantic and syntax.

- A logic error occurs when the idea that is being programmed doesn't make sense.
- A semantic error is when the code does not match the idea.
- A syntax error means the code was not typed correctly.

Syntax and semantic errors will usually be caught by the debugging tool.

Debugging tools are essential for programming. Students will get the hands on experience of interpreting the messages from debugging tools and will over time find them useful in fixing programming errors. It is important that teachers represent debugging and related problem solving as a routine step in the programming process.

Sample Performance Indicator

Interpret five error messages from a debugging program. Make suggestions on how the problem can be solved.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide students with snippets of code that exemplify each type of error.

Connection

Students may

- Debug sample code provided by the teacher.
- Use the debugging process while they write code.
- Work collaboratively to debug each others code.
- Add errors to samples of code for other students to find.

Extension

Students may

- Create a coding troubleshooting guide.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 3
  - 10 Common Mistakes
  - 29 Common Programming Errors

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Writing a Text Based Program

Outcomes

Students will be expected to

30.0 design a concept organizer to plan logical sequences and decision making in a text-based program [GCO 2]

Focus for Learning

It is common to use flowcharts and/or pseudo-code to lay out logical ideas, features, processes and sequences that the program should include before writing in an established programming language. Block-based programs share many qualities with concept organizers but text-based programs do not. For this reason, it is particularly important to create a concept organizer for text-based programs.

Flowcharts are diagrams that represent algorithms, work-flow or processes. They show the steps in various shapes, connecting them with arrows. The diagram is meant to illustrate a solution to a given problem. Teachers should teach the industry standards for flowcharting. This should include standard flowchart symbols.

Pseudo-code is a detailed yet readable description of what a computer program or algorithm must do, expressed in a formally-styled natural language rather than in a programming language. Pseudo-code is sometimes used as a detailed step in the process of developing a program.

Text-based programs can become very complicated and students will need a road map to keep track of their progress. A concept organizer will give students a template to follow in order to accomplish their programs' goals.

The teacher should present all activities as an exercise in solution development. It is important that students interpret computer science as primarily an exercise in solution development through many different levels of complexity.

Writing a Text Based Program

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide students with a computer programming problem and challenge them to create a plan for the solution.

Connection

Students may

- Use pseudocode, flowchart or a block-based program to plan their code and use a text-based programming tool to code.
- Create a concept organizer for a given text-based program.

Resources and Notes

Suggested

Resource LInks: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 2
  - Concept Maps
  - Advanced Organizers

Suggested

Teacher Resource Guide: computerscience.nlesd.ca

- Unit 3

Developing Solutions

Outcomes

Students will be expected to

31.0 create a solution to an authentic problem using a text-based programming language
[GCO 1]

Focus for Learning

Students should create programs that complete a task or solve a problem using a text-based programming language.

The program should include at least three of the previously discussed programming elements. This is intended as a minor creative project spanning a few classes. Large design challenges should be reserved for unit five.

Teachers should verify that students understand the code they are writing by having them annotate what each part of their program does. While it is acceptable for students to borrow small sections of code from other programmers, it is also very important that students learn the programming language for themselves. Writing a program should not be a cut and paste exercise.

Sample Performance Indicator

Using a text-based programming language create a program that makes common calculations as how much taxes is built into the price per litre of gasoline at a local service station.

Developing Solutions

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Review a real world problem with students and brainstorm solutions that can be applied using text-based programming. i.e., calculating the average speed necessary to travel a specified distance or calculating the average gas mileage of a car based on the reading from the car's tachometer.

Connection

Students may

- Duplicate the functionality of an existing/sample text-based program.

Consolidation

Students may

- Create a program that calculates the gratuity from a restaurant bill.

Extension

Students may

- Create a program that outputs a printable chart, such as Celsius to Fahrenheit conversion table.

Resources and Notes

Suggested

Teacher Resource Guide:
computerscience.nlesd.ca

- Unit 3

Section Three:

Specific Curriculum Outcomes

Unit 4: Coding Interfaces

Focus

In this unit students will apply skills and knowledge from units one, two and three to program software and hardware interfaces. Students will explore the capabilities and limitations of a variety of hardware resources. They will experiment with coding input and output devices with code they write themselves. They will also get their first experiences in solving authentic problems using sensing and control. Concepts learned in unit four will support the innovation challenge in unit five.

Outcomes Framework

GCO 1: Technological Problem Solving: Students will be expected to design, develop, evaluate and articulate technological solutions.

29.0 design a GUI for a specified purpose
33.0 employ coding concepts that detect sensor inputs through a physical interfacing device
34.0 employ coding concepts to control outputs through a physical interfacing device
35.0 create a prototype that employs sensing and control events

GCO 2: Technological Systems: Students will be expected to operate and manage technological systems.

28.0 explain the purpose of a graphical user interface (GUI)
30.0 differentiate between hardware and software interfacing
31.0 identify input and output devices associated with physical interfacing

GCO 5: Technological Responsibility: Students will be expected to demonstrate an understanding of the consequences of their technological choices.

32.0 employ safety measures and practices

SCO Continuum

Unit 4 Coding Interfaces

Control Technology 8	Computer Science 1204
• define control in terms of the processes sensing, switching and regulating • identify components of a simple electronic control system and describe each of their functions • identify the health and safety hazards associated with specific control technology systems	• differentiate between hardware and software interfacing • identify input and output devices associated with physical interfacing • employ safety measures and practices

Suggested Unit Plan

The suggested time for unit four is 25 hours. Interfacing is a logical advancement after students have been introduced to programming in a block-based environment in unit two and in a text-based environment in Unit three. It is highly recommended that students engage in all the outcomes of unit four in the order they are provided. This unit provides an opportunity for students to explore the hardware resources provided with the course. There will be opportunities for students to experiment with input and output devices so that they will develop a comfort level that will facilitate the innovation challenge in unit five.

Introduction to Interfacing

Outcomes

Students will be expected to

28.0 explain the purpose of a graphical user interface (GUI)
[GCO 2]

Focus for Learning

Interface can be interpreted as a surface forming the boundary between two things. In computer science these interfaces are associated with forming the connection between the machine language that runs the electronics and the user. Students should have the opportunity to reflect on the concept of interfaces first by exploring a variety of interfaces:

- A Command Line Interface (CLI) requires commands be manually typed directly into the computer’s disk operating system.
- A Graphical User Interface (GUI) is the most common user interface. It is graphical with user friendly images that can be manipulated by a mouse or other pointing device.
- A Menu Drivin Interface (MDI) is commonly used on cash machines where the user has to select from a series of menus by touching buttons on the kiosk or on a touch screen.
- A Form Based Interface (FBI) uses a form of fields’ such as drop down menus, radio buttons or check boxes so that the usr can enter data quickly and uniformly.
- A Natural Language Interface (NLI) interprets spoken words so that it can perform commands based on what the user asks it to do.

Students can then explore a variety of GUIs. A GUI allows users to interact with computers through graphical objects and visual indicators, as opposed to a strictly textual input and output. GUIs were developed to make computer use easier. They have advanced from requiring a device such as a mouse to manipulate graphics to the latest touch technologies found in mobile telephones.

In this course interfaces will include a collection of hardware, the operating system and software components. Each is designed to perform specific functions in this interface layer.

Sample Performance Indicator

Identify a common GUI that you use on a daily basis. Reflect on its practical use and how it allows you to interact with a computing device.

Introduction to Interfacing

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Identify common elements among user interfaces.
- Provide examples of various interfaces and discuss their applications.

Connection

Students may

- Compare command line and GUI versions of programs that serve similar tasks. A good example of this could be utilities on a Raspberry Pi that can be manipulated both in the Raspbian GUI and in the command line terminal window.

Students may

- Make a comparison between copying a file using a file explorer and a command line copy.

Consolidation

Students may

- Maintain a daily log documenting the types of interfaces they use and what they are using them for.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Rationale for GUI
  - Advantages and disadvantages

Suggested

Teacher Resource Guide computerscience@nlesd.ca

- Unit 4

Introduction to Interfacing

Outcomes

Students will be expected to

29.0 design a GUI for a specified purpose
[GCO 1]

Focus for Learning

Students should explore aspects of GUI design. Visual design is the art of presenting something visually. The visual presentation usually has a purpose and a function. In interface design, the designer wants the user to quickly and easily understand the purpose and function of the design. A vehicle instrument panel, a remote control layout, or a piece of software are all designed in a purposeful way, following the elements and principles of design.

The elements of design include:

- line, shape, form, texture; and
- color, direction, size, value.

The principles of visual design determine best practices for combining the elements into a particular visual arrangement. They are:

- balance, contrast, proportion;
- Ppattern, gradation, proximity; and
- alignment, repetition, unity.

Sample Performance Indicator

Customize the basic layout of a GUI, to meet the needs of a program written in unit three.

Introduction to Interfacing

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Lead a discussion about common user interfaces (e.g.,For example, the dashboard of a snowmobile).
- Provide a sample of a basic GUI for students to customize.

Consolidation

Students may

- Determine the features that their GUI needs to have based on the functions that the program performs.
- Design an interface that best compliments the function of that program, given the code from a simple program.

Resources and Notes

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 4

Introduction to Interfacing

Outcomes

Students will be expected to

30.0 differentiate between hardware and software interfacing [GCO 2]

Focus for Learning

An interface is a shared connection between two or more components of a computer system that exchange information. The connection can be between software, computer hardware, peripheral devices, humans and combinations of these.

A hardware interface is a physical component used as an input or output device where a software interface is a virtual input or output device. A keyboard is a hardware interface, whereas a scroll bar on the side of a document is a software interface.

In this course, students will build their own custom software interfaces. These software interfaces should meet the needs of the prototype they create.

The components used to create the hardware interfaces have been carefully selected so that students can have as many options as possible for creating innovative prototypes in unit five.

Sample Performance Indicator

Create a personal log for two to five days tracking each time you use an interface of any kind. Organize your list into software and hardware interfaces.

Introduction to Interfacing

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide examples of various systems and subsystems of a computer and discuss whether they are hardware or software interfaces or a combination of both.

Connection

Students may

- Select one room in the school and document the software and hardware interfaces in that room.

Consolidation

Students may

- Describe a system that utilizes both hardware and software interfaces (e.g., a security alarm).

Extension

Students may

- Create a graphical user interface (GUI) that can be used to control a motor.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Note: The supplementary resources for hardware interfacing have been carefully selected and tested. If teachers wish to introduce other hardware resources they should be careful to research the component to verify that it is compatible with the components provided. Failure to do this could cause permanent failure of the components provided with the course.

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4

Interface Computing

Teacher Resource Guide computerscience@nlesd.ca

- Unit 4

Introduction to Interfacing

Outcomes

Students will be expected to

31.0 identify input and output devices associated with physical interfacing [GCO 2]

Focus for Learning

General Purpose Input and Output (GPIO) are pins on a computing device, such as the Raspberry Pi©, that can be used to connect to devices to receive data (input) or to control devices (output). Manipulating the GPIO pins individually can be risky for students and could lead to permanent damage to the computing device being used. For the purposes of this course, an Explorer Hat Pro© has been supplied with the Raspberry Pi©. This hat maps the GPIO pins to minimize the risk of permanent damage of to the Raspberry Pi©.

Teachers should introduce the devices that are included with the course resources.

Supplied input devices include but are not limited to:

- Keyboard/mouse
- Switch
- Ultrasonic Range Finder
- Photocell
- Moisture Sensor
- Passive Infrared Sensor (PIR)

Supplied output devices include but are not limited to:

- Light Emitting Diode (LED)
- Speaker/buzzer
- DC gear motor
- Servo motor

Sample Performance Indicator

- Take photos of the components supplied for the course. create a web-page that labels the components and indicate if they are an input or an output device.
- As you experiment with components log any new information about the component on your webpage. This could include:
  - links to drivers,
  - operating limitations,
  - scripts for configuration and
  - code to make the component operate.
- The digital collection of information on input and output devices will become your personal reference tool for unit four and five.

Introduction to Interfacing

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Discuss common input and output devices and their differences.

Connection

Students may

- Keep a log of input and output devices they use in a two to five day period. Categorize these devices as software and hardware interfaces.
- Create a concept map of input and output devices associated with physical interfacing.

Extension

Students may

- Design a system that solves a problem by researching what input and output devices could be used.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Note: It is not recommended that the Raspberry Pi© GPIO pins be manipulated by students. The Explorer Hat Pro© has been resourced to map the GPIO pins and guard against damage to the devices.

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Using Sensors with a Raspberry Pi

Teacher Resource Guide computerscience@nlesd.ca

- Unit 4

Introduction to Interfacing

Outcomes

Students will be expected to

32.0 employ safety measures and practices [GCO 5]

Focus for Learning

Electricity of any amount can be dangerous. As such, safety practices must be given priority. Teachers should model good health and safety rules and standards, while developing a culture of safety with students in classrooms and labs.

Although the devices are low voltage they are susceptible to voltage or current beyond their specification. While, it is unlikely that capacities will be exceeded with the components supplied for the course, there is always a risk of injury. For this reason having students wear safety glasses when working with discrete components is essential. An example of this is an LED. It is common for an LED to be over powered and pop. For the purposes of this course, discreet LEDs are not resourced. Instead an LED brick is provided. This component has overload protection built in.

Sample Performance Indicator

- Identify potential hazards in the classroom and describe how to control each hazard.
- Record any safety hazards for any of the supplied components.

Introduction to Interfacing

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Review general classroom safety rules and safe practices associated with all necessary devices and tools.
- Test students on their knowledge of appropriate safety practices.

Connection

Students may

- Create a series of safety info-graphics to be posted around the classroom.

Consolidation

Students may

- Take on the responsibility of providing a safety moment at the beginning of each class. They can present the information in a medium of their choice.

Extension

Students may

- Create a video infomercial on good safety practices specific to activities for this course.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Electronics Safety
  - Raspberry Pi Safety
  - Light Emitting Diodes
  - Rpi Power Limitations
  - Rpi GPIO information

Teacher Resource Guide
computerscience@nlesd.ca

Unit 4

Sensing and Control

Outcomes

Students will be expected to

33.0 employ coding concepts that detect sensor inputs through a physical interfacing device [GCO 1]

Focus for Learning

At this point it should be clear to students that hardware components all need to have some sort of code to operate. That code can either be python code that processes the data from a sensor or a custom driver or library from the component manufacturer to make the component work.

Sensors are either analog or digital:

- Digital Input: The interface component can detect a simple change in an electrical voltage. It can detect if a switch is on or off. An example of this could be a momentary push-button reading 0v until the contacts touch, at which point the reading becomes 5v.
- Analog Input: The interface device can detect a continuously changing electrical voltage coming from sensors which may be responding to changing parameters, such as light, sound, or moisture. The changing voltage levels are converted to digital information to be manipulated and stored in the computer (e.g., a cadmium sulfide photocell changes resistance depending on the amount of light hitting its surface).

It is important to reinforce that sensors are simply sending values that they sense to the processing device. They are either converting these values to digital signals or sending the digital signals directly to the software interface. Code that is written for input is usually written to compare values and trigger an event based on those values. Once students learn how to create the code to manipulate one sensor, they are well on their way to creating the code for any sensor.

It is common for teachers to refer to the universal systems model outlined in unit one for this outcome. These concepts represent the input side of the model.

Sample Performance Indicator

Create code that obtains and displays an input value from a sensor connected to an interface device.

Sensing and Control

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate the use of inputs with a physical interfacing device.

Connection

Students may

- Choose a sensor that interests them and write a piece of code to compare values from that sensor.

Consolidation

Students may

- Solve an authentic problem by writing code that triggers an event when it processes the data from an input device.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Explorer Hat Pro
  - Python programming tutorial

Teacher Resource Guide computerscience@nlesd.ca

- Unit 4

Senensing and Control

Outcomes

Students will be expected to

34.0 employ coding concepts to control outputs through a physical interfacing device [GCO 1]

Focus for Learning

Output devices are the components that perform the event that is the result of processing some data from the input devices in the previous outcome.

A physical interfacing output device can output data in a digital or analog format.

Digital Output: The interface device can change an electrical signal to on or off and, therefore, turn on motors or lights based on the decisions made in the code.

Analog Output: The interface device can take a series of stored numbers and change them into a varying electrical voltage. This voltage can be used to control an external device. The most common example would be digital music stored in a file being changed into a varying signal which is amplified and sent through a speaker system.

It is common for teachers to refer to the universal systems model outlined in unit one for this outcome. These concepts represent the output side of the model.

Actuators are output devices that are used to physically move and control an object.

Sample Performance Indicator

Add to the code created to manipulate the values delivered by an input device to create an event that engages an output device, such as a motor or an LED.

Sensing and Control

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate the controlling of various output devices.

Connection

Students may

- Use code to control an actuator through a physical interfacing device.

Consolidation

Students may

- Create code to solve a real world problem that controls an actuator through a physical interfacing device.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Halloween Explorer Hat Eyes
  - Explorer Hat Pro lights out
  - Explore Hat Pro Buggy

Teacher Resource Guide computerscience@nlesd.ca

- Unit 4

Sensing and Control

Outcomes

Students will be expected to

35.0 create a prototype that employs sensing and control events [GCO 1]

Focus for Learning

When students create a prototype that employs sensing and control events it should have a GUI, code, and hardware appropriate to the task. An example may be a fan that activates when a temperature reaches a certain threshold. The GUI may include a readout of the calculated temperature and manual start/stop or override buttons.

Students should be given choice in which sensors and actuators they experiment with. They should be encouraged to work with a variety of components.

This is intended to be a guided activity that provides students with an opportunity to become more familiar with the capacities of the sensors and actuators provided with the course. Familiarity with these components will facilitate more advanced innovation projects in unit five.

Sample Performance Indicator

Develop a solution to an authentic problem by writing a program to sense and control events external to the computer. This should include an appropriate GUI. This may be a program that combines or remixes code from previous activities.

Sensing and Control

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Demonstrate having an actuator respond to changing input from a sensor.
- Demonstrate examples of devices that incorporate sensing and control concepts such as a programmable thermostat or an ice maker in a refrigerator.

Consolidation

Students may

- Work cooperatively to experiment with code, input and output devices to solve an authentic problem.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 4
  - Explorer Hat Pro Pin Entry

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 4

Section Three:

Specific Curriculum Outcomes

Unit 5: Innovation Challenge

Focus

This unit represents the culminating activity of the course. The course is designed so that students spend the first four units gaining the computer science knowledge and skills necessary to complete an innovation challenge in unit five. Students will be challenged to think critically about the world around them to identify an authentic problem that can be solved with the available resources. Working cooperatively in design teams, students will create innovative solutions to identified problems and present prototypes and documentation in a class or school based innovation fair.

Outcomes Framework

GCO 1: Technological Problem Solving: Students will be expected to design, develop, evaluate and articulate technological solutions.

39.0 investigate authentic problem situations to determine opportunities to develop an innovative solution using sensing and control
40.0 generate a variety of solutions to a chosen problem and select the best solution
43.0 test and refine the prototype's design to ensure that it solves the identified problem
44.0 present the project portfolio and prototype in an in-class or school-based technology fair

GCO 2: Technological Systems: Students will be expected to operate and manage technological systems.

41.0 identify specific tools, consumable materials and interfacing devices needed to develop the solution
42.0 build a prototype using the identified hardware and software resources

GCO 3: History and Evolution of Technology: Students will be expected to demonstrate an understanding of the history and evolution of technology and of its social and cultural implications

36.0 identify examples of technological innovations in Atlantic Canada

GCO 4: Technology and Careers: Students will be expected to demonstrate an understanding of current and evolving careers and of the influence of technology on the nature of work

37.0 work cooperatively and collaboratively in design teams
38.0 maintain a design portfolio of documentation for the design activity

SCO Continuum

Unit 5 Innovation Challenge

Control Technology 8	Computer Science 1204
• work cooperatively and collaboratively in design teams • maintain a complete design portfolio of the design process and design activity • identify real life control technology problem situations and opportunities, and select one for further development • identify and clearly state control technology problems • identify specific tools/machines and resources that are required to effectively develop the solution • present the design portfolio, the design solution and the design activity report to the class	• work cooperatively and collaboratively in design teams • maintain a design portfolio of documentation for the design activity • generate a variety of solutions to a chosen problem and select the best solution • identify specific tools, consumable materials and interfacing devices needed to develop the solution • build a prototype using the identified hardware and software resources • test and refine the prototype's design to ensure that it solves the identified problem • present the project portfolio and prototype in an in-class or school-based technology fair

Suggested Unit Plan

The suggested time for unit five is 30 hours. The innovation challenge is the culminating activity of the course. It should be planned as one large project based learning (PBL) activity where students work cooperatively in design teams and teachers facilitate the process so that all design teams feel the success of developing an authentic, innovative solution to a real world problem. It is recommended that student design teams present their prototype and documentation at an event such as a class or school-based innovation fair.

Getting Organized

Outcomes

Students will be expected to

36.0 identify examples of technological innovations in Atlantic Canada [GCO 3]

Focus for Learning

Students should develop a context for the concept of innovation and how it applies to coding and the deployment of new technologies.

Students should have the opportunity to explore the processes startup companies have used to bring a new technological innovation from a simple design challenge to viable solution to an authentic real world problem. Students should also have the opportunity to explore how these solutions evolved into a viable business opportunity.

Teachers should take every opportunity to relate real success stories to the design challenge students are about begin. Where possible, teachers should provide examples of innovations initiated by young entrepreneurs originating in Newfoundland and Labrador and Atlantic Canada.

Sample Performance Indicator

White a profile of an Atlantic Canadian company that has taken a product from the prototyping stage to the marketplace.

Getting Organized

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Invite guest speakers into the classroom to discuss their process of moving a new innovation from concept to market.
- Lead a class discussion on some of the most successful inventions in history, highlighting milestones in their development process (e.g., velcro, ball point pen, telephone or light bulb).

Consolidation

Students may

- Select a technological innovation from an Atlantic Canadian company and create a timeline of its development.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Virtual Hike
  - Redspace
  - The Story of Verafin

Teacher Resource Guide computerscience@nlesd.ca

- Unit 5

Getting Organized

Outcomes

Students will be expected to

37.0 work cooperatively and collaboratively in design teams [GCO 4]

Focus for Learning

Students should complete the design project as a team effort. This project should be completed using a project based learning (PBL) approach. A PBL approach includes

- content that is significant and authentic to students' lives;
- the four Cs of learning: critical thinking, creativity, collaboration and communication;
- problems that require in-depth inquiry and research;
- intrinsically motivating activities that stimulate students' need to know;
- a student centred environment where students have voice and choice in the activities they select;
- opportunities for students to test their solutions and revise as necessary; and
- a culminating activity presented to a public audience.

In industry, product development is usually completed by teams of professionals who bring a variety of skills to the project. Effective collaboration is an essential employability skill. Design teams are most effective in groups of two or three.

Students should establish a design team structure, determining roles and developing an initial plan of action.

Team members should

- share responsibilities,
- share ideas,
- participate,
- assume leadership in the area of expertise/interest when called upon to do so,
- allow others to take the lead when necessary,
- compromise on some issues, and
- show respect for the opinions of other group members.

Getting Organized

Sample Teaching and Assessment Strategies

Activation

Teachers may

- review the characteristics of a good team member and principles of good group collaboration.
- Invite guest speakers to speak about how design teams work in real life situations in industry.

Connection

Students may

- Discuss their own skills and interest and identify which roles they can fill in the design team.

Consolidation

Students may

- Work as a team to create a thought-web of some important skills they believe should make up a successful design team.
- Work as a team to create a poster that illustrates graphically the practices that make up good group collaboration
- Work as a team to create an assessment tool to use to assess each other at the end of the project.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Five Basic Elements
  - What is Cooperative Learning?

Teacher Resource Guide computerscience@nlesd.ca

- Unit 5

Getting Organized

Outcomes

Students will be expected to

38.0 maintain a design portfolio of documentation for the design activity [GCO 4]

Focus for Learning

Each design team should document the design process in the form of a portfolio.

The design portfolio is an essential component of the design process. It will make up most of the content of the final team presentation at the end of the project. It should contain

- an introductory page,
- a design team page,
- a daily log page, and
- major headings providing space for the documentation from each of the parts of a design process.

Because of time restraints, teachers may wish to provide a design portfolio template. It is recommended that the portfolio be compiled as a website. This will provide flexibility for teachers and students to access content from any location. It also facilitates the upload of photos and other multimedia content.

While providing a design portfolio template is recommended, It is important that design teams have the opportunity to make their design portfolio reflective of them as a design team. Students need to feel ownership of the portfolio and the process in general.

Design portfolios are essential to the design process. They are like diaries and need to be routinely maintained to have meaning. They should track all ideas, decisions, actions and activities. Pages may contain but are not limited to

- photographs of members working on the various aspects of the project,
- sketches/documents related to the topic,
- short videos of prototype development and testing, or
- notes and questions related to research and group decisions.

Sample Performance Indicator

Create and maintain a design portfolio that includes all sections required to hold documentation collected from the innovation challenge.

Getting Organized

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Provide exemplars of completed design portfolios from past years.
- Provide examples of documentation from recognized inventions.

Connection

Students may

- Make aesthetic presentation of their design portfolio unique to their design team.
- Reflect on situations where documenting something in their personal lives led to successful final results.

Consolidation

Students may

- Use the design portfolio with full documentation as the basis of the presentation of their project at the end of the innovation challenge.

Resources and Notes

Suggested

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 5

Getting Started

Outcomes

Students will be expected to

39.0 investigate authentic problem situations to determine opportunities to develop an innovative solution using sensing and control [GCO 1]

Focus for Learning

The purpose of this stage of design is to investigate situations that may provide opportunities to design innovative solutions to authentic problems. In this design activity students should be challenged to be solution developers of real world problems.

There are three parts to this section of the process

- Identify opportunities from which to select a suitable project.
- Collect information and pose questions to clarify the problem to be solved.
- Write a design brief summarizing the problem and general approach to solving it.

Although it may not be realistic to require students to identify their own problem situations, it is not the intent of the innovation challenge to give students everything they need to develop a solution from a prescribed set of directions. Design teams should be given every opportunity to question, problem solve and troubleshoot as part of the design activity. Regardless of the method of identifying authentic problems, it is essential that students feel ownership of the unique problem they have chosen to solve.

Teachers should challenge students' ideas but be aware of the abilities of students so that they do not get frustrated with the process. All students should feel success in this process.

Students should have the opportunity to reflect on and think critically about the hardware and software they have been using throughout the course. They will need to consider their innovative ideas in the context of the resources they have and the skills they have acquired.

Getting Started

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Lead a class discussion about common problems that have been solved by innovative products.
- Guide an investigation into the world of product development.

Connection

Students may

- Reflect on product development processes of devices they use daily.

Consolidation

Students may

- Use the terminology associated with the process of solution development in a discussion of project ideas.
- Participate in a thorough problem analysis activity to establish an authentic problem that needs a solution.
- Create a list of capabilities of the technologies used in the first four units of the course.
- Create an inventory of the electronic components available to them when they begin planning solutions.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Making Project Based Learning Real World
  - Making Project Based Learning Authentic
  - Project Based Learning: A Real World Solution

Teacher Resource Guide computerscience@nlesd.ca

- Unit 5

Getting Started

Outcomes

Students will be expected to

40.0 generate a variety of solutions to a chosen problem and select the best solution [GCO 1]

Focus for Learning

Students should engage in brainstorming exercises. It may be useful to ask students to think about solutions in advance and share their ideas with other design team members.

Teachers should ensure that all students have an opportunity to express their ideas. All ideas should be given equal weight. Students should try for a minimum of 3 different ideas or variations of an idea.

Once ideas are generated and presented in the design team, students should use some sort of assessment strategy to select the best possible solution to the problem. Assessment of possible solutions can involve students creating scoring charts that include the list of characteristics a prototype should include and a corresponding grade. All team members can use this instrument to assess their possible solution.

Sample Performance Indicator

Document three alternative solutions in the design portfolio and record an assessment strategy to select the best possible solution.

Getting Started

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Lead a discussion about the systematic process of assessing and choosing the best solution.

Consolidation

Students may

- Create an assessment checklist to use as part of the assessment process of alternative solutions.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Making Project Based Learning Real World
  - Making Project Based Learning Authentic

Teacher Resource Guide computerscience@nlesd.ca

- Unit 5

Develop the Solution

Outcomes

Students will be expected to

41.0 identify specific tools, consumable materials and interfacing devices needed to develop the solution [GCO 2]

Focus for Learning

Students should develop the solution to the chosen problem including identifying tools, resources, and skills required.

This step is the most time consuming step of the design process.

Preparation should include

- identification and preparation of appropriate work spaces and tools for the design teams;
- the collection of resources, including consumable items, for the design activity;
- development of a strategy that ensures design work is shared equitably among the design team membership; and
- assurance that students understand and are following the safety guidelines.

At this point teachers may have to negotiate with design teams to ensure that what they are proposing is realistic in the context of available consumable materials and the hardware. It is essential that the teacher as the facilitator set up a situation where student projects are attainable within an environment where every student feels the success of being part of the creation of a solution to a real world problem.

Sample Performance Indicator

Compile a list of tools, interfacing devices and consumable materials required to complete the prototype you have chosen to develop.

Develop the Solution

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Review the materials and tools used throughout the course so that students know what they have at their disposal.

Connection

Students may

- Create a list in their design portfolio outlining the necessary resources to complete the project.
- Collaboratively decide on a tools and materials coordinator from members of the design team

Consolidation

Students may

- Create an estimated bill for materials to predict the cost of developing the prototype.

Resources and Notes

Supplementary Resources

- Raspberry Pi Innovation Kit
- Interfacing Components Kit

Develop the Solution

Outcomes

Students will be expected to

42.0 build a prototype using the identified hardware and software resources [GCO2]

43.0 test and refine the prototype's design to ensure that it solves the identified problem [GCO 1]

Focus for Learning

Each design team should finish the solution during this phase. Modeling and prototyping of the solution are required to complete the innovation challenge. Once students have completed initial tests on their prototype they will likely enter into a cycle of tuning and testing their solution.

This section of the design challenge will likely require the most significant amount of time. It is important to ensure that students are using some sort of systematic process to test and improve their prototype. It is critical that students document the changes being made so that the testing process results in a successful prototype.

During this part of the process, teachers will need to work closely with each design team; challenging them to think critically about what they are building.

Sample Performance Indicator

- Sense and control events external to the computer using a programming language physical interface equipment and a computing device.
- Document the results of prototype tests to facilitate the improvement of the technical functionality of your prototype.

Develop the Solution

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Cycle through each design team looking for opportunities to question and clarify what teams are creating.
- Lead a discussion with students about best practices for testing a prototype.

Connection

Students may

- Assign roles to team members to facilitate the building process.
- Create an assessment tool to assess the prototypes ability to solve the selected problem.

Consolidation

Students may

- Develop testing criteria for the testing of the prototype.
- Maintain a digital portfolio with results of prototype testing. Document strategies for prototype improvement based on testing data.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Project Based Learning: A Real World Solution

Teacher Resource Guide computerscience@nlesd.ca

- Unit 5

Develop the Solution

Outcomes

Students will be expected to

44.0 share the project portfolio and prototype in an in-class or school based technology fair [GCO 1]

Focus for Learning

Students should demonstrate their understanding and use of the design process in developing an innovative solution to an authentic problem. They should share their documentation using appropriate tools and strategies. Sharing the documentation and project solution is an integral part of any PBL activity.

All members of the design team should participate in sharing the documentation. When representing their documentation, students should use appropriate language and terminology.

It is important that teachers provide a variety of choices for students in how they represent their work. In sharing their project documentation students could use but not be limited to:

- photographs,
- video, music,
- documentation and other notes,
- daily journals and reflections,
- posters, drawings or other visual art,
- slide-shows, and
- websites

Sample Performance Indicator

Present a full report on your innovation including

- a summary of the design brief, a summary of how the design process enabled the design team to achieve the solution;
- documentation on successes and challenges encountered;
- a demonstration of the solution,
- an evaluation of the solution, including a log of any improvements made to the design based on the evaluation, and
- evidence of shared responsibility among the design team members usually illustrated through a daily log.

The presentation structure should be based on the structure of the team's design portfolio. The portfolio should contain evidence of all aspects of the design activity.

Develop the Solution

Sample Teaching and Assessment Strategies

Activation

Teachers may

- Lead a discussion about being respectful and supportive when classmates are presenting.
- Design a class based innovation fair and have the rest of the school view exhibits.

Consolidation

Students may

- Use their design portfolio as a multimedia aid to share their solution to the innovation challenge.
- Create displays that illustrate the process they used to work through their innovation challenge.

Extension

Students may

- Explore options to market their prototype as a commercially viable product.

Resources and Notes

Suggested

Resource Links: https://www.k12pl.nl.ca/curr/10-12/te/cs1204/links.html

- Unit 5
  - Top tips
  - Designers

Teacher Resource Guide
computerscience@nlesd.ca

- Unit 5

References

Conference Board of Canada. (2014). Employability Skills Profile 2000+. Retrieved July 5, 2017 from http://www.conferenceboard.ca/Libraries/EDUC_PUBLIC/esp2000.sflb

Foundation for the Atlantic Canadian Technology Education Curriculum. (2000). Technology Education. Newfoundland and Labrador Departments of Education.

International Society for Technology in Education (ISTE). 2016. ISTE Standards for Students. Retrieved July 8, 2017 from https://www.iste.org/standards/standards/for-students

UNESCO. (2004). The plurality of literacy and its implications for policies and programmes. Education Sector Position Paper. Paris, UNESCO.

University of Western Ontario. (2009). Understanding inclusive education. Retrieved April 15, 2015 from http://www.inclusiveeducationresearch.ca/about/inclusion.html

White, Ron. (2015). How Computers Work, Tenth Editin. Que Publishing

World Commission on Environment and Development. (1987). Our common future, from one earth to one world. Retrieved April 15, 2015 from http://www.un-documents.net/our-common-future.pdf

September 2019
ISBN: 978-1-55146-706-1