Algonquin College

CST8116
Introduction to Computer Programming
Course Outline

2021-2022

Pre-requisite(s)	N/A
Co-requisite(s)	N/A
Prepared by	Stanley Pieda
Approved by	Sandra Brancatelli, M.Eng., P.Eng., Chair, ICT - Applications & Programming
Normative hours	70.00
Grading system	A+ Through F
Experiential Learning	No

Applicable Program(s)	Level	Core/Elective
0006X01FWO - Computer Eng. Technology - Comp. Science	1	Core
0006X03FWO - Computer Eng. Technology - Comp. Science	1	Core
0336X01FWO - Computer Programming	1	Core
0336X03FWO - Computer Programming	1	Core
0336X07PAO - Computer Programming	1	Core
0336X09FAO - Computer Programming	1	Core

Course Description

Possessing the fundamentals of logic, problem-solving and programming language structure provides a solid foundation for further study in the field. Students develop introductory knowledge of computer programming with emphasis on problem analysis and design, using algorithms, pseudocode, flowcharts, UML Class Diagrams and testing, with the Java programming language used as a means to implement problem solution designs. Through an introduction to the Java programming language students use sequential structures, selection structures, repetition structures, variables, constants, methods, constructors, one-dimensional arrays, object-oriented programming, classes, objects, abstraction, encapsulation, inputs, outputs, coding conventions and documentation. Theory is reinforced with application by means of practical laboratory assessments.

Vocational Learning Outcomes

This course provides the opportunity for you to achieve the following outcomes:

0006X01FWO - Computer Eng. Technology - Comp. Science

VLO 1 Identify, analyze, design, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Diagnose, troubleshoot, document and monitor technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Analyze, develop and maintain robust computing system solutions through validation testing and industry best practices. (T, A)

VLO 8 Adhere to ethical, social media, legal, regulatory and economic requirements and/or principles in the development and management of the computing solutions and systems. (T, A)

VLO 14 Develop, test and maintain software applications for systems integration. (T, A)

VLO 15 Apply general software principles of data structures, design patterns and structured data parsing in accordance with industry standards. (T, A)

0006X03FWO - Computer Eng. Technology - Comp. Science

VLO 1 Identify, analyze, design, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Diagnose, troubleshoot, document and monitor technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Analyze, develop and maintain robust computing system solutions through validation testing and industry best practices. (T, A)

VLO 8 Adhere to ethical, social media, legal, regulatory and economic requirements and/or principles in the development and management of the computing solutions and systems. (T, A)

VLO 14 Develop, test and maintain software applications for systems integration. (T, A)

VLO 15 Apply general software principles of data structures, design patterns and structured data parsing in accordance with industry standards. (T, A)

0336X01FWO - Computer Programming

VLO 1 Identify, analyze, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Implement robust computing system solutions through validation testing that aligns with industry best practices. (T, A)

VLO 8 Adhere to ethical, legal, and regulatory requirements and/or principles in the development and management of computing solutions and systems. (T)

VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. (T, A)

VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. (T, A)

0336X03FWO - Computer Programming

VLO 1 Identify, analyze, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Implement robust computing system solutions through validation testing that aligns with industry best practices. (T, A)

VLO 8 Adhere to ethical, legal, and regulatory requirements and/or principles in the development and management of computing solutions and systems. (T)

VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. (T, A)

VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. (T, A)

0336X07PAO - Computer Programming

VLO 1 Identify, analyze, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Implement robust computing system solutions through validation testing that aligns with industry best practices. (T, A)

VLO 8 Adhere to ethical, legal, and regulatory requirements and/or principles in the development and management of computing solutions and systems. (T)

VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. (T, A)

VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. (T, A)

0336X09FAO - Computer Programming

VLO 1 Identify, analyze, develop, implement, verify and document the requirements for a computing environment. (T, A)

VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. (T, A)

VLO 4 Implement robust computing system solutions through validation testing that aligns with industry best practices. (T, A)

VLO 8 Adhere to ethical, legal, and regulatory requirements and/or principles in the development and management of computing solutions and systems. (T)

VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. (T, A)

VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. (T, A)

Assessment Levels —T: Taught **A**: Assessed **CP**: Culminating Performance

Essential Employability Skills

This course contributes to your program by helping you achieve the following Essential Employability Skills:

EES 1 Communicate clearly, concisely and correctly in the written, spoken and visual form that fulfills the purpose and meets the needs of the audience. (T, A)

EES 2 Respond to written, spoken or visual messages in a manner that ensures effective communication. (T)

EES 3 Execute mathematical operations accurately. (T, A)

EES 4 Apply a systematic approach to solve problems. (T, A)

EES 5 Use a variety of thinking skills to anticipate and solve problems. (T)

EES 10 Manage the use of time and other resources to complete projects. (T, A)

EES 11 Take responsibility for one's own actions, decisions and consequences. (T, A)

Assessment Levels —T: Taught **A**: Assessed **CP**: Culminating Performance

Course Learning Requirements / Embedded Knowledge and Skills

When you have earned credit for this course, you will have demonstrated the ability to:

1. Analyze a problem and define a solution in the form of an algorithm.

- Define an algorithm as a sequence of steps used to solve a problem.
- Use sequence structures, decision structures, and repetition structures in forming an algorithm.
- Use variables, constants, operators (arithmetic, comparison, and logic), one dimensional arrays, inputs, outputs and modules (methods) in forming an algorithm.
- Define and use objects, properties and behaviours, in forming an algorithm
- Refine existing algorithms to improve them.
- Determine the viability of the solution to the given problem.
- Differentiate between data, processing, and information.
- Recognize that there are multiple programming paradigms, examples include but are not limited to imperative (procedural, object-oriented), and declarative (logical, functional).

2. Document an algorithm using flowcharts, pseudocode and UML class diagrams.

- Define pseudocode and flowcharts, and explain the differences between them.

- Use flowcharts and pseudocode to document sequence structures, decision structures, and repetition structures of an algorithm.
- Use flowcharts and pseudocode to document algorithms.
- Use flowcharts and pseudocode to document variables, constants, arrays, program inputs, outputs of an algorithm.
- Use flowcharts and pseudocode to document arithmetic, comparison, and logic operators of an algorithm.
- Use flowcharts and pseudocode to document modules (methods), the calling of methods, and the returning from methods.
- Use UML Class Diagrams to document object designs.

3. Test algorithms and / or associated flowcharts, using a test plan.

- Test using a paper-oriented desk-check, using the specified format.
- Define the expected outputs to evaluate the validity of the expectations.
- Identify boundary conditions.
- Develop test conditions for both sides of boundaries.
- Develop comprehensive test data, including valid and invalid data.
- Test sequence structures, decision structures, repetition structures, and arithmetic and logic expressions, module (method) parameter and return values, using test data.

4. Describe and manipulate data as represented in a computer, using the mathematics available within a computer.

- Use data types numeric (integer, floating), string (text), as seen in a high-level programming language (e.g. Java).
- Use unary and binary arithmetic operators, comparison operators, logic operators and order of operations as seen in a high-level programming language (e.g. Java).

5. Implement the solution of a given problem by writing the appropriate code in a high-level language (Java).

- Use algorithms, flowcharts, programmer-centric test plans, UML Class Diagrams and reference to the Java Application Programmer Interface (API) documents to guide development of a Java program.
- Use the Java compiler to detect syntax errors, and to produce Java byte-code, and use the Java Virtual Machine to execute Java byte-code.
- Use Java sequence structures, selection structures (if, switch, ternary operator), and iterative repetition structures (while, do-while, for) to implement algorithms in Java including comparisons based on primitive types, as well as the String reference type.
- Perform input and output operations in Java at run time via the standard input stream with class Scanner, the standard output stream with formatted output (print, println, printf) and String concatenation, as well as use input from command line arguments at program start up.
- Explain and use Java modifiers, keywords, and reserved words including public, private, static, and final (final as used only with variables), this, and null.
- Use variables in a Java program including primitive and reference type variables, constant variables, instance and non-instance variables, variable scope and scope resolution, in the context of method-call-stack, heap-memory-area, classes, constructors, methods and objects.
- Use the object-oriented Programming OOP) concepts of abstraction and encapsulation to design and create Java classes and objects including constructors, accessors, mutators, and methods.
- Use references to one-dimensional array objects containing elements of primitive type values and references to one-dimensional array objects containing elements of reference type values (references to reference type instances) to store and manipulate data and /or process data into information.
- Practice industry recognized Java coding conventions in authoring Java program source code, and documenting Java code with single-line and multiline comments.

6. Describe in detail the representation of data in Java and be able to manipulate data as represented in the mathematics used within a Java program.

- Detail the characteristics of the primitive data types used in Java, including size, format, range and precision.
- Perform casts between Java primitive data types with reference to possible truncation and loss of precision.
- Use the primitive wrapper classes to convert numbers represented as String data, into appropriate numerical primitive type
- Evaluate expressions written in Java containing one or more of unary and binary arithmetic operators, comparison operators, logic operators, and order of operations.
- Explain in detail the concepts of integer overflow and underflow.
- Explain the concept of Not a Number (NaN), in the context of the floating numerical types.
- Use utility methods of the Java API Math class.
- Explain why float and double are not suited for financial calculations listing appropriate alternatives.

7. Install and use a Java Development Kit (JDK), documentation libraries, Integrated Development Environment (IDE) and version control software.

- Install and use a JDK, either Oracle JDK or an Open-JDK.
- Install and use the Eclipse IDE.
- Install and use Git software.
- Compile and execute Java programs consisting of multiple classes from the command prompt environment, as well as in the Eclipse IDE.
- Use Git commands within the command prompt environment, as well as within the Eclipse IDE.
- Explain the importance of reviewing software license agreements with respect to using software development tools and API libraries.

8. Create programmer-oriented test plans and memory maps using them in conjunction with an integrated debugger to evaluate and document Java program execution.

- Use programmer-oriented test plan to predict variable value(s) at any point during program execution.
- Create memory maps to document the state of a Java program at points of execution, based on information reported by the debugging tools including stack, heap, primitive and reference types (including one-dimensional arrays of primitives and references).
- Use the IDE debug tools: breakpoints, step-in, step-over, locals, watch, and call stack.
- Use console output for debugging.
- Use the debugger to find run-time logic errors, correct those errors, verifying the corrective action.
- Use debugger to verify the correct logical operation and correct any logic errors detected (to be used in conjunction with the IDE debugger)

9. Create a local Git repository and use Git commands to perform version control tasks on a Java program.

- Create, modify, and monitor a local Git repository using git init, git add, git commit, git status, git log.

Learning Resources

Textbooks Required:
Joyce Farrell. 2018. Programming Logic & Design Comprehensive. 9th Ed. Cengage Learning.
ISBN-13: 978-1-337-10207-0. ISBN-10: 1-337-10207-5

Cay Horstmann. 2019. Big Java Early Objects. 7th Ed. Wiley.
eText: 978-1-119-49909-1 or loose-leaf paper: 978-1-119-74020-9
(eText is the preferred format of the Horstmann text)

Computer Required:
This course is part of the Bring Your Own Device (laptop) program initiative at Algonquin College. Students are required to have a functioning laptop at all lecture and lab classes. The specifications for the required laptop and additional information about the BYOD program initiative can be found at www.algonquincollege.com/byod/device.htm .

Learning Activities

Hybrid Activities
Lectures and Readings
Software demonstrations
Online lectures, readings and software demonstrations for online students
Individual lab exercises (smaller tasks) and assignments (larger tasks)
Research of course-related material

Hybrid Activities
This course consists of 2 hours of lecture (on-campus or online) and 2 hours of lab work per week, as well as one hour per week of online material and activities available through Brightspace software.

Lectures and Readings
Theoretical course material will be presented, aided by use of overhead projections, videos, demonstrations and brief lecture notes.
This course is hybrid. Approximately one hour per week outside of regular classroom lecture time will be required to review online materials and complete activities. This is in addition to your own study time.

Pre-defined Evaluation / Earning Credit

The following list provides evidence of this course's learning achievements and the outcomes they validate:

Lab Activity(ies) (10%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, EES 1, EES 3, EES 4

Assignment(s) (20%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, CLR 9, EES 1, EES 3, EES 4, EES 10, EES 11

Hybrid Assignment(s) (10%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, CLR 9, EES 1, EES 3, EES 4

Quiz(zes)/Test(s) (20%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, CLR 9, EES 1, EES 3, EES 4

Practical Assessment (s) (20%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, EES 1, EES 3, EES 4, EES 10, EES 11

Final Exam (20%)

Validates Outcomes: CLR 1, CLR 2, CLR 3, CLR 4, CLR 5, CLR 6, CLR 7, CLR 8, CLR 9, EES 1, EES 3, EES 4, EES 10, EES 11

Prior Learning Assessment and Recognition

Students who wish to apply for Prior Learning Assessment and Recognition (PLAR) need to demonstrate competency at a post-secondary level in all outlined course learning requirements. Evidence of learning achievement for PLAR candidates includes:

- Portfolio
- Challenge Exam
- Project/Assignment

Other Information

Grade Scheme

Final Grade	Mark Equivalent	Numeric Value	Final Grade	Mark Equivalent	Numeric Value
A+	90% - 100%	4.0	A	85% - 89%	3.8
A-	80% - 84%	3.6	B+	77% - 79%	3.3
B	73% - 76%	3.0	B-	70% - 72%	2.7
C+	67% - 69%	2.3	C	63% - 66%	2.0
C-	60% - 62%	1.7	D+	57% - 59%	1.4
D	53% - 56%	1.2	D-	50% - 52%	1.0
F	0% - 49%	0	FSP	0	0

Course Related Information

Please refer to the Course Section Information (CSI) / weekly schedule for specific course-related information as provided by your professor.

IN ORDER TO CONTINUE IN THE FOLLOW-UP COURSE CST8132 IN SECOND SEMESTER OF COMPUTER ENGINEERING TECHNOLOGY-COMPUTING SCIENCE OR CST8284 IN SECOND SEMESTER OF COMPUTER PROGRAMMING, STUDENTS MUST ACHIEVE AT LEAST A GRADE OF C- in CST8116.

Note: You will be expected to demonstrate some lab work to your professor during the course, details will be made available in the CSI as well as the assessment materials. There is also practical assessment(s) in the form of a lab exam(s) conducted either in the last week of the course, or midway and at the end of the course. In this (these) case(s) you will be provided a practical assessment at the start of the lab period and must complete the tasks assigned by the end of the lab period.

In order to pass the course, the student must have a grade of at least 50% on both the theory component as well as in the applied (i.e. lab) component. i.e. Even if your combined grade exceeds 60% for the entire

course, if you fail either the theory component or the applied component you will not achieve a passing grade in the course.

Note that the Lab Activity(ies), Assignment(s), and Practical Assessment constitute the applied component of the course. The Hybrid Assignment(s), Quiz(zes)/Test(s), and Final Exam constitute the theory component.

The Information and Communications Technology Department requires that all course assignments (homework exercises, laboratory work, projects, etc) be submitted by students using a standard, which could be specific to one or more courses. Professors will ensure, at the beginning of the term that students are advised of the exact details of these course specific submission requirements. Professors will also post them online alongside the course outline. Student submissions that do not meet the course published submission standards may not be marked, and may incur a penalty of up to 100% of the submission mark.

All students are required to write the final exam. There are no provisions for 'making up' a missed final exam. If, because of being off-track in your program or some unforeseen circumstance, you note that there is a scheduling conflict in your final exam schedule, it is your responsibility to alert your course professor no later than one week before final exams start, to allow for any special arrangements.

Department Information

STUDENT ACADEMIC RESPONSIBILITIES

Each student is responsible for:
• Knowing the due dates for marked out-of-class assignments.
• Attending all classes and knowing the dates of in-class marked assignments and exercises.
• Maintaining a folder of all work done in the course during the semester for validation claims in cases of disagreement with faculty.
• Keeping both paper and electronic copies of all assignments, marked and unmarked, in case papers are lost or go missing.
• Regularly checking both Brightspace announcements as well as one's Algonquin e-mail account for important messages from both professors and college administration.
• Participating in on-line and classroom exercises and activities as required.
• Retaining course outlines for possible future use to support applications for transfer of credit to other educational institutions.

Department Grading Policy - For all courses that have both a theory and practical (lab) component, students must have a grade of at least 50% (or "D-") on both the theory component as well as in the practical (i.e. lab) component in order to achieve a passing grade in the course. i.e. Even if your combined grade exceeds 50% for the entire course, if you fail either the theory component or the practical component, you will not achieve a passing grade in the course.

Harassment/Discrimination/Violence will not be tolerated. Any form of harassment (sexual, racial, gender or disability-related), discrimination (direct or indirect), or violence, whether involving a professor and a student or amongst students, will not be tolerated on the college premises. Action taken will start with a formal warning and proceed to the full disciplinary actions as outlined in Algonquin College Policies - HR22 and SA07.
Harassment means one or a series of vexatious comment(s) (whether done verbally or through electronic means), or conduct related to one or more of the prohibited grounds that is known or ought reasonably to be known to be unwelcome/unwanted, offensive, intimidating, derogatory or hostile. This may include, but is not limited to: gestures, remarks, jokes, taunting, innuendo, display of offensive materials, offensive graffiti, threats, verbal or physical assault, stalking, slurs, shunning or exclusion related to the prohibited grounds.

For further information, a copy of the official policy statement can be obtained from the Student Association.

Violation of the Copyright Act
General – The Copyright Act makes it an offence to reproduce or distribute, in whatever format, any part of a publication without the prior written permission of the publisher. For complete details, see the Government of Canada website at http://laws.justice.gc.ca/en/C-42 . Make sure you give it due consideration, before deciding not to purchase a textbook or material required for your course.

Software Piracy - The Copyright Act has been updated to include software products. Be sure to carefully read the licensing agreement of any product you purchase or download, and understand the terms and conditions covering its use, installation and distribution (where applicable). Any infringement of licensing agreement makes you liable under the law.

Disruptive Behaviour is any conduct, or threatened conduct, that is disruptive to the learning process or that interferes with the well being of other members of the College community. It will not be tolerated. Members of the College community, both students and staff, have the right to learn and work in a secure and productive environment. The College will make every effort to protect that right. Incidents of disruptive behaviour must be reported in writing to the departmental Chair as quickly as possible. The Chair will hold a hearing to review available information and determine any sanctions that will be imposed. Disciplinary hearings can result in penalties ranging from a written warning to expulsion.

For further details, consult the Algonquin College Policies AA32, SA07 and IT01 in your Instaguide.

College Related Information

Algonquin College's policies have been developed to ensure the health, safety and security of all students, faculty and staff, and the proper and fair operation of the College as an academic institution and employer. Please refer to the Algonquin College Policies website for the most current policy information available at http://www.algonquincollege.com/policies/.

Students are especially encouraged to be aware of the following College expectations

Academic Integrity

Algonquin College is committed to the highest standards of academic integrity, and students are expected to uphold these standards as part of the learning process. Any academic work submitted by a student is expected to be their own work, unless designated otherwise and all sources must be attributed. All students should be familiar with the Algonquin College policy AA48: Academic Integrity . In some courses, online proctoring may be used to discourage cheating. Additional information can be found at https://www.algonquincollege.com/studentsupportservices/student-learning-kit/preparing-to-learn-online/. Students with any questions about the course expectations for academic dishonesty and plagiarism are encouraged to speak to their professor.

Centre for Accessible Learning

Students with visible and/or non-visible disabilities are encouraged to register with the Centre for Accessible Learning (CAL) in order to be eligible for appropriate learning supports and/or accommodations. Students are strongly encouraged to make an appointment at the Centre for Accessible Learning as early as possible when starting a program. Once your needs are identified, a Letter of Accommodation (LOA) will be issued which you can share with your professors. If you are a returning student, please ensure that professors are given a copy of your LOA each semester.

College Email

Students at Algonquin College are provided with a college email account. This is the address that will be used when the College, your professors, or your fellow students communicate important information about your program or course activities. Your network credentials can be found in the ACSIS portal and you are expected to check your Algonquin email regularly and to use it to send and receive college-related email. Support is available through the college Information Technology Service (ITS) at: https://www.algonquincollege.com/its/

Retroactive Accommodations

Students are expected to meet evaluation and completion deadlines as stated in course outline and course section information documents. In circumstances where evaluation and/or completion deadlines are missed or student performance has been affected by a temporary or permanent disability (including mental health), interim or retroactive accommodations may be considered. In such instances, please consult your course faculty member. For other situations where deferral of evaluations may be warranted, please consult Algonquin College Policy AA21: Deferred Evaluation.

Student Course Feedback

Algonquin College's invites students to share their course experience by completing a student course feedback survey for each course they take. For further details consult Algonquin College Policy AA25: Student Course Feedback.

Use of Mobile Devices in Class

With the proliferation of small, personal mobile devices used for communications and data storage, Algonquin College believes there is a need to address their use during classes and examinations. During classes, the use of such devices can be disruptive and disrespectful to others. During examinations, the use of such devices may facilitate cheating. For further details consult Algonquin College Policy AA32: Use of Mobile Devices in Class

Technology Requirements

As Algonquin College continues to respond to public health guidelines, many courses will be offered through remote delivery. As such, students will be required to have access to a computer and to the internet. There may also be additional technology-related resources required to participate in a course that are not included in the course materials fee, such as headphones, webcams, specialized software, etc. Details on these requirements can be found in the Course Section Information of the course outline for each course available on Brightspace.

Transfer of Credit

It is the student's responsibility to retain course outlines for possible future use to support applications for transfer of credit to other educational institutions.