EECS2011: Fundamentals of Data Structures

Sections N & Z – Winter 2022

Last Updated: January 10, 2022

Course Syllabus

1 Course Policies

To ensure a smooth, fair, and effective online delivery of this course:

1. **Plagiarism**: When submitting each of your **assignments/labs** or **programming tests**, you claim that it is **solely** your work. It is considered as **an violation of academic integrity** if you copy or share **any** parts of your work (e.g., code, notes) during **any** stage of your development. The instructor and TAs **will** examine **all** submitted code, and suspicious submissions will be reported *immediately* to Lassonde as *a breach of academic integrity*. **We do not tolerate academic dishonesty**, so please be fully responsible for your learning.

2. **Online Submission/Assessment**: **Stringent deadlines** are imposed on all **scheduled written & programming tests**, as well as **assignments/labs**. An **exam** is scheduled online (via eClass) with **stringent timing requirements** (start time, duration, and end time to be announced by the registrar office).

   All announced deadlines are in the Eastern Time Zone (Toronto time). Students on a different time zone must figure out the corresponding local time.

   Students are responsible for **taking proactive steps and/or seeking assistance** well in advance to ensure that their technical setup (e.g., stable internet connection, a computer which does not freeze sporadically) allows them to complete and submit each assessment item (written test, programming test, lab, assignment, exam) in time.

   Rationales for this policy are to: **urge** students with technical issues to take steps or seek assistance to fix/improve them (otherwise, how can they benefit from the online setting in the first place?); and **discourage** students trying to take an unfair advantage (e.g., a student ignorant of the submission deadline or starting late may claim technical failure to have an extension, a student who has already seen the exam questions may claim network/computer failure in order to gain extra time or a deferred exam).

   When it comes to assessments, your instructor's priorities are fairness and academic integrity.

3. **No Team Work**: All **labs/assignments** and **written & programming tests** are to be developed and completed **individually** (i.e., **team work is forbidden**). This is meant for avoiding students having difficulties finding a suitable teammate and disputes between teammates (e.g., non-responsiveness, overdue progress, last-minute notice of withdrawal): the online nature of this course would only exacerbate these problems.

4. **Late Enrolment**: Students who are not yet officially registered should assume an eventual successful enrolment into the course and are responsible for: **1)** contacting the section instructor **within Week 1** for course information (e.g., lecture materials, labs/assignments access and deadlines); and **2)** studying lecture videos, attending Q&A sessions, taking quizzes, and submitting labs/assignments in time.

   No deadline extensions or deferred tests will be accommodated.

2 Academic Integrity

Labs/Assignments

- All labs/assignments are to be completed ***individually***: no group work is allowed.

  TAs will perform thorough checks on **all** lab submissions: convincingly suspicious submissions will be reported to the Lassonde Student Service for a ***formal investigation*** immediately.

- To protect yourself from ending up a submission that is suspiciously similar to someone else's, you want to ***avoid***:

  - Discussing code-level details about labs/assignments/project with anyone.
  - Discussing concrete steps about your solution or someone's solution.
  - Sharing any part(s) of your code (e.g., file transfer via email, discord channels, SMS, screen sharing via Zoom) at any stage of your development.
  - Giving or receiving instructions about what exactly you should type for a fragment of code.

    (e.g., it is *acceptable* to ask about how to write a loop in general, but *unacceptable* to ask about how to write a loop specifically for solving a problem related to the assignment).

- The best ways to help your fellow students are clarifying instructions and showing them how to use breakpoints/debugger.

Written Tests & Programming Tests

- All written and programming tests, as well as the final exam, are to be completed ***individually***: no group work is allowed.

  TAs will perform thorough checks on **all** programming test submissions: convincingly suspicious submissions will be reported to the Lassonde Student Service for a ***formal investigation*** immediately.

- It is considered ***a breach of academic honesty*** if:

  - You collaborate with someone on completing a written or programming test during any stage of your development.
  - After you have attempted the written or programming test and before that test is closed, share your test questions with someone.

Reporting Cases

Enforcing the policy of academic honesty not only maintains the ***standard*** of the course, but also ensures ***fairness*** among all students in the class. If you have sufficient reasons to believe that cases of violation are present, let the instructor know and confidentiality will be maintained.

3 Instructors

– Chen-Wei (Jackie) Wang [Section B & Section E]

- Contact: jackie@eecs.yorku.ca (http://www.eecs.yorku.ca/~jackie/)
- Virtual Office: `https://yorku.zoom.us/my/jackie.loves.oxford`
- Office Hours: 14:00 – 15:00 (EST), Mon, Tue, Thu; or by Appointments.

4 eClass Site

– There is a single eClass site shared by Sections N & Z:

`https://eclass.yorku.ca/course/view.php?id=62334`

5 Study Materials

– There will be no textbooks for this course. Study your instructor's lecture materials:

- The lectures page:
  `https://www.eecs.yorku.ca/~jackie/teaching/lectures/index.html#EECS2011_W22`

– For a thorough review on OOP in Java, consider the study materials for EECS2030-F21:
`https://www.eecs.yorku.ca/~jackie/teaching/lectures/index.html#EECS2030_F21`

– Here are some optional reference books:

- Data Structures and Algorithms in Java, 6th Edition (2014), Wiley
  Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser
- Algorithms, 4th Edition (2011), Addison-Wesley Professional
  Robert Sedgewick, Kevin Wayne [`https://algs4.cs.princeton.edu/home/`]

6 Available Help Resources

– Your instructor's office hours

– TA office hours

– Weekly Q&A sessions (held by the instrutor)

7 Prerequisites

– **General Prerequisites**: A cumulative grade point average (GPA) of 4.50 or better over all previously completed Major EECS courses. The GPA computation excludes all EECS courses that have a second digit 5, or are Co-Op/PEP courses.

– LE/EECS1030 3.00 or LE/EECS2030 3.00

– LE/EECS1028 3.00 or SC/MATH1028 3.00 or LE/EECS1019 3.00 or SC/MATH1019 3.00

8 Course Description

This course discusses the fundamental data structures commonly used in the design of algorithms. At the end of this course, students will know the classical data structures, and master the use of abstraction, specification and program construction using modules. Furthermore, students will be able to apply these skills effectively in the design and implementation of algorithms.

Abstract operations on data structures are specified using pre and post conditions and/or system invariants. Trade-offs between a number of different implementations of each abstract data types (ADT) are analyzed.

Each algorithm operating on data structures is proved correct using loop invariants or induction. Both formal and informal proofs are introduced though most of the reasoning is done informally.

Data structures are coded and unit tested in an object-oriented language. Selecting the appropriate ADT and a suitable implementation depending on the application is covered.

9 Course Learning Outcomes (CLOs)

Upon completion of the course, students are expected to be able to:

CLO1 Instantiate a range of standard abstract data types (ADT) as data structures.

CLO2 Implement these data structures and associated operations and check that they satisfy the properties of the ADT.

CLO3 Apply best practice software engineering principles in the design of new data structures.

CLO4 Demonstrate the ability to reason about data structures using contracts, assertions, and invariants.

CLO5 Analyse the asymptotic run times of standard operations for a broad range of common data structures.

CLO6 Select the most appropriate data structures for novel applications.

10 GRADING SCHEME

		SUBTOTAL
3 Assignments (5% each)	15%	35%
2 Programming Tests (10% each)	20%	
2 Written Tests (10% each)	20%	65%
Exam (Cumulative)	45%	

11 SECTION N VS. SECTION Z

– Assignments, programming & written tests, and exam are common to both sections (N & Z).

Instructions will be posted on the common (N & Z) eClass site.

* ∗ Assignments will be submitted via the web submit.
* ∗ Programming & written tests and exam will be submitted via eClass.

12 EXPECTED WEEKLY WORKLOAD

– Lassonde's recommendation is 3 – 4.5 hours per credit: *9 – 13.5 hours* for a 3.00 course.

– "In-Class" Hours:

- Lecture Videos [≈ 3 hours]

  Optional: Q&A sessions, Office Hours

– "Out-of-Class" Hours:

- Completing Assignments, Studying for Lectures/Tests [6 to 10.5 hours]

– Given that this is a ***fundamental course***, it is **not unreasonable** that you find yourself needing more time to digest the materials and build the skills.

The harder you work in this course, the easier you may find in subsequent years.

13 Mapping Raw Marks to Letter Grades

According to the Common Grading Scheme for Undergraduate Faculties approved by Senate:

Letter Grade	Grade Point	Interpretation
A+	9	Exceptional
A	8	Excellent
B+	7	Very Good
B	6	Good
C+	5	Competent
C	4	Fairly Competent
D+	3	Passing
D	2	Marginally Passing
E	1	Marginally Failing
F	0	Failing

- For each grading unit, you will receive a **raw mark score** (not necessarily out of 100).
- The **weighted sum** of all grading units will be mapped to its letter grade.

  e.g., Say there are only two grading units: Exam (60%) and Midterm (40%).

  A student receiving 150 marks (out of 200) for Exam and 2 marks (out of 3) for Midterm has:

  Weighted sum: $\frac{150}{200} \times 60 + \frac{2}{3} \times 40 \approx 71.7$

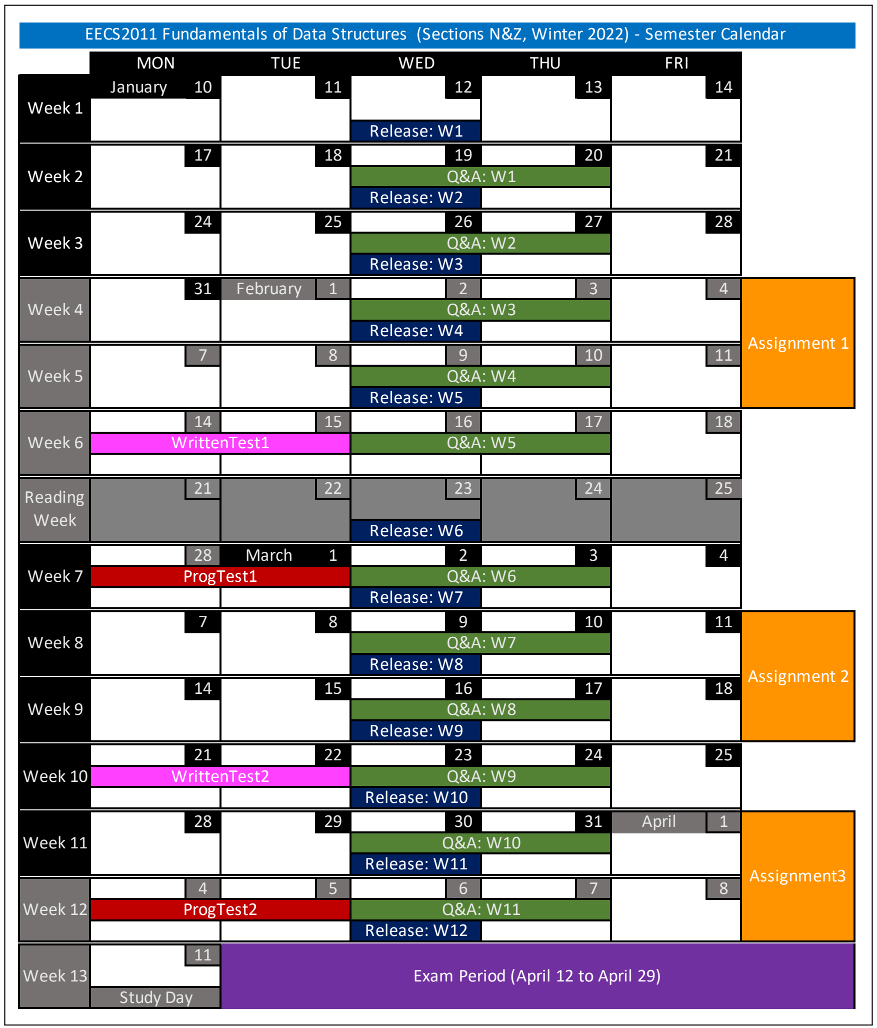
  Letter grade: B

14 Semester Calendar

– Figure 1 summarizes the schedule of required work items:

- Pre-recorded lectures are released on Wednesdays.
- Optional Q&A sessions (for discussing example programming interview problems and questions related to lectures) are held during the scheduled class times:

  Wednesdays (16:00 to 17:30, EST) and Thursdays (8:30 to 10:00, EST).
- Assignments are released on Fridays.
- Each written test lasts for **30 minutes** and takes place:
  - ∗ either between 16:00 and 16:30 on the corresponding Monday,
  - ∗ or between 8:30 and 9:00 on the corresponding Tuesday.
- Each programming test lasts for **90 minutes** and takes place:
  - ∗ either between 16:00 and 17:30 on the corresponding Monday,
  - ∗ or between 8:30 and 10:00 on the corresponding Tuesday.



EECS2011 Fundamentals of Data Structures (Sections N&Z, Winter 2022) - Semester Calendar

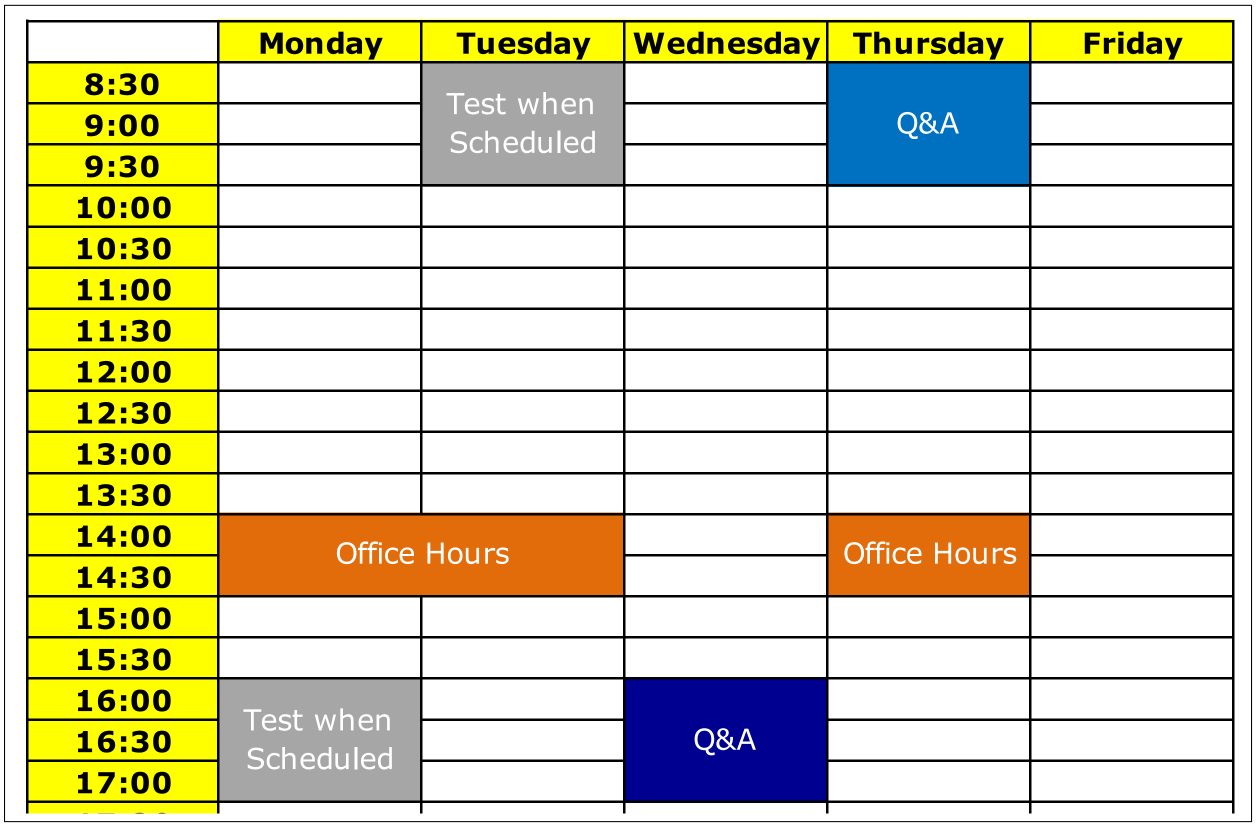
	MON	TUE	WED	THU	FRI	
Week 1	January 10	11	12 Release: W1	13	14	
Week 2	17	18	19 Q&A: W1 Release: W2	20	21	
Week 3	24	25	26 Q&A: W2 Release: W3	27	28	
Week 4	31	February 1	2 Q&A: W3 Release: W4	3	4	Assignment 1
Week 5	7	8	9 Q&A: W4 Release: W5	10	11	
Week 6	14 WrittenTest1	15	16 Q&A: W5	17	18	
Reading Week	21	22	23 Release: W6	24	25	
Week 7	28 ProgTest1	March 1	2 Q&A: W6 Release: W7	3	4	
Week 8	7	8	9 Q&A: W7 Release: W8	10	11	Assignment 2
Week 9	14	15	16 Q&A: W8 Release: W9	17	18	
Week 10	21 WrittenTest2	22	23 Q&A: W9 Release: W10	24	25	
Week 11	28	29	30 Q&A: W10 Release: W11	31	April 1	Assignment3
Week 12	4 ProgTest2	5	6 Q&A: W11 Release: W12	7	8	
Week 13	11 Study Day	Exam Period (April 12 to April 29)				

Figure 1: EECS2011-N&Z W22 Semester Calendar – Expected Work Items

15 Q&A SESSIONS

– In the time table below, each cell denotes a 30-minutes interval. For examples:

- Cell 8:30 denotes the interval starting at 8:30 and ending at 10:00.
- The fact that the Q&A session on Wednesday occupies 3 cells indicates that it lasts for 1.5 hours (starting at 16:00 and ending at 17:30).



	Monday	Tuesday	Wednesday	Thursday	Friday
8:30		Test when Scheduled		Q&A	
9:00					
9:30					
10:00					
10:30					
11:00					
11:30					
12:00					
12:30					
13:00					
13:30					
14:00	Office Hours			Office Hours	
14:30					
15:00					
15:30					
16:00	Test when Scheduled		Q&A		
16:30					
17:00					

– For Section N and Section Z:

- Lecture time slots on Mondays and Tuesdays are used to host written tests and programming tests. See the test schedule on Figure 1 (p8).
- Lecture time slots
  * <u>**16:00 – 17:30 on Wednesdays**</u>
  * <u>**08:30 – 10:00 on Thursdays**</u>

  are used to hold (optional, Zoom) Q&A sessions to answer your questions related to the lecture materials, and to discuss example interview-type problems.
- You are welcome to attend <u>**any**</u> of them to ask questions related to lectures.

Remark. For both Q&A sessions, it is <u>completely your decision</u> on attending one, more, or none of them. However, I would <u>**not**</u> advise that you skip all of them, unless you are absolutely confident with the course materials.

16 (Tentative) Weekly Lecture Topics

Lecture videos are being actively recorded, so the order of topics below are subject to changes.

Week	Topics
1	• Asymptotic Analysis of Algorithms • Contracts (Pre- and Post-Conditions), Assertions • (Iterative) Sorting: Selection, Insertion
2, 3	• Linked Lists: Singly-Linked vs. Double-Linked
4	• Abstract Data Types (ADTs) • Stacks, Queues, Deque • Maps, Hash Table
5, 6	• Recursion, Tail-Recursion, Backtracking • Searching (Binary Search) and Sorting (MergeSort, QuickSort) • Running Time, Correctness via Induction
Reading Week	
7	• General Trees, Terminology, Tree Traversals, Binary Trees (BTs)
8	• Binary Search Trees (BSTs) • Balanced BSTs, AVL Trees
9	• Heap, HeapSort • ADT: Priority Queue
10	• Graphs, Terminology • Implementing Graphs: edge list, adjacency list, adjacency matrix
11	• Graph Traversals (Depth-First vs. Breadth-First) • Directed Acyclic Graphs (DAGs), Topological Sort
12	• Shortest Path (Dijkstra's Algorithm) • Minimum Spanning Tree • Wrap-Up