COMP 2402 (Early Summer 2026)

Abstract Data Types and Algorithms

Course Information

Instructor: Prof. Alexa Sharp (she/her)	**Class Location:** See Carleton Central
Email: alexasharp3@cunet.carleton.ca	**Lecture Times:** Tue/Thu 5:35-8:25pm
Course Website: brightspace	**Course Forum:** discord
Best Ways to be in Touch: on discord, via email, or during student hours	**Teaching Assistants:** a list of TAs and their office hours will be posted on Brightspace

Course Calendar Description

Introduction to the design and implementation of abstract data types and to complexity analysis of data structures. Topics include: stacks, queues, lists, trees and graphs. Special attention is given to abstraction, interface specification and hierarchical design using an object-oriented programming language.

Precludes additional credit for SYSC 2100.
Prerequisites: COMP 1406 or COMP 1006 with a **minimum grade of C-**.

Inclusivity Statement

I am committed to fostering an environment for learning that is inclusive for everyone regardless of gender identity, gender expression, sex, sexual orientation, race, ethnicity, ability, age, class, etc. All students in the class, the instructor, the course staff, and any guests should be treated with respect during all interactions.

Land Acknowledgement

I would like to acknowledge that the land on which we gather is the traditional, unceded territory of the Anishinaabe Algonquin nation. In doing so, I acknowledge that I and Carleton University have a responsibility to the Algonquin people and a responsibility to adhere to Algonquin cultural protocols. More information on Prof Alexa's work to take some responsibility can be found on brightspace in the module titled "Beyond the Land Acknowledgement."

Learning Materials

- **Textbook:** Pat Morin's *Open Data Structures (Java)* — free online
- **Java environment** of your choice

Course Goals

Students learn to **design, choose, and reason about data structures and algorithms**.

By the end of the course, you should be able to:

- **Model problems using appropriate abstractions (ADTs)**
  What problem am I solving? What operations matter? What guarantees do I need?
- **Move from problem to solution**
  Problem → Algorithm → Operations → ADT(s) → Implementation → Runtime
- **Evaluate tradeoffs**
  time vs space; worst-case vs amortized vs expected; structure vs flexibility
- **Control computational cost**
  avoid unnecessary work; recognize expensive operations; choose better strategies
- **Use invariants to reason about correctness and guarantees**
  what must always be true, what does it guarantee?
- **Identify and use "workhorse" operations**
  fundamental operations that simplify implementation and analysis
- **Combine and compose structures when needed**
  layer or partition responsibilities across structures
- **Communicate algorithmic ideas clearly**
  explain reasoning, justify choices, and analyze performance

Topics Overview

Readings, videos and course work for each week are available on Brightspace.

Week	Topics
1	Foundations: ADTs, algorithms, problem solving
2	Dynamic arrays, circular arrays, amortized analysis
3	Combining structures and lists
4	Skiplists and binary search trees
5	Balanced trees
6	Heaps and hash tables
7	Graphs and synthesis

Course Format (Flipped + Compressed)

This is a **compressed 6.5-week course** covering the full content of a standard 13-week course. Each week (Week 1 excepted) follows this cycle:

Before Tuesday

- Watch **~3–4.5 hours of videos** (views are tracked)
- Start the weekly lab (released prior Thursday)

Tuesday Class (3 hours)

- Activity-based session (participation is tracked)
- Build intuition and core ideas

Between Tuesday and Thursday

- Complete the lab (due **Wednesday 11:59pm**, +24h grace)

Thursday Class (3 hours)

- Activities for more practice (participation is tracked)
- **Checkpoint A (≈30 min)**
- **Checkpoint B (≈30 min)**

Then the next week begins.

A Note on Learning

You need both:

- **Videos** → core ideas
- **Activities** → problem-solving and reasoning

Skipping either makes the course much harder. Both will be tracked.

How to Succeed

Students who do well:	Students who struggle:
Watch videos **before Tuesday**	Only watch enough video to complete labs
Write things down during activities	"Just think" or avoid committing to answers
Engage in activities (even when unsure)	Disengage during activities or leave early
Ask questions early	Struggle alone and/or at length
Focus on reasoning, not memorization	Rely on tools instead of developing understanding

Assessment Overview

This course uses a **bucket-based grading system**.

Category	Available	Bucket Capacity	Schedule & Deadlines
Checkpoints* (12 × 4.5%)	54%	50%	**Thursdays** in class
Labs (6 × 4%)	24%	20%	**Wednesdays 11:59pm**
Engagement (Participation)	13%	12%	Every class
Final Exam*	18%	18%	Exam Period
Bonus	>3%	3%	See below

* See Passing Requirements

How the Bucket System Works

Each category has **more marks available than needed**.

- Your total in a category is **capped at the bucket capacity**
- Extra marks provide flexibility

This means:

- Missing one assessment usually won't hurt much, if at all
- Doing poorly still contributes and helps fill your bucket (every bit helps)
- Consistent effort is rewarded

This system is at least as favourable as a "drop the lowest," while encouraging continued effort.

Checkpoints (50%)

- 12 total (2 per week on Thursdays, Weeks 2–7), **~30 minutes each**
- **4.5% (54% total available → capped at 50%)**
- Assess your understanding of the material without external assistance

Format: In-person, paper-based, individual, closed notes (cheat sheet provided)

Rules:

- Must be written at scheduled time
- **No make-up checkpoints**
- Missing a Thursday = missing two checkpoints

The bucket system provides flexibility; repeated absences will affect your grade.

Labs (20%)

- 6 labs, **4% each (24% total available → capped at 20%)**
- **Due: Wednesdays at 11:59pm** (+24h grace)
- **Submitted** via Brightspace, partially **autograded**

Labs:

- Provide **hands-on practice** and prepare you directly for checkpoints
- Must be completed individually

Submit early. Last-minute issues are not grounds for extension.

Engagement (12%)

- Many opportunities across 13 class meetings **(13% total available → capped at 12%)**
- Assessed across all classes.

May include:

- Wooclap submissions
- Written activity work
- Group participation
- Responsiveness when called on

Important:

- Attendance ≠ engagement
- Passive attendance may receive little or no credit
- Missing multiple classes will significantly impact this component

Final Exam (18%)

- **In-person, paper-based**
- Cumulative
- Mostly multiple-choice and/or short-answer

Bonus (3%)

A bonus bucket of up to **3%** may be filled through:

- Overflow from other categories (when you exceed a bucket)
- Optional activities (e.g., drills)

Added after all other categories are computed.

Passing Requirements

To pass, you must:

1. Pass the **final exam**, and
2. Pass the **combined Checkpoints + Final (68%)**

Late and Missed Work Policy

The grading structure has **built-in flexibility** through its bucket system.

- **No extensions** or make-ups for checkpoints, labs, or participation
- Please do not email about individual missed work as the buckets already account for it

This ensures fairness and reduces administrative overhead.
For exceptional cases affecting a large portion of the course, contact the prof within **24 hours**.

Undergraduate Academic Advisors

The Undergraduate Advisor for the School of Computer Science is available in Room 5302 HP, or by email at scs.ug.advisor@carleton.ca. The undergraduate advisor can assist with information about prerequisites and preclusions, course substitutions/equivalencies, understanding your academic audit and the remaining requirements for graduation. The undergraduate advisor will also refer students to appropriate resources such as the Science Student Success Centre, Learning Support Services and Writing Tutorial Services.

SCS Computer Laboratory

Students taking a COMP course can access the SCS computer labs. The lab schedule and location can be found at: https://carleton.ca/scs/tech-support/computer-laboratories/. All SCS computer lab and technical support information can be found at: https://carleton.ca/scs/tech-support/. Technical support staff may be contacted in-person or virtually, see this page for details: https://carleton.ca/scs/tech-support/contact-it-support/

Mental Health and Wellness

University life can be challenging, especially in a compressed course.

Students are encouraged to make use of the supports available through Carleton's Wellness Services: https://wellness.carleton.ca/

If you are struggling, reaching out early can make a significant difference.

Academic Accommodations and Regulations

Academic Accommodation

The Carleton University Information on Academic Accommodation applies to this course. Here is information on how to apply for academic accommodation. If you are allowed extra time on tests, ventus should take care of it (and it is your responsibility to confirm such accommodations.) If there are accessibility-related needs, please contact me early so we can coordinate with official accommodations.

Chat GPT/Generative AI Usage

Allowed:

- Syntax help
- Debugging assistance

Not allowed:

- Designing algorithms
- Writing solutions

Using AI to bypass that process will leave you unprepared for checkpoints and exams.

Academic Integrity

All submitted work must reflect **your own ideas and understanding**.

For labs:

- Discussion of high-level ideas is allowed
- Code must be written independently

Not allowed:

- Copying or sharing code
- Using online or repository code
- Using AI to generate solutions or algorithms

Typing code yourself is not enough — the ideas must be your own.
If you cannot explain how your code works, you should not submit it.

Violations of academic integrity will be handled according to Carleton University's Academic Integrity Policy. A list of standard sanctions in the Faculty of Science can be found here. Additional details about this process can be found on the Faculty of Science Academic Integrity website.

Student Rights & Responsibilities

Students are expected to act responsibly and engage respectfully, collaboratively, and constructively with other students and members of the Carleton and the broader community. Participation requires a classroom where people feel comfortable contributing. See the 7 Rights and Responsibilities Policy for details regarding the expectations of non-academic behaviour of students. Those who participate with another student in the commission of an infraction of this Policy will also be held liable for their actions.

Student Concerns

If you have any concerns regarding this course, your first point of contact is me. Please email me or visit during my student hours, and I will do my best to address your concerns. If I cannot resolve the issue, the next point of contact is the School of Computer Science at studentconcerns@scs.carleton.ca. If the concern remains unresolved, the final point of contact is the Office of the Dean of Science at ODScience@carleton.ca. Please follow this order of contact.

Note: You can also bring your concerns to Ombuds Services or Carleton Equity and Inclusive Communities.