Carleton UNIVERSITY | Department of Systems and Computer Engineering

SYSC 2100
Algorithms and Data Structures

Calendar description

Thorough coverage of fundamental abstract collections: stacks, queues, lists, priority queues, dictionaries, sets, graphs. Data structures: review of arrays and linked lists; trees, heaps, hash tables. Specification, design, implementation of collections, complexity analysis of operations. Sorting algorithms.

Includes: Experiential Learning Activity.

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

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

Prerequisites

SYSC 2006 with a minimum grade of C-, and second-year status in Engineering.

Precludes additional credit for SYSC 2002 (no longer offered) and COMP 2402.

Prior knowledge

Students should have knowledge of:

- Types and variables
- Function argument passing
- Recursion
- Data structures (e.g., arrays, linked lists)
- Encapsulation and information hiding
- Object-oriented programming

Course objectives

To introduce students to common programming techniques/algorithms (recursion, searching and sorting, etc.). To introduce students to several fundamental abstract data types (ADTs) and data structures. Students will learn how to develop the specifications for ADTs, design their underlying data structures, and implement the ADTs as Java classes. Some common applications of these ADTs will be presented. To introduce students to techniques for designing and analyzing algorithms.

List of topics

- Recursion
- Data Abstraction

- Example Abstract Data Type: Linked Lists
- Problem Solving with Recursion
- Stacks
- Queues
- Algorithm Efficiency/Complexity
- Trees
- Tables and Priority Queues
- Advanced Tables: Hashing
- Graphs

Learning outcomes

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

- Estimate the computational complexity (Big-O notation etc.) of functions and basic algorithms.
- Choose the appropriate data structures and algorithms for a given application/problem.
- Write recursive Java methods to solve a variety of computational problems. This includes the ability to:
    a. Write pseudo code instructions of an algorithm
    b. Abide by Java programming rules
    c. Use object-oriented programming
- Solve problems that involve the management of data. This includes the ability to:
    a. Develop specifications for ADTs
    b. Design underlying data structures
    c. Implement ADTs as Java classes

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	1
2.2: Problem Analysis: Developed: Approach to the problem	2, 3, 4
2.4: Problem Analysis: Developed: Interpreting the solution – validity of results	2, 3, 4
3.1: Investigation: Developed: Complex problem assessment	2, 3, 4
3.3: Investigation: Introduced: Experimental procedure	2, 3, 4
3.5: Investigation: Introduced: Interpretation of data (synthesis) and discussion	2, 3, 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)