ICS3C

* This course introduces students to computer programming concepts and practices.
* Students will write and test computer programs, using various problem-solving strategies.
* They will learn the fundamentals of program design and apply a software development life-cycle model to a software development project.
* Students will also learn about
  + computer environments and systems,
  + and explore environmental issues related to computers,
  + safe computing practices, emerging technologies,
  + and postsecondary opportunities in computer-related fields.

ICS4C

* This course further develops students’ computer programming skills.
* Student teams will plan and carry out a software development project using industry-standard programming tools and proper project management techniques.
* Students will learn object-oriented programming concepts and create object-oriented software solutions.
* Students will design graphical user interfaces.
* Students will also investigate
  + ethical issues in computing and
  + expand their understanding of environmental issues,
  + expand their understanding of emerging technologies,
  + expand their understanding of computer-related careers.

A. Programming Concepts and Skills (ICS3C)

A1. demonstrate the ability to use different data types in expressions in simple computer programs;

A1.1 use constants and variables,

A1.1 use integers, floating points,

A1.1 use strings,

A1.1 use Boolean values,

A1.2 demonstrate the ability to manipulate string data in a computer program

A1.3 use assignment statements correctly with arithmetic expressions in computer programs

A1.3 use assignment statements correctly with string expressions in computer programs

A1.4 use Boolean operators *(e.g., AND, OR, NOT)*,

A1.4 use comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to),

A1.4 use arithmetic operators *(e.g., addition, subtraction, multiplication, division, xponentiation, parentheses)*, and order of operations correctly.

A2. demonstrate the ability to use control structures and simple algorithms in computer programs;

A2.1 write programs that incorporate user input,

A2.1 write programs that incorporate processing,

A2.1 write programs that incorporate screen output;

A2.2 use sequence control structures to create programming solutions;

A2.2 use selection control structures to create programming solutions;

A2.2 use repetition control structures to create programming solutions;

A2.3 demonstrate the ability to write algorithms with nested structures.

A3. use proper code maintenance techniques and conventions when creating computer programs.

A3.1 explain the difference between syntax, logic, and run-time errors in computer programs;

A3.2 demonstrate the ability to correct syntax, logic, and run-time errors in computer programs;

A3.3 use workplace and professional conventions *(e.g., naming, indenting, commenting)* correctly to write programs and internal documentation;

A3.4 demonstrate the ability to interpret error messages displayed by programming tools *(e.g., compiler, debugging tool),* at different times during the software development process *(e.g., writing, compilation, testing)*;

A3.5 demonstrate the ability to validate a program using test cases.

A. Programming Concepts and Skills (ICS4C)

A1. use data structures in the design and creation of computer programs;

A1.1 perform operations on data types typically used in business applications (e.g., express

money amounts as dollars and cents, express dates and times in various formats);

A1.2 use Boolean operators (e.g., AND, OR, NOT),

A1.2 use comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to),

A1.2 use arithmetic operators (e.g., addition, subtraction, multiplication, division, exponentiation, parentheses), and order of operations correctly in programming;

A1.3 describe the structure of one-dimensional and two-dimensional arrays and related concepts including elements, indexes, and bounds.

A2. demonstrate the ability to use standard algorithms in the design and creation of computer programs;

A2.1 demonstrate the ability to manipulate and convert data in a computer program (e.g., parse strings; convert data types such as numeric to string, and string to numeric; convert ‘yes’ or ‘no’ to Boolean);

A2.2 demonstrate the ability to read from, and write to, an external file (e.g., sequential file, database, XML file, relational database via SQL);

A2.3 demonstrate the ability to declare, initialize, modify, and access one-dimensional and two-dimensional arrays and elements within a program;

A2.4 demonstrate the ability to add, change, or delete elements of an array of objects in a program;

A2.5 demonstrate the ability to use search (e.g., string.indexOf (“cool”)) in a program.

A2.5 demonstrate the ability to use sort routines (e.g., Arrays.sort(intArray)) in a program.

A3. demonstrate an understanding of object-oriented programming concepts and practices in the design

and creation of computer programs;

A3.1 explain the importance of designing reusable code in computer programs;

A3.2 explain fundamental object-oriented programming concepts (e.g., classes, objects, methods);

A3.3 apply the concepts of scope and visibility for variables, constants, and methods when creating classes in computer programs;

A3.4 compare and contrast object-oriented and procedural programming paradigms.

A4. create clear and accurate internal and external documentation to ensure the maintainability of

computer software.

A4.1 write maintainable computer programs by creating clear and accurate internal documentation that provides in-depth explanations of complex blocks of code;

A4.2 create clear and maintainable external user documentation (e.g., Help file, how-to manual, FAQ, installation procedures, backup and recovery procedures, training materials) as part of a complete software development project;

A4.3 develop and implement a formal testing plan for a software development project to ensure

program correctness.

B. Software Development (ICS3C)

B1. use a variety of problem-solving strategies to solve different types of problems;

B1.1 use various problem-solving strategies (e.g., divide and conquer, working backwards, process analysis, examples, extreme cases, tables and charts, trial and error) to solve programming problems;

B1.2 use the input-process-output model to solve programming problems.

B2. design software solutions to meet a variety of challenges, using a set of standards;

B2.1 design a simple program from a program template or skeleton (e.g., teacher-supplied skeleton, Help facility code snippet);

B2.2 use appropriate vocabulary and mode of expression (i.e., written, oral, diagrammatic) to describe alternative program designs

B2.2 use appropriate vocabulary and mode of expression (i.e., written, oral, diagrammatic) to explain the structure of a program;

B2.3 write subprograms (e.g., functions, procedures) that perform one well-defined task

B2.3 write subprograms (e.g., functions, procedures) that use parameter passing

B2.3 write subprograms (e.g., functions, procedures) that us appropriate variable scope (e.g., local, global);

B2.4 use industry-standard programming tools (e.g., structure chart, flow chart, UML [Unified Modeling Language], data flow diagram, pseudocode) to represent the structure and components of a computer program;

B2.5 design user-friendly software interfaces (e.g., prompts, messages, screens, forms).

B3. design simple algorithms according to specifications;

B3.1 use simple algorithms (e.g., validate entered data, count, accumulate, use random numbers) to design a program according to specifications;

B3.2 solve problems (e.g., calculation of gross pay; fuel consumption on a car trip; average of students’ marks; temperature at a given altitude, using the environmental lapse rate) by applying mathematical equations or formulas in an algorithm;

B3.3 design algorithms to detect, intercept, and handle run-time errors (e.g., division by zero,

roots of negatives).

B4. apply a software development life-cycle model to a software development project.

B4.1 describe the phases (i.e., problem definition, analysis, design, writing code, testing, implementation, maintenance), milestones (e.g., date of completion of program specification), and products (e.g., specification, flow chart, program, documentation, bug reports) of a software development life cycle;

B4.2 use a variety of techniques (e.g., dialogue, questionnaires, surveys, research) to clarify program specifications;

B4.3 use project management tools (e.g., Gantt chart, critical path diagram, PERT chart) to show

tasks and milestones in a teacher-led project;

B4.4 use a test plan to test programs (i.e., identify test scenarios, identify suitable input data,

calculate expected outcomes, record actual outcomes, and conclude ’pass‘ or ’fail‘) by comparing expected to actual outcomes;

B4.5 use a variety of methods to debug programs (e.g., manual code tracing, extra code to output the state of variables);

B4.6 communicate information about the status of a project (e.g., milestones, work completed,

work outstanding) effectively in writing throughout the project.

B. Software Development (ICS4C)

B1. design standard algorithms according to specifications;

B1.1 design algorithms to solve practical mathematical problems (e.g., amount of paint or carpet needed, number of shingles needed, energy costs, amount of water needed for an aquarium, projection of Aboriginal youth population growth);

B1.2 design algorithms that require precision and accuracy when rounding numbers (e.g., currency calculations, amortization, recipe volume changes);

B1.3 design data validation routines (e.g., capitalization; formatting of postal codes, telephone numbers, SINs, and alphanumeric data; length and range checking).

B2. design software solutions using object-oriented programming concepts;

B2.1 demonstrate the ability to create and use instance methods (e.g., constructors, mutators, accessors) in a computer program;

B2.2 design a simple base class to represent objects or concepts in a problem statement, using program templates or skeletons;

B2.3 write methods that require parameter passing in a computer program.

B3. design user-friendly graphical user interfaces (GUIs) that meet user requirements;

B3.1 design graphical user interfaces that contain common controls (e.g., buttons, labels, text boxes);

B3.2 design a user-friendly graphical user interface that helps to improve user accessibility (e.g., for multilingualism; for those with limited eyesight or colour vision deficiency);

B3.3 evaluate a user interface for conformity with a given accessibility standard (e.g., Section 508 of the Rehabilitation Act (US), W3C User Interface Domain, or a student- or teacher-created standard);

B3.4 design responses to user events in a graphical user interface.

B4. participate in a large student-managed project, using proper project management tools and

techniques to manage the process effectively.

B4.1 describe the phases of a model (e.g., waterfall, iterative, XP [Extreme Programming], RAD [Rapid Application Development]) of the software development life cycle;

B4.2 create a project plan for a software development project, outlining the tasks at each phase

of the software development life cycle;

B4.3 use project management tools (e.g., Gantt chart, PERT chart) and time management tools (e.g., organizer, calendar) to help develop a software project;

B4.4 use industry-standard programming tools (e.g., UML [Unified Modeling Language], diagrams,

structure charts, flow charts, pseudocode) to develop a software project.

C. Computer Environments and Systems (ICS3C)

C1. demonstrate an understanding of the functions of different types of computer components;

C1.1 describe the functions and features of the internal components of a computer (e.g., CPU,

RAM, ROM, cache, hard drive, motherboard, power supply, video card, sound card);

C1.2 use correct terminology to describe computer features and specifications (e.g., processor type, bus speed, storage capacity, amount of memory);

C1.3 describe the functions and features of common computer peripheral devices (e.g., printer, monitor, scanner, keyboard, mouse, speakers, USB flash drive);

C1.4 compare and contrast common ISP services (e.g., DSL, cable, dial-up, regional Wi-Fi) and

home networking hardware (e.g., NICs, routers, hardware used for wired and wireless connections).

C2. use appropriate file maintenance practices to organize and safeguard data;

C2.1 use an operating system to logically organize computer files for easy retrieval, backup,

and recovery;

C2.2 use standard backup procedures to back up user files.

C3. use a software development environment to write and run computer programs.

C3.1 describe the functions and features of a software development environment and use

it to write and run a computer program;

C3.2 describe the differences between applications, programming languages, and operating systems;

C3.3 use Help documentation as a guide to designing and writing programs.

C. Computer Environments and Systems (ICS4C)

C1. demonstrate the ability to use project management tools to plan and track activities for a software

development project;

C1.1 use software tools (e.g., email, wikis, blogs, task lists, bulletin boards, spreadsheets, shared calendars) to plan and track activities during a software development project;

C1.2 communicate information about project status (e.g., completed, in progress, not started, problems encountered, recommended modification to deadlines) effectively in writing throughout the project.

C2. demonstrate the ability to use software development tools to design and write a computer program.

C2.1 use the features of a software development environment to debug programs and create functioning computer programs;

C2.2 work independently, using the Help function, to resolve syntax issues while programming;

C2.3 work independently, using reference materials (e.g., code snippets, sample programs, APIs, tutorials), to design and write functioning computer programs.

D. Computers and Society (ICS3C)

D1. describe computer use policies that promote environmental stewardship and sustainability;

D1.1 describe negative effects of computer use on the environment (e.g., creation of waste, unnecessary printing of emails, heavy power consumption)

D1.1 describe negative effects of computer use on human health (e.g., exposure to radiation, musculoskeletal disorders, eye strain, various health consequences of reduced activity levels);

D1.2 identify measures that help reduce the impact of computers on the environment (e.g., lab regulations, school policies, corporate policies, provincial policies, paperless workplaces, computer recycling and reuse)

D1.2 identify measures that help reduce the impact of computers on human health (e.g., ergonomic standards);

D1.3 describe ways in which computers are or could be used to reduce resource use and to support environmental protection measures (e.g., computer modelling to reduce use of physical resources; interpretation of large amounts of environmental data; management of natural resources; programmable temperature control to reduce energy consumption);

D1.4 identify government agencies and community partners that provide environmental stewardship opportunities (e.g., local community recycling centres, private companies that refurbish computers, printer cartridge recycling programs).

D2. describe and apply procedures for safe computing to safeguard computer users and their data;

D2.1 explain the need for an acceptable-use policy for using computers at school and at work;

D2.2 describe and use appropriate strategies to avoid potential health and safety problems associated with computer use (e.g., musculoskeletal disorders, eye strain);

D2.3 describe procedures to safeguard data and programs from malware (e.g., viruses, spyware, adware).

D3. explain key aspects of the impact that emerging technologies have on society;

D3.1 explain how emerging technologies can affect personal rights and privacy (e.g. video surveillance, cyberbullying, identity theft);

D3.2 describe some emerging technologies and their implications for, and potential uses by, various members of society;

D3.3 describe some of the solutions to complex problems affecting society that have been or are

being developed through the use of advanced computer programming and emerging technologies (e.g., monitoring and regulating electrical supply and demand; using facial recognition programs to verify the identity of persons entering a country; analysing criminal activity by overlaying crime data on satellite imagery; analyzing large-scale meteorological data to predict catastrophic storms).

D4. describe postsecondary education and career prospects related to computer studies.

D4.1 research and describe trends in careers that require computer skills, using local and

national sources (e.g., local newspaper, national newspaper, career websites);

D4.2 identify opportunities for experiential learning (e.g., co-op programs, job shadowing, career fairs) related to computer science;

D4.3 research and report on postsecondary educational programs leading to careers in the field

of information systems and computer science (e.g., institutions offering relevant programs,

industry certifications, courses of study, entrance requirements, length of programs, costs);

D4.4 identify groups and programs that are available to support students who are interested

in pursuing non-traditional career choices in computer-related fields (e.g., mentoring

programs, virtual networking/support groups, specialized postsecondary programs, relevant

trade/industry associations);

D4.5 describe the Essential Skills and work habits that are important for success in computer

studies, as identified in the Ontario Skills Passport.

D. Computers and Society (ICS4C)

D1. analyse and apply strategies that promote environmental stewardship with respect to the use

of computers and related technologies;

D1.1 outline and apply strategies to reduce the impact of computers and related technologies on the environment (e.g., reduce, reuse, and recycle; turn computers and monitors off at end of day; participate in printer cartridge recycling) and on human health (e.g., ergonomic standards);

D1.2 investigate and describe governmental and community initiatives promoting environmental stewardship and sustainability (e.g., local community recycling centres, private companies that refurbish computers, printer cartridge recycling programs).

D2. demonstrate an understanding of ethical issues and practices related to the use of computers;

D2.1 investigate and describe an ethical issue related to the use of computers (e.g., piracy, privacy, security, phishing, spyware, cyberbullying);

D2.2 describe the essential elements of a code of ethics for computer programmers, and explain why there is a need for such a code (e.g., plagiarism, backdoors, spyware, unethical programming practices);

D2.3 outline and apply strategies to encourage ethical computing practices at home, at school, and at work.

D3. investigate and report on emerging computer technologies and their potential impact on society

and the economy;

D3.1 describe the evolution of some emerging programming languages;

D3.2 investigate and report on innovations in information technology (e.g., webcasting, VoIP, multiplayer online gaming) and their potential impact on society and the economy;

D3.3 describe programming requirements for a variety of emerging technologies (e.g., web programming, smartphones, embedded systems).

D4. research and report on the range of career paths and lifelong learning opportunities in software

development or a computer-related field.

D4.1 research and report on the range of career opportunities in software development, including

duties, responsibilities, qualifications, and compensation;

D4.2 research and report on opportunities for lifelong learning in software development or a computer-related field; D4.3 evaluate their own development of Essential

Skills and work habits that are important for success in computer studies, as identified in the Ontario Skills Passport.