TFS High School

5635 Yong St. Suite 206,
Toronto, Ontario M2M 3S9

COURSE OUTLINE

Introduction to Computer Science University Preparation

ICS4U (University)

Department	Computer Studies
Instructor	Maliheh Mohseni
Course Development Date	Jan 2019
Ministry Course Code	ICS4U
Credit Value	1.00
Ministry Curriculum Document	Policy Document: Computer Studies, The Ontario Curriculum Grades 10 to 12, Revised 2008 http://www.edu.gov.on.ca/eng/curriculum/secondary/computer10to12_2008.pdf http://www.edu.gov.on.ca/eng/policyfunding/growSuccess.pdf
Prerequisites	ICS3U Introduction to Computer Science, Grade 11, University Preparation
Course Revision Date (TFS)	August 2023

COURSE DESCRIPTION

This course enables students to further develop knowledge and skills in computer science.
Students will use modular design principles to create complex and fully documented programs, according to industry standards.
Student teams will manage a large software development project, from planning through to project review.
Students will also analyse algorithms for effectiveness. They will investigate ethical issues in computing and further explore environmental issues, emerging technologies, areas of research in computer science, and careers in the field.

OVERALL EXPECTATIONS

Course Standards, Overall expectation	***Concepts***
A. Programming Concepts & Skills	A1. Demonstrate the ability to use different data types and expressions when creating computer programs; A2. describe and use modular programming concepts and principles in the creation of computer programs; A3. Design and write algorithms and subprograms to solve a variety of problems; A4. Use proper code maintenance techniques when creating computer programs.
B. Software Development	B1. Demonstrate the ability to manage the software development process effectively, through all of its stages – planning, development, production, and closing; B2. Apply standard project management techniques in the context of a student-managed team project.
C. Designing Modular Programs	C1. Demonstrate the ability to apply modular design concepts in computer programs; C2. Analyse algorithms for their effectiveness in solving a problem.
D. Topics in Computer Science	D1. Assess strategies and initiatives that promote environmental stewardship with respect to the use of computers and related technologies; D2. Analyse ethical issues and propose strategies to encourage ethical practices related to the use of computers; D3. Analyse the impact of emerging computer technologies on society and the economy; D4. Research and report on different areas of research in computer science, and careers related to computer science.

Specific Curriculum Expectations

A. Programming Concepts and Skills http://edu.gov.on.ca/eng/curriculum/secondary/computer10to12_2008.pdf	
A1. **Data Types and Expressions**	**A 3. Designing Algorithms**

By the end of this course, students will:
A1.1 demonstrate the ability to use integer division and resultant remainders in computer programs;
A1.2 demonstrate an understanding of type conversion *(e.g., string-to-integer, character-to integer, integer-to-character, floating point-to integer, casting in an inheritance hierarchy)*;
A1.3 demonstrate the ability to use non-numeric comparisons *(e.g., strings, comparable interface)* in computer programs;
A1.4 demonstrate an understanding of the limitations of finite data representations *(e.g., integer bounds, precision of floating-point real numbers, rounding errors)* when designing algorithms;
A1.5 describe and use one-dimensional arrays of compound data types *(e.g., objects, structures, records)* in a computer program.

A 2. Modular Programming

By the end of this course, students will:
A2.1 create a modular program that is divided among multiple files *(e.g., user-defined classes, libraries, modules)*;
A2.2 use modular design concepts that support reusable code *(e.g., encapsulation, inheritance, method overloading, method overriding, polymorphism)*;
A2.3 demonstrate the ability to modify existing modular program code to enhance the functionality of a program.

By the end of this course, students will:
A3.1 demonstrate the ability to read from, and write to, an external file *(e.g., text file, binary file, database, XML file)* from within a computer program;
A3.2 create linear and binary search algorithms to find data in an array;
A3.3 create subprograms to insert and delete array elements;
A3.4 create a sort algorithm *(e.g., bubble, insertion, selection)* to sort data in an array;
A3.5 create algorithms to process elements in two dimensional arrays *(e.g., multiply each element by a constant, interchange elements, multiply matrices, process pixels in an image)*;
A3.6 design a simple and efficient recursive algorithm *(e.g., calculate a factorial, translate numbers into words, perform a merge sort, generate fractals, perform XML parsing).*

A4. Code Maintenance

By the end of this course, students will:
A4.1 work independently, using support documentation *(e.g., IDE Help, tutorials, websites, user manuals)*, to resolve syntax issues during software development;
A4.2 develop and implement a formal testing plan *(e.g., unit testing, integration testing, regression testing)* for a software project to ensure program correctness;
A4.3 create fully documented program code according to industry standards *(e.g., doc comments, docstrings, block comments, line comments)*;
A4.4 create clear and maintainable external user documentation *(e.g., Help files, training materials, user manuals).*

B. Software Development

http://edu.gov.on.ca/eng/curriculum/secondary/computer10to12_2008.pdf

B1. Project Management

By the end of this course, students will:
B1.1 create a software project plan by producing a software scope document and determining the

B1. Project Management

By the end of this course, students will:
B1.1 create a software project plan by producing a software scope document and determining the tasks, deliverables, and schedule;
B1.2 develop the software product according to the project plan (i.e., ensure that the software meets end user needs, functions as intended, and can be produced within quality standards, budget, and timelines);
B1.3 produce the software according to specifications (i.e., code, test, deploy), and create user documentation and training materials;
B1.4 use an appropriate project management tool *(e.g., Gantt chart, PERT chart, calendar)* to manage project components;

project components;
B1.5 close the project (i.e., confirm that software meets all user requirements, deliver software in appropriate format, plan software support
B1.6 review the management of the project *(e.g., compare plan to actual performance, outline successes, make recommendations for improvement)* and prepare a report in an appropriate format;
B1.7 demonstrate the ability to use shared resources to manage source code effectively and securely *(e.g., organize software components using shared files and folders with timestamps, and proper version control).*

B2. Software Project Contribution

By the end of this course, students will:
B2.1 demonstrate the ability to contribute, as a team member, to the planning, development, and production of a large software project;
B2.2 demonstrate the ability to meet project goals

B1.5 close the project (i.e., confirm that software meets all user requirements, deliver software in appropriate format, plan software support and maintenance);

and deadlines by managing individual time during a group project;
B2.3 reflect on, and assess, team and individual progress during the project review.

C. Designing Modular Programs

http://edu.gov.on.ca/eng/curriculum/secondary/computer10to12_2008.pdf

C1. Modular Design

By the end of this course, students will:
C1.1 decompose a problem into modules, classes, or abstract data types *(e.g., stack, queue, dictionary)* using an object-oriented design methodology *(e.g., CRC [Class Responsibility Collaborator] or UML [Unified Modeling Language])*;
C1.2 demonstrate the ability to apply data encapsulation in program design *(e.g., classes, records, structures)*;
C1.3 demonstrate the ability to apply the process of functional decomposition in subprogram design;
C1.4 apply the principle of reusability in program design *(e.g., in modules, subprograms, classes, methods, and inheritance)*.D2.2 describe the essential elements of a code of
ethics for computer programmers *(e.g., ACM[Association for Computing Machinery] and IEEE [Institute of Electrical and Electronics Engineers] standards)* and explain why there is a need for such a code *(e.g., plagiarism, backdoors, viruses, spyware, logic bombs)*;

C2. Algorithm Analysis

By the end of this course, students will:
C2.1 demonstrate the ability to analyse a precondition (i.e., starting state) and a post condition (i.e., ending state) in an algorithm;
C2.2 compare the efficiency of linear and binary searches, using run times and computational complexity analysis *(e.g., to analyse the number of statements executed, the number of iterations of a loop, or the number of comparisons performed)*;
C2.3 compare the efficiency of sorting algorithms, using run times and computational complexity analysis *(e.g., to analyse the number of statements executed, the number of iterations of a loop, or the number of comparisons performed)*;
C2.4 identify common pitfalls in recursive functions *(e.g., infinite recursion, exponential growth in Recursive algorithms such as Fibonacci numbers).*

D . Environmental Stewardship and Sustainability

http://edu.gov.on.ca/eng/curriculum/secondary/computer10to12_2008.pdf

By the end of this course, students will:
D1.1 outline 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 report on governmental and community initiatives that encourage environmental stewardship and promote programs and practices that support sustainability *(e.g., local community recycling centres, private companies that refurbish computers, printer cartridge recycling programs).*

D 2. Ethical Practices
By the end of this course, students will:
D2.1 investigate and analyse an ethical issue related to the use of computers *(e.g., sharing passwords, music and video file downloading, software piracy, keystroke logging, phishing, cyberbullying)*;
D2.2 describe the essential elements of a code of ethics for computer programmers *(e.g., ACM*

D 3. Emerging Technologies and Society
By the end of this course, students will:
D3.1 explain the impact of a variety of emerging technologies on various members of society and on societies and cultures around the world and on the economy;
D3.2 investigate an emerging technology and produce a report using an appropriate format *(e.g., technical report, website, presentation software, video).*

D 4. Exploring Computer Science
By the end of this course, students will:
D4.1 report on some areas of collaborative research between computer science and other fields *(e.g., bioinformatics, geology, economics, linguistics, health informatics, climatology, sociology, art)*, on the basis of information found in industry publications *(e.g., from the ACM and IEEE)*;
D4.2 investigate a topic in theoretical computer science *(e.g., cryptography, graph theory, logic, computability theory, attribute grammar, automata*

[Association for Computing Machinery] and IEEE [Institute of Electrical and Electronics Engineers] standards) and explain why there is a need for such a code *(e.g., plagiarism, backdoors, viruses, spyware, logic bombs)*; D2.3 outline and apply strategies to encourage ethical computing practices at home, at school, and at work.	*theory, data mining, artificial intelligence, robotics, computer vision, image processing)*, and produce a report, using an appropriate format *(e.g., website, presentation software, video)*; D4.3 research and describe careers associated with computer studies *(e.g., computer scientist, software engineer, systems analyst)*, and the postsecondary education required to prepare for them; D4.4 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.

Outline of Course Content

Unit Title and Description	Time
Introduction to the NetBeans Integrated Development Environment and Java Review In this unit students will set up the NetBeans Integrated Development Environment, review basic programming theories and Java programming concepts covered in ICS3U. Students will also study environmental stewardship and ethical issues related to computers and technology.	14 hours
Graphical User Interface In this unit students will explore and implement topics such as a GUI (Graphical User Interface) using Java and NetBeans IDE. Students will also study and practice the second stage of Software Development: System Design.	15 hours
***Arrays(One-Dimensional and Two-dimensional)** In this unit students will study and implement programming concepts pertaining to one-dimensional arrays. Students will also compare the different careers and fields related to Computer Science. students will expand their knowledge of one-dimensional arrays to Two-dimensional arrays.	22 hours

Sorting In this unit students will study and implement sorting algorithms such as Bubble Sort and Selection Sort and Insertion Sort and Analyse the number of statements executed, the number of iterations of a loop, or the number of comparisons performed	12 hours
Object-Oriented Programming In this unit students will study and implement programming theories related to Object Oriented Programming. Students will also begin to investigate concepts of Classes and Objects, basic concepts of Object Oriented Programming which revolve around the real life entities.	20 hours
Advanced Topics In this unit students will study and implement advanced topics such as Recursion, Binary Search. Students will also study and practice the last stage of Software Development: Deployment and Maintenance.	15 hours
Project This course includes a final project worth 15% of the final grade. Working independently, students will create thrir project as object –oriented programming by following the Software Development Life-Cycle in Java-NetBEans.	6 hours
Final Evaluation This Assessment includes project presentation (using power point, worth 15%) and written exam worth 15%, Total 30% of final grade.	6 hours
Total	110 Hours

Teaching and Learning Strategies:

The aim of this course is to advance students understanding of computer programming concepts in order to create more complex computer programs that meet industry standards.
In order to achieve this goal, a wide variety of instructional strategies are used to provide learning opportunities to accommodate a variety of learning styles, interests, and ability levels. The following are used throughout the course as strategies for teaching and learning the concepts presented:

- Communicating: Through the use of discussions, this course offers students the opportunity to share their understanding both in oral as well as written form. They also have the opportunity to broaden their understanding by discussing concepts with other students and their teacher.
- Problem Solving: This course scaffolds learning by reviewing basic knowledge attained earlier in the course and in prior computer science courses and building off of this knowledge as the student progresses through the course. The course guides students toward recognizing opportunities to apply knowledge they have gained to solve problems.
- Connecting: This course connects the concepts taught to real-world applications (e.g. students will write programs that can read and write files, a very useful skill in many real-world applications).
- Representing: Through the use of examples, practice problems, and sample code, the course models various coding practices, poses questions that require students to use different representations as they are working at each level of conceptual development - concrete, visual or symbolic, and allows individual students the time they need to solidify their understanding at each conceptual stage.
- Guided Exploration: The course and teacher guide students through the exploration of a variety of coding practices and procedures necessary to be successful in computer science.

In addition, teacher and students have at their disposal a number of tools that are unique to electronic learning environments:

- Video presentations
- Discussion boards and google classroom
- Assessments with real-time feedback
- Interactive activities that engage both the student and teacher in the subject
- Peer review and assessment
- Internet Instructional Videos

The aim of this course is to prepare students with advanced computer programming. Students are taught a lesson in zoom platform and then practice independently to apply what they have learned in different contexts as problem solving process.

ASSESSMENT and EVALUATION

Evaluations will consist of tests & quizzes, assignments, projects, group work, and presentations.

To promote student success, ongoing formative assessment and feedback will be given to students. The course expectations will be evaluated according to the four categories of the achievement chart.

The evaluation for this course is based on the student's achievement of curriculum expectations and the demonstrated skills required for effective learning. The final percentage grade represents the quality of the student's overall achievement of the expectations for the course and reflects the corresponding level of achievement as described in the achievement chart for the discipline. A credit is granted and recorded for this course if the student's grade is 50% or higher.

Diagnostic assessment is used at the beginning of a unit to assist in determining a starting point for instruction. Assessment for Learning (AFL) provides information to students as they are learning and refining their skills. Assessment as Learning (AAL) acts as a stepping-stone for students to begin applying their understanding using critical thinking; it bridges the gap between AFL and AOL. Assessment of Learning (AOL), at the end of units and course, provides students with the opportunity to synthesize/apply/demonstrate their learning and the achievement of the expectations.

Evaluation is based on gathering evidence of student achievement through:

- Products
- Observations
- Conversations
- Conversations

The following is a list of specific assessment/evaluation strategies that the teacher may use but is not limited to:

Evaluation & Additional Information Evaluation in this course will be diagnostic, formative and summative.			
Summative Evaluation	70%	Knowledge and Understanding	25%
		Thinking/Inquiry & Problem-solving	25%
		Application	25%
		Communication	25%
Final Evaluation	30%	project	15%
		Examination	15%

Rating Scale	
Excellent	80 - 100 %
Good	66 - 79 %
Satisfactory	50 - 65 %
Needs Improvement	Below 50%

Teaching and Learning Strategies:

Communication	• Report/Presentation	• Collaborative/Cooperative
Problem Solving	• **Homework**	• Software Life Cycle Design Process
Connecting to real-world	• **Critical thinking**	• Computer-based Tutorials/Exploration

		Activities
Guided exploration	• **Brainstorming**	• Independent Study

Learning Skills:

Learning Skills are skills and habits are essential to success in school and in the workplace. Teachers report achievement on the six Learning Skills shown below the table below using letter codes:

♦Responsibility ♦Organization ♦Independent Work ♦Collaboration ♦Initiative
♦ Self-regulation

E = Excellent G = Good S = Satisfactory N = Needs Improvement.

.

Learning Skills	Sample Behaviors
Responsibility	The student fulfils responsibilities and commitments within the learning environment; completes and submits class work, homework, and assignments according to agreed-upon timelines; takes responsibility for and manages own behavior.
Organization	The student devises and follows a plan and process for completing work and tasks; establishes priorities and manages time to complete tasks and achieve goals; identifies, gathers, evaluates, and uses information, technology, and resources to complete tasks.
Independent Work	The student independently monitors, assesses, and revises plans to complete tasks and meet goals; uses class time appropriately to complete tasks; follows instructions with minimal supervision.
Collaboration	The student accepts various roles and an equitable share of work in a group; responds positively to the ideas, opinions, values, and traditions of others; builds healthy peer-to-peer relationships through personal and media-assisted interactions; works with others to resolve conflicts and build consensus to achieve group goals; shares information, resources, and expertise and promotes critical thinking to solve problems and make decisions.
Initiative	The student looks for and acts on new ideas and opportunities for learning; demonstrates the capacity for innovation and a willingness to take risks; demonstrates curiosity and interest in learning; approaches new tasks with a positive attitude; recognizes and advocates appropriately for the rights of self and others.
Self-Regulation	The student sets own individual goals and monitors progress towards achieving them; seeks clarification or assistance when needed; assesses and reflects critically on own strengths, needs, and interests; identifies learning opportunities, choices, and strategies to meet personal needs and achieve goals; perseveres and makes an effort when responding to challenges.

STRATEGIES FOR ASSESSMENT AND EVALUATION OF STUDENT PERFORMANCE

The tools below will be used for the three different types of assessments:

Assessment as Learning

Student Product

- Learning logs

Observation

- Whole class discussions

Conversation

- Student teacher discussions
- Small group discussions
- Pair works

Assessment for Learning

Student Product

- Assignment
- Pre-tests (scale/rubric)
- Quizzes (scale/rubric)

Observation

- Class discussions
- PowerPoint presentations (rubric)

Conversation

- Student teacher conferences (checklist)
- Small group discussions (checklist)
- Pair work (checklist)

Assessment of Learning

Student Product

- Assignment
- Tests (scale/rubric)
- Exam (scale/rubric)
- Project (rubric/scale)

Observation

- PowerPoint presentations (rubric)

- Performance tasks (rubric/scale)

Conversation

- Student teacher conferences (checklist)

- Question and answer session (checklist)

- Oral tests (scale/rubric)

Assessment Methods and Tools:

Term Assessment and Evaluation: 70% (Tests, Exams, Assignments, Projects)

Knowledge and Understanding (25%): Knowledge of content (e.g., facts, terms, definitions and procedures.) Understanding of content (e.g., concepts, principles, theories, relationships and methodologies)

Thinking and Inquiry (25%): Planning skills (e.g., focusing research, gathering information, selecting strategies, organizing a project) Processing skills (e.g., analyzing, interpreting, assessing, reasoning, gathering ideas, evaluating, seeking a variety of perspectives, forming conclusions)

Communication (25%): Expression of original ideas and information (e.g., logical organization) in oral, visual, and written forms

Application/Making connection (25%): The use of the knowledge and skills to make connections within and between various contexts.

Categories	50-59% (Level 1)	60-69% (Level 2)	70-79% (Level 3)	80-100% (Level 4)
Knowledge and Understanding - Subject-specific content acquired in each course (knowledge), and the comprehension of its meaning and significance (understanding)				
	The student:			
Knowledge of content (e.g., facts, terms, procedural skills, use of tools)	demonstrates limited knowledge of content	demonstrates some knowledge of content	demonstrates considerable knowledge of content	demonstrates thorough knowledge of content
Understanding of mathematical concepts	demonstrates limited understanding of content	demonstrates some understanding of content	demonstrates considerable understanding of content	demonstrates thorough and insightful understanding of content
Thinking - The use of critical and creative thinking skills and/or processes				

	The student:			
Use of planning skills -understanding the problem (e.g., formulating and interpreting the problem, making conjectures) -making a plan for problem solving	uses planning skills with limited effectiveness	uses planning skills with moderate effectiveness	uses planning skills with considerable effectiveness	uses planning skills with a high degree of effectiveness
Use of processing skills -carrying out a plan (e.g., collecting data, questioning, testing, revising, modelling, solving, inferring, forming conclusions) -looking back at the solution (e.g., evaluating reasonableness, making convincing arguments, reasoning, justifying, proving, reflecting)	uses processing skills with limited effectiveness	uses processing skills with some effectiveness	uses processing skills with considerable effectiveness	uses processing skills with a high degree of effectiveness
Use of critical/creative thinking processes (e.g., problem solving, inquiry)	uses critical / creative thinking processes with limited effectiveness	uses critical / creative thinking processes with some effectiveness	uses critical / creative thinking processes with considerable effectiveness	uses critical / creative thinking processes with a high degree of effectiveness
Communication - The conveying of meaning through various forms				
	The student:			
Expression and organization of ideas and mathematical thinking (e.g., clarity of expression, logical organization), **using oral, visual, and written forms** (e.g., pictorial, graphic, dynamic, numeric, algebraic forms; concrete materials)	expresses and organizes mathematical thinking with limited effectiveness	expresses and organizes mathematical thinking with some effectiveness	expresses and organizes mathematical thinking with considerable effectiveness	expresses and organizes mathematical thinking with a high degree of effectiveness
Communication for different audiences (e.g., peers and teachers) **and purposes** (e.g., to present data, justify a solution, express a mathematical argument) **in oral, visual,**	communicates for different audiences and purposes with limited effectiveness	communicates for different audiences and purposes with some effectiveness	communicates for different audiences and purposes with considerable effectiveness	communicates for different audiences and purposes with a high degree of effectiveness

and written forms				
Use of conventions, vocabulary, and terminology of the discipline (e.g., terms, symbols) **in oral, visual, and written forms**	uses conventions, vocabulary, and terminology of the discipline with limited effectiveness	uses conventions, vocabulary, and terminology of the discipline with some effectiveness	uses conventions, vocabulary, and terminology of the discipline with considerable effectiveness	uses conventions, vocabulary, and terminology of the discipline with a high degree of effectiveness
Application - The use of knowledge and skills to make connections within and between various contexts				
	The student:			
Application of knowledge and skills in familiar contexts	applies knowledge and skills in familiar contexts with limited effectiveness	applies knowledge and skills in familiar contexts with some effectiveness	applies knowledge and skills in familiar contexts with considerable effectiveness	applies knowledge and skills in familiar contexts with a high degree of effectiveness
Transfer of knowledge and skills to new contexts	transfers knowledge and skills to new contexts with limited effectiveness	transfers knowledge and skills to new contexts with some effectiveness	transfers knowledge and skills to new contexts with considerable effectiveness	transfers knowledge and skills to new contexts with a high degree of effectiveness
Making connections within and between various contexts (e.g., connections between concepts, representations, and forms within mathematics; connections involving use of prior knowledge and experience; connections between mathematics, other disciplines, and the real world))	makes connections within and between various contexts with limited effectiveness	makes connections within and between various contexts with some effectiveness	makes connections within and between various contexts with considerable effectiveness	makes connections within and between various contexts with a high degree of effectiveness

Resources required by the student:

Java Development Kit (JDK) from Oracle (A link to download this software for free is provided in the course).

Eclipse IDE (A link to download this free software for Mac or Windows is provided in the course)

RESOURCES

Ontario Ministry of Education (EDU) – curriculum documents page http://www.edu.gov.on.ca/eng/document/curricul/curricul.html

http://www.edu.gov.on.ca/eng/policyfunding/growSuccess.pdf

Study Guides

Note: This course is entirely remote learning.