BIG IDEAS

The **design cycle** is an ongoing reflective process.	Personal design choices require self-exploration, collaboration, and evaluation and refinement of skills.	Tools and technologies can be adapted for specific purposes.

Learning Standards

Curricular Competencies	Content
Students are expected to be able to do the following: **Applied Design** ***Understanding context*** • Conduct **user-centred research** to understand design opportunities and barriers ***Defining*** • Establish a point of view for a chosen design opportunity • Identify potential users, intended impact, and possible unintended negative consequences • Make decisions about premises and **constraints** that define the design space ***Ideating*** • Identify gaps to explore a design space • Generate ideas and add to others' ideas to create possibilities, and prioritize them for prototyping • Critically analyze how competing social, ethical, and sustainability considerations impact designed solutions to meet global needs for preferred futures • Work with users throughout the design process	*Students are expected to know the following:* • design opportunities • design cycle • advanced **programming structures** • standardized source code **documentation** • **self-documenting** code • **collaboration tools** for programming • **advanced pair programming** • User **interface design** • **error handling** • **debugging** tools • management of **complexity** • uses of **pre-built data structures** • bug reports and feature requests from users • appropriate use of technology, including digital citizenship, etiquette, and literacy • **interpersonal skills** necessary to work effectively within the IT sector

Learning Standards (continued)

Curricular Competencies	Content
Prototyping • Identify and apply **sources of inspiration** and **information** • Choose an appropriate form, scale, and level of detail for prototyping, and plan procedures for prototyping multiple ideas • Analyze the design for the life cycle and evaluate its **impacts** • Construct prototypes, making changes to tools, materials, and procedures as needed • Record **iterations** of prototyping ***Testing*** • Identify feedback most needed and possible **sources of feedback** • Develop an **appropriate test** of the prototype • Collect feedback to critically evaluate design and make changes to product design or processes • Iterate the prototype or abandon the design idea ***Making*** • Identify appropriate tools, technologies, materials, processes, and time needed for production • Use **project management processes** when working individually or collaboratively to coordinate production ***Sharing*** • **Share** progress while creating to increase feedback, collaboration, and, if applicable, marketing • Decide on how and with whom to share or promote their **product**, creativity, and, if applicable, **intellectual property** • Consider how others might build upon the design concept • Critically reflect on their design thinking and processes, and identify new design goals • Assess ability to work effectively both as individuals and collaboratively while implementing project management processes	

Learning Standards (continued)

Curricular Competencies	Content
Applied Skills • Apply safety procedures for themselves, co-workers, and users in both physical and digital environments • Identify and assess skills needed for design interests, and develop specific plans to learn or refine them over time **Applied Technologies** • Explore existing, new, and emerging tools, **technologies**, and systems to evaluate their suitability for their design interests • Evaluate impacts, including unintended negative consequences, of choices made about technology use • Analyze the role technologies play in societal change • Examine how cultural beliefs, values, and ethical positions affect the development and use of technologies	

Big Ideas – Elaborations	APPLIED DESIGN, SKILLS, AND TECHNOLOGIES – Computer Programming Grade 12

- **design cycle:** includes updating content, tools, and delivery. The design process can be non-linear.

Curricular Competencies – Elaborations	APPLIED DESIGN, SKILLS, AND TECHNOLOGIES – Computer Programming Grade 12

- **user-centred research:** research done directly with potential users to understand how they do things and why, their physical and emotional needs, how they think about the world, and what is meaningful to them
- **constraints:** limiting factors, such as available technology, expense, environmental impact, copyright
- **sources of inspiration:** may include experiences, users, experts, and thought leaders
- **information:** may include professionals as experts, secondary sources, collective pools of knowledge in communities and collaborative atmospheres both online and offline
- **impacts:** including the social and environmental impacts of extraction and transportation of raw materials, manufacturing, packaging, transportation to markets, servicing or providing replacement parts, expected usable lifetime, and reuse or recycling of component materials
- **iterations:** repetitions of a process with the aim of approaching a desired result
- **sources of feedback:** may include peers; users; First Nations, Métis, or Inuit community experts; other experts and professionals both online and offline
- **appropriate test:** includes evaluating the degree of authenticity required for the setting of the test, deciding on an appropriate type and number of trials, and collecting and compiling data
- **project management processes:** setting goals, planning, organizing, constructing, monitoring, and leading during execution
- **Share:** may include showing to others, use by others, giving away, or marketing and selling
- **product:** for example, a physical product, a process, a system, a service, or a designed environment
- **intellectual property:** creations of the intellect such as works of art, invention, discoveries, design ideas to which one has the legal rights of ownership
- **technologies:** tools that extend human capabilities

- **programming structures:** higher-level structures, such as functions, methods, or classes, that help improve the organization of source code
- **documentation:** documenting source code with industry standard tools
- **self-documenting:** writing source code in such a way that makes inline comments seem unnecessary
- **collaboration tools:** for example, online tools to facilitate pair and collaborative programming
- **advanced pair programming:** While reviewing, the observer considers the "strategic" direction of the work, coming up with ideas for improvements and likely future problems to address. The driver focuses their attention on the "tactical" aspects of completing the current task, using the observer as a safety net and guide.
- **interface design:** focus on maximizing usability and the user experience. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals
- **error handling:** the response and recovery procedures from error conditions present in a software application; the process comprised of anticipation, detection and resolution of application errors, programming errors or communication errors
- **debugging:** use of a debugger that is capable of stepping through code and monitoring variables
- **complexity:** for example, a project whose scale requires multiple source files or functions
- **pre-built data structures:** the data structures that are provided (e.g., from a standard library)
- **interpersonal skills:** for example, people skills, social skills, communication, attitudes, collaboration, follow-ups, courtesies, record keeping