



Carleton
UNIVERSITY

Department of
**Systems and
Computer Engineering**

SYSC 2004

Object-Oriented Software Development

Calendar description

Designing and implementing small-scale programs as communities of collaborating objects, using a dynamically-typed or statically-typed programming language. Fundamental concepts: classes, objects, encapsulation, information hiding, inheritance, polymorphism. Iterative, incremental development and test-driven development.

Includes: Experiential Learning Activity.

Lectures three hours a week, laboratory two hours a week.

<http://calendar.carleton.ca/undergrad/courses/SYSC/>

Prerequisites

SYSC 2006 or permission of the department, and second-year status in Engineering.

Precludes additional credit for SYSC 1101, COMP 1006 and COMP 1406.

Prior knowledge

Students should:

- understand the concepts that underlie most imperative programming languages and be able to use this knowledge to help them learn new languages;
- understand how memory is managed by an executing program, and demonstrate this knowledge pictorially;
- understand different designs for simple abstract linear collections such as lists (vectors), queues and stacks;
- be able to construct simple recursive functions;
- and be prepared to undertake this course that provides a thorough introduction to object-oriented programming principles.

Course objectives

Designing and implementing small-scale programs as communities of collaborating objects, using a dynamically-typed or statically-typed programming language. Fundamental concepts: classes, objects, encapsulation, information hiding, inheritance, polymorphism. Iterative, incremental development and test-driven development.

List of topics

- Object-oriented concepts (class, object, encapsulation, fields/instance variables, constructors and instance methods, accessors and mutators, inheritance, polymorphism, casting)
- Classes as types; References
- UML representation of a class, an object; Introduction to the Unified Modeling Language (UML)
- Generic collections
- Documenting
- Testing and debugging
- Introduction to Design Patterns
- Event handling

Learning outcomes

By the end of this course, students should be able to:

- Identify the fundamental concepts of object-oriented programming (classes, objects, encapsulation, information hiding, inheritance, polymorphism).
- To implement small-scale programs as communities of interacting (collaborating) objects.
- To apply lightweight, modern techniques commonly used during object-oriented software development (iterative, incremental development; test-driven development).
- To draw basic class and object diagrams.
- To apply their object-oriented knowledge to developing simple graphical user interfaces.

Graduate Attributes (GAs)

The Canadian Engineering Accreditation Board requires graduates of engineering programs to possess 12 attributes at the time of graduation. Activities related to the learning outcomes listed above are measured throughout the course and are part of the department's continual improvement process. Graduate attribute measurements will not be taken into consideration in determining a student's grade in the course. For more information, please visit: <https://engineerscanada.ca/>.

Graduate Attribute	Learning outcome(s)
1.4.S: Knowledge Base: Developed: Programming and algorithms	2, 5
1.8.S: Knowledge Base: Introductory: Software engineering	1
3.1: Investigation: Introductory: Complex problem assessment	3, 5
3.3: Investigation: Introductory: Experimental procedure	2, 3, 4
4.2: Design: Introductory: Detailed design specifications and requirements	3
4.5: Design: Introductory: Design implementation / task(s) definition	3
5.1: Use of Engineering Tools: Introductory: Diagrams and engineering sketches	4

Accreditation Units (AUs)

For more information about Accreditation Units, please visit:

<https://engineerscanada.ca/>.

The course has a total of 49 AUs, divided into:

- Engineering Science: 60%
- Engineering Design: 40%

Instructor and TA contact

Specific to course offering (tbd)

Textbook (or other resources)

Specific to course offering (tbd)

Evaluation and grading scheme

Specific to course offering (tbd)

Breakdown of course requirements

Specific to course offering (tbd)

Tentative week-by-week breakdown

Specific to course offering (tbd)

General regulations

Specific to course offering (tbd)