McGill

Faculty of Science

COMP 250 - Introduction to Computer Science

Course Outline

McGill University, Winter 2026

Instructor:

Giulia Alberini (she/her)
Office: McConnell Engineering (MC) 233
Email: giulia.alberini@mcgill.ca
Office Hours: TBD

1 Land Acknowledgment

McGill University is on land which has long served as a site of meeting and exchange amongst Indigenous peoples, including the Haudenosaunee and Anishinabeg nations. We acknowledge and thank the diverse Indigenous peoples whose presence marks this territory on which peoples of the world now gather.

L'Université McGill est sur un emplacement qui a longtemps servi de lieu de rencontre et d'échange entre les peuples autochtones, y compris les nations Haudenosaunee et Anishinabeg. McGill honore, reconnaît et respecte ces nations à titre d'intendant traditionnel des terres et de l'eau sur lesquelles nous réunissions aujourd'hui.

Please see here for more details: `https://www.mcgill.ca/fph/welcome/traditional-territory`.

2 Course Overview

Welcome to COMP 250. This course is designed for students with some programming experience who are looking to deepen their understanding of computer science.

COMP 250 is a course that will challenge you. It covers a broad range of topics, from learning a new programming language to tackling more complex concepts like recursion. We understand that this material can be demanding, and our goal is to support you as you navigate these challenges. We're committed to creating a learning environment where everyone feels respected, supported, and focused on growth rather than just performance.

We invite you to engage in online and in-class discussions with curiosity and an open mind. Your active participation will not only help you learn but will also contribute to a richer learning experience for everyone.

This one-semester, 3-credit course, has two parts. The first part of the course introduces object-oriented programming using Java. You'll learn the fundamentals of Java programming, including principles of object-oriented design. We'll cover creating and working with Java objects and classes and organizing these classes into hierarchies. This foundation will prepare you for later courses, such as COMP 303 - Software Design.

In the second part, we'll explore essential topics in computer science: data structures and algorithms. You'll learn about basic data structures like lists (arrays, linked lists, stacks, queues), trees (search trees, heaps), and graphs, as well as fundamental algorithms for working with these structures. The assignments will provide hands-on experience with these concepts in Java, and you'll learn to analyze algorithms based on their computational efficiency, a skill that will be critical in future courses like COMP 251 - Algorithms and Data Structures.

Please note that while this course introduces the Java programming language, it is not an introductory programming course. We expect you to be familiar with a high-level programming language like Python or Java, and to have experience with programming larger assignments. If you haven't used Java before, expect an adjustment period in Weeks 2–4; that's normal and planned for.

The following sections provide detailed information on course policies and assessment methods.

2.1 Learning Outcomes

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

1. Demonstrate proficiency in Java syntax, including classes, methods, control structures, and data types.
2. Apply core Object-Oriented Programming principles such as encapsulation, inheritance, and polymorphism in software design.
3. Construct, manipulate, and critically evaluate essential data structures, including ArrayLists, linked lists, and trees, understanding their strengths and limitations in different scenarios.
4. Implement standard algorithms to sort or traverse data structures and develop simple variants using both iterative and recursive approaches.
5. Apply recursion effectively in problem-solving and in the design and manipulation of data structures.
6. Analyze the computational efficiency of algorithms using asymptotic notation, and recurrences.
7. Understand and work with more complex structures like heaps, hash maps, and graphs.
8. Reflect critically on your understanding of course material to assess your preparedness for tackling coding challenges, evaluating your level of knowledge and identifying areas for improvement (metacognition).

2.2 Prerequisites

As per the e-calendar the official prerequisites are MATH 140 (or equivalent), and COMP 202/204/208 (or equivalent). The co-requisite is MATH 133. Please note that, COMP 250 is not open to students who have taken or are taking ECSE 250, and cannot be taken at the same time as COMP 202/204/208.

3 Course Content and Material

All the material needed for this class will be available on myCourses; this includes lecture slides, lecture recordings, recommended exercises, as well as links to any other material that might help you reinforce and broaden your understanding of the course material.

There is no required textbook.

3.1 Required Software

We use **Java 21 (LTS)**. The Java compiler (`javac`) and the Java Virtual Machine (JVM) are included in the Java Development Kit (JDK).

You can use any **plain-text editor** to write code and compile/run with the JDK tools. For this course, we recommend *Eclipse* or *IntelliJ IDEA Community*; both provide features like code navigation and a built-in debugger. The teaching staff will support both.

You are encouraged to install the JDK and an IDE on your own machine. If you need help, see the tutorials on myCourses or come to office hours. We will also run a setup tutorial session early in the semester.

Summary

- **Required: JDK 21 (LTS)**. We recommend an OpenJDK distribution such as *Temurin (Adoptium). Install the JDK before any IDE.*
- **Recommended:** An IDE:
  - *IntelliJ IDEA Community* (free) or
  - *Eclipse IDE for Java Developers.*

After installation (quick checks)

- Open a terminal and verify:
  - `java -version` and `javac -version` both report version 21.
- In your IDE, set the project SDK / language level to **Java 21**.

For a step-by-step guide, see the tutorials share on myCourses under *Content* → *Resources* → *Tutorials* → *Software Installations.*

3.2 Copyright policy

You are not allowed to post any course materials on github, coursehero, any other websites. This includes PDFs of lecture slides, lecture notes, exercises, quizzes, assignment questions or anything else that we provide for you.

Stated more formally: "Instructor-generated course materials are protected by law and may not be copied or distributed in any form or in any medium without explicit permission of the instructor(s). Note that infringements of copyright can be subject to follow up by the University under the Code of Student Conduct and Disciplinary Procedures."

3.3 Tentative Lecture Schedule

Week	Date	Topic
1	Jan 6	Introduction
	Jan 8	Java Fundamentals 1 - general syntax, primitive data types, type conversion
2	Jan 13	Java Fundamentals 2 - Strings, Arrays, and reference types
	Jan 15	OOP 1 - Packages, fields, and modifiers
3	Jan 20	OOP 2 - Constructors, Getters/Setters, toString()
	Jan 22	OOP 3 - Mutable vs Immutable, and Inheritance
4	Jan 27	OOP 4 - Type Conversion, Polymorphism, and Abstract classes
	Jan 29	ArrayLists
5	Feb 3	Linked Lists
	Feb 5	Quadratic Sorting, and Asymptotic Notations
6	Feb 10	Case Study #1
	Feb 12	Stacks
7	Feb 17	Queues
	Feb 19	OOP 5 - Interfaces, Comparable
8	Feb 24	OOP 6 - Iterable/Iterator, case study #2
	Feb 26	Induction
9	Mar 3	**Reading week: no classes**
	Mar 5	
10	Mar 10	Recursion 1
	Mar 12	Recursion 2
11	Mar 17	Recursion 3
	Mar 19	Trees: definitions, implementations, and traversals
12	Mar 24	Binary Trees, and Binary Search Trees
	Mar 26	REVIEW
13	Mar 31	Heaps, case study # 3
	Apr 2	Maps, and hashing
14	Apr 7	Graphs 1
	Apr 9	Graphs 2

Tested on: A1 A2 A3 Final Assessment

4 Communication Policies

The University is committed to maintaining teaching and learning spaces that are respectful and inclusive for all. To this end, offensive, violent, or harmful language arising in course contexts may be cause for disciplinary action under the Article 10 of the Code of Student Conduct and Disciplinary Procedures and Section 2.7 of the Policy on Harassment, Sexual Harassment, and Discrimination Prohibited by Law.

4.1 Office Hours

Teaching Assistants (TAs), TEAM Mentors and instructor will be available for office hours to support your learning throughout the semester.

A link to a Google calendar with everyone's office hours will be shared with you on myCourses.

4.2 Discussion board

We will be using Ed Discussion as the official discussion board. The system is highly catered to getting you help fast and efficiently from classmates, the TAs/Mentors, and the instructor. A link to COMP 250's Ed page will be shared on myCourses.

Rather than emailing questions to the teaching staff, **we encourage you to post all your questions related to the course content and the assessments on Ed**. If your question does not reveal sensitive information (e.g., your answers or progress on an assignment question, or a personal situation), *please* make your post public so that other students can also benefit from the answer.

Discussion Board Guidelines

Please follow common sense rules and etiquette for discussion board postings: be polite and respectful, avoid texting shorthand, choose a suitable subject line for your posting, use multiple postings for multiple subjects, proof read before posting, and keep your posts brief if possible.

You may freely answer other students' questions as well, with one important exception: do not post your own code or answers to assignments publicly on Ed.

4.3 Contacting Instructor and Teaching Assistants

For private matters only, you e-mail the instructor directly. Be sure to send your email from your `@mail.mcgill.ca` address and include your student ID. When emailing, please follow the guidelines on etiquette described in the video here.

You should **not** reach out to the TAs using their personal emails, unless otherwise instructed. You can instead:

- Post your question on the Ed Discussion Board.
- OR, meet with them during their office hours.

Note that, especially if your questions require you to share part of your own work, we highly encourage you to take the time to go and see someone from the teaching staff during their office hours.

4.4 Course Announcements

Important information about the course will be announced in class and on Ed Discussion.

Students are expected to monitor both their McGill e-mail account, myCourses, and Ed Discussion for course-related news and information.

5 Means of Assessment

Traditional grading practices have been criticized for being biased and inequitable, often failing to account for students' diverse backgrounds, experiences, and learning needs [Bro11; Cai+22; Har03; Fel23]. These practices can reduce achievement, increase stress, and discourage students from engaging fully with their learning [Dec+24]. Grades that reward or punish behavior rather than learning do not encourage genuine understanding and instead contribute to anxiety and stress.

This semester, COMP 250 will use a grading scheme inspired by competency- and equity-based approaches, moving away from traditional methods. Competency-based grading measures your mastery of specific skills and knowledge, while equity-based grading considers the individual circumstances and systemic factors that may affect your ability to succeed [Ger16; Fel19; Fel23]. Our grading system also incorporates principles of Universal Design for Learning (UDL), which ensures that educational materials, methods, and assessments are accessible to all learners from the outset, rather than retrofitting accommodations later [Bur09].

Our goal is to provide a grading system that:

- Directly ties evaluations to specific learning outcomes. You do not need to achieve perfection to demonstrate mastery.
- Offers flexibility in demonstrating mastery at your own pace and through your preferred methods.
- Provides opportunities to receive timely and personalized feedback and support.
- Encourages learning from mistakes, redoing assignments, and overcoming challenges.
- Allows you to view assignments and exams as opportunities for growth, even in the face of setbacks.

Throughout the course, different assessment methods are used to assess the level of mastery achieved for different topics and competencies. To demonstrate your understanding of the course content, you will be able to choose between the two following schemes.

Assessment Method	Codecrafter *(default)*	Problem Solver
Programming Assignments (3)	✓	✓
Midterms (2)	✓	✓
Final Project	✓	
Technical Interview		✓
Mini-Interviews (0-3)		(✓)
Code Review	(✓)	

Codecrafter vs Problem Solver - making a choice

You will have the opportunity to choose your preferred grading scheme by completing a Survey on myCourses. Please note that if you do not fill out this survey by the provided deadline (**Jan 31**), then you'll be assigned to the default grading scheme (*Codecrafter*).

Competency Level and Final Letter Grade

Whenever you submit work for an assessment, you will receive feedback on the competency level you have achieved. Personalized feedback and support will be provided whenever possible. Inspired by [Fin24], the five competency levels are:

Mastery: You demonstrate a thorough and detailed understanding of the material, and are able to combine multiple concepts to solve entirely new problems.

Approaching Mastery: You have a strong understanding of the material and can apply it to slightly more challenging or unfamiliar contexts not explicitly covered in class.

Proficiency: You understand all course material and can apply it within familiar contexts. Simply knowing the definitions or being able to implement methods already seen in class is an example of a student at this level.

Basic: Your understanding of the material is limited or incomplete.

Inconclusive: You have not demonstrated adequate understanding, or your performance is insufficient for assessment.

At the end of the semester, the university requires us to assign a letter grade representing your performance. The following table outlines how to map the competency levels you have achieved to a corresponding letter grade:

Final Letter Grade	Mapping
A	At least 5 Mastery levels, with all other competencies at Approaching Mastery or higher.
B	No more than 2 Proficiency levels, or 1 Proficiency + 1 Basic level, compensated by an equal number of Mastery levels. All other competencies at Approaching Mastery or higher.
C	No more than 2 Basic levels, or 1 Basic + 1 Inconclusive level, compensated by 1 Approaching Mastery and 1 Mastery level respectively. All other competencies at Proficiency or higher.
D	Otherwise.
F	Two or more Inconclusive levels.

How to collect competency levels? Each programming assignment and each midterm counts for one competency, while the final assessment (final project or technical interview) counts for two. The midterm in which you achieve the highest level of mastery will count for double. Overall, you will be assigned eight competency levels throughout the semester.

How to Unlock +/-'s

To reward nuanced achievements and better reflect the competency-based grading philosophy, +/- distinctions will now be applied to final letter grades. Here is how:

- **A-**: If your base mapping yields a B, you'll receive A- if:
  - At least 2 of the 5 competency levels from midterms and the final assessment are Mastery, and the remaining are Approaching Mastery.
- **B+**: If your base mapping yields a B, you'll receive B+ if:
  - The competency level achieved in both midterms is Approaching Mastery or higher, OR
  - All competency levels from assessments other than midterms are Mastery.
- **B-**: If your base mapping yields a C, you'll receive B- if:
  - At least 2 of the 5 competency levels from midterms and the final assessment are Approaching Mastery or higher, and the remaining are Proficiency.
- **C+**: If your base mapping yields a C, you'll receive C+ if:
  - The competency level achieved in both midterms is Proficiency or higher, OR
  - All competency levels from assessments other than midterms are at least Approaching Mastery.

Let's look at some examples:

Competency Levels Achieved	Final Letter Grade
Midterms: AM, AM; Assignments: M, M, M; Final Assessment: M	A
Midterms: AM, M; Assignments: P, M, AM; Final Assessment: AM	A-
Midterms: P, AM; Assignments: M, M, M; Final Assessment: M	B+
Midterms: P, AM; Assignments: P, M, M; Final Assessment: AM	B
Midterms: P, P; Assignments: B, AM, AM; Final Assessment: AM	B-
Midterms: P, P; Assignments: M, AM, P; Final Assessment: P	C+
Midterms: I, P; Assignments: M, AM, B; Final Assessment: P	C
Midterms: B, P; Assignments: M, AM, P; Final Assessment: B	D
Midterms: I, I; Assignments: M, M, M; Final Assessment: P	F

Check out our website www.comp250.com where you can find a grade calculator that can help you further understand how the final letter is computed.

Competency Tokens: Understanding Your Assessment

To evaluate each assessment, we assign tokens to individual questions or parts of the assignment. These tokens represent whether or not a certain level of understanding has been reached. Tokens are then counted at the end of the assessment based on the level of competency they target. The number of tokens you earn determines the competency level assigned for that assessment.

The specific formulas used to calculate competency levels will be shared with students for each assessment. The primary goal of this approach is to evaluate whether a certain level of understanding has been achieved rather than focusing on perfection.

Key Points to Remember:

- **Different from Points and Percentages:** Unlike traditional percentage-based grading schemes, tokens are not analogous to points. Points in percentage-based grading measure total performance, while tokens explicitly assess specific competencies.
- **Focus on Competency:** The allocation of tokens is designed to reflect whether or not you have met the learning objectives, rather than measuring your performance relative to a perfect score.
- **Misleading as Percentages:** It is important to note that looking at tokens from a percentage perspective is misleading and may not accurately represent your demonstrated understanding of the material.
- **Fair and Transparent:** This token-based system is tailored to provide fair and meaningful evaluations that are closely tied to the course learning outcomes.

Official language policy for graded work

In accordance with McGill University's Charter of Students' Rights, students in this course have the right to submit in English or in French any written work that is to be graded. See here for more details: `https://www.mcgill.ca/study/2019-2020/university_regulations_and_resources/undergraduate/gi_lang_policy`.

Conformément à la Charte des droits de l'étudiant de l'Université McGill, chaque étudiant a le droit de soumettre en français ou en anglais tout travail écrit devant être noté, sauf dans le cas des cours dont l'un des objets est la maîtrise d'une langue. (Énoncé approuvé par le Sénat le 21 janvier 2009)

5.1 Assignments

There will be **three** assignments consisting of writing Java programs. It is important to complete all assignments, as they are the best way to learn the material. By working hard on the assignments, you

will gain essential experience needed to solve programming problems. Your work will be evaluated for both correctness and efficiency (starting from Assignment 2) and will be graded based solely on the results of automated tests. Assignments will primarily focus on testing the following skills:

1. Your proficiency in Java, including your understanding of fundamental OOP principles.
2. Your ability to implement, and manipulate data structures learned in class.
3. Your capacity to adapt the algorithms seen in class and to analyze their efficiency.

Assignments are meant to be challenging and may involve creative thinking. Therefore, it is recommended to read the questions early to allow yourself time to implement, test, and debug your solutions.

Timeline (tentative)

	Released	**Suggested Deadline**	**Topic**
Assignment 1	Jan 27	Feb 10	Java Syntax and OOP
Assignment 2	Feb 20	Mar 13	Lists and Interfaces
Assignment 3	Mar 20	Apr 3	Recursion and Trees

Assignments (and all other course work) **MUST** represent your own personal efforts (see the section on Plagiarism Policy and Assignments below).

If you do not do submit any work for an assignment, then you will receive an Inconclusive for it.

Mastery Requirement and Suggested Deadlines

To achieve **Mastery** on an assignment, you must submit a version that earns at least **Proficiency** by the suggested deadline. This requirement is designed to encourage students to engage with the assignments early and keep up with the material presented in class. Attempting the assignments on time also provides valuable preparation for the midterms, as the assignments are a great way to practice and reinforce your understanding.

Key Points:

- **Suggested Deadlines:** Assignments will be posted with suggested deadlines that are designed to be reasonable under standard circumstances. Submitting by the suggested deadline ensures that you can aim for Mastery, by resubmitting later.
- **Final Submission Deadline:** You can submit assignments until April 16th. The competency level achieved in your most recent submission will be considered for your final grade. However, if you achieve Mastery but did not submit a version that reached at least Proficiency by the suggested deadline, your Mastery will be capped at **Approaching Mastery**.
- **Office Hours and Support:** Questions during office hours and on the discussion board will be prioritized based on upcoming deadlines.

This structure allows you to continuously improve your understanding and mastery through repeated attempts and feedback. Start early, and take advantage of the available resources to maximize your learning and performance.

5.2 Midterms

You will be required to complete **two in-person**, *closed book* midterm examinations. The midterms are scheduled for **February 18** and **March 25**, from 6pm to 8pm.

These exams will include a variety of question types, such as multiple choice, short answer, and long answer, all designed to assess your understanding of the material covered in class.

Importantly, the midterm in which you receive the highest level of mastery **will count for double** toward the final letter grade. Note that this effectively means that if you receive an Inconclusive level in *both* midterms, you will receive an F as your final course grade. The grading scheme is designed as such for the following pedagogical reasons:

- Midterms assess your mastery of core concepts and skills, which are foundational for more advanced topics in the course and program.
- They may be the only proctored assessments in this course, ensuring the integrity and fairness of the grading process.
- In-person midterms allow you to demonstrate knowledge under controlled conditions, proving your ability to apply what you've learned without external assistance.

Optional Third Midterm: To provide an opportunity for improvement, an optional exam is scheduled for **April 8** from **6pm to 8pm**. This exam will cover the material of either Midterm 1 or Midterm 2, based on your selection, and allows you to replace the competency level achieved in one of those exams. The competency level achieved in the third midterm will **fully replace** the one of the selected midterm, even if it is lower. Therefore, you should only take this optional midterm if you believe your understanding of the material has improved. This third midterm is designed to encourage reflection on your learning progress and provide an additional opportunity to demonstrate your knowledge.

Missed midterms (valid reasons). If you miss Midterm 1 or 2 for a *valid reason* (e.g., illness, religious observance, exam conflict, emergency), the April 8 exam serves as your **replacement** for that midterm. If you also cannot attend April 8 for a valid reason, please contact us **as soon as possible**.

If you missed one of the two regular midterms for a valid reason and would therefore have only *two* total opportunities while the rest of the class has three, you may contact us to discuss an **equitable alternative** so you have the same number of opportunities as everyone else. Unexcused absences result in **Inconclusive**.

Notes: We recognize that not all circumstances are documentable; a brief explanation is sufficient. In rare cases we may ask for supporting information to ensure fairness across the class.

Religious observance: If a midterm conflicts with a holy day, please reach out **at least two weeks in advance**.

5.3 Final Assessment

Depending on your grading scheme selection, you will be required to complete one of the following final assessments:

1. A **final coding project** with an **in-person code review** (required for achieving Mastery), or
2. A **live technical interview** examination, which may include optional mini-interviews.

In both cases, the level of mastery you achieve will count for double toward your final letter grade.

Codecrafters

Codecrafters will submit a final coding project, which is a large programming assignment in Java that tests your ability to implement complex data structures and write efficient algorithms. Your algorithms will be assessed for both correctness and efficiency.

- **General Deadline:** All projects must be submitted by **April 30th**.
- **Early Submission Benefits:** To achieve a **Mastery** level in the final project:
  - Submit by **April 25, 23:59 ET** and receive Mastery from the preliminary evaluation through automated tests.
  - Schedule and participate in a 15-minute **in-person code review** with a TA between April 28th and April 30th.
- **Code Review:** The review evaluates your understanding of the submitted code. It will confirm or adjust your Mastery level based on the depth of understanding demonstrated.
- **Important:** If you do not participate in a code review, or if your submission is after after **April 25, 23:59 ET** (and by **April 30, 23:59 ET**), the maximum level you can achieve for the final project is **Approaching Mastery**. If Mastery is not your goal, you are not required to participate in the code review.

Problem Solvers

Problem Solvers will complete a final technical interview designed to simulate the experience of a coding interview.

- **Format:** You will solve and present solutions to three problems within a 45-minute in-person session. These problems will be selected from a pre-shared list of 45-50, and you may choose one of the three problems yourself.
- **Interactive Component:** There will be follow-up questions on your solutions to assess your problem-solving approach and depth of understanding.
- **Scheduling:** Interviews will be scheduled during the final exam period based on the availability of both students and teaching staff. Further details will be provided in class and via myCourses/Ed Discussion.
- **Note:** If you complete all three mini-interviews (see below), the final technical interview will only count for **one competency level** instead of two if this adjustment benefits your overall grade. This change is designed to reduce the stress associated with the final technical interview.

5.4 Mini Interviews

If you select the *Problem Solver* scheme, you have the option to participate in mini-interviews, which are designed to help you practice for the final technical interview and receive early feedback from teaching staff.

- **Format:** Each mini-interview is a 15-minute in-person session where you will present one of the problems (of your choice) provided at the start of the semester, corresponding to the current assignment topic.
- **Scheduling:** You must sign up for your presentation slot at least one week before the relevant assignment deadline.

- **Optional Nature:** Participation in mini-interviews is optional. You may choose to complete none, one, two, or all three.
- **Benefits:**
  - Mini-interviews provide structured practice to build confidence and reduce stress related to oral assessments.
  - **If you complete all three mini-interviews:** the final technical interview counts for **1 competency** and the **average** of your three mini-interviews counts for **1 competency**. If this substitution hurts your grade, we keep the default (final = 2, minis formative).

5.5 Regrade Requests

If you believe there has been an error in the grading of any of your assessments (assignments, midterms, or interviews), you are entitled to request a regrade. **Regrade requests must be submitted within 5 working days from the date you received the level of mastery achieved**. Requests submitted after this deadline will not be considered.

To ensure fairness and accuracy, regrade requests may lead to a complete re-evaluation of the work in question. This means that not only the part you contest but also other sections of your exam or assignment might be re-graded. As a result, the final level of mastery assigned could be higher, lower, or the same as the original.

5.6 Supplemental/Deferred Exam

There will be no supplemental or deferred exam for this course as there is no final exam.

5.7 Additional Work

Students who receive unsatisfactory final grades will **NOT** have the option to submit additional work in order to improve their grades.

5.8 Extraordinary Circumstances beyond the University's Control

In the event of extraordinary circumstances beyond the University's control, the evaluation scheme in a Course is subject to change, provided that there be timely communications to the students regarding the change. See section 3.2.3 of the *University Student Assessment Policy*.

6 Policies on Academic Integrity

Official policy: " *McGill University values academic integrity. Therefore all students must understand the meaning and consequences of cheating, plagiarism, and other academic offenses under the Code of Student Conduct and Disciplinary Procedures (see* `www.mcgill.ca/integrity/` *for more information)* ".

6.1 Plagiarism Policy

You must include your name and McGill ID number at the top of all submitted work. By doing so, you certify that the work is entirely your own and solely the result of your individual effort.

Work submitted for this course must represent your own efforts. Assignments **must** be done **individually**; you **must not** work in groups. Do not rely on friends or tutors to do your work for you. You **must not** copy any other person's work in any manner (electronically or otherwise), even if this work is in the public domain or you have permission from its author to use it and/or modify it in your own work. Furthermore, you **must not** give a copy of your work to any other person.

The plagiarism policy is not meant to discourage interaction or discussion among students. We encourage you to discuss the assignment problems with each other. However, these discussions should be limited to collaborative problem solving: do not share code or give away answers. Do not share anything with friends that you would not share with the class broadly.

Importantly, we ask you to indicate on your assignments the names of the people with whom you collaborated or with whom you discussed assignment problems (including the TA's and the instructor). You will not be penalized for indicating collaborators, but failure to do so may result in your assignment being flagged or your grade being affected.

Text matching software

Submissions will be checked using text-matching software to detect similarities that suggest plagiarism. Cases with high levels of similarity will undergo a manual review.

You may also be asked to present and explain your assignment submissions to an instructor at any time.

When the instructor suspects that plagiarism has occurred, the instructor will report the case to the Disciplinary Officer in the student's Faculty (Science, Arts, Engineering, etc). For more details on the process, see Section III Articles A.37 (p. 10) and A.48 (p. 13) of the Code of Student Conduct and Disciplinary Procedures:

`https://www.mcgill.ca/deanofstudents/files/deanofstudents/code_of_conduct_revision_jan2019.pdf`

Posting assignment solutions on a website

We encourage you to use tools like GitHub for version control systems. However, you must **not** share your assignment solutions by posting them on a public space such as your GitHub account. If you do and if another student copies your solution from there, then there will be no way to discriminate who did the work, and you may be accused of plagiarism along with the other student(s).

This rule extends beyond the duration of the course.

6.2 Use of Generative AI

Students are encouraged to make use of technology, including generative artificial intelligence tools, to contribute to their understanding of course materials.

Students are not encouraged, unless otherwise stated, to make use of artificial intelligence tools, including generative AI, to help produce assignments. This includes tools such as ChatGPT or GitHub CodePilot. We believe that working through the assignments on your own will help you gain a better understanding of the course material and will better prepare you not only for the other course examinations, but also for the subsequent CS courses, internships, research opportunities, and jobs. However, students are ultimately accountable for the work they submit. Any content produced by an artificial intelligence tool must be cited appropriately. Many organizations that publish standard citation formats are now providing information on citing generative AI (e.g., MLA: `https://style.mla.org/citing-generative-ai`).

7 Post-Coronavirus-Pandemic Public Health issues

The Quebec government (`https://www.quebec.ca/en/health/health-issues/a-z/2019-coronavirus`) provides the following guidelines for educational institutions (as of August 2023). Wearing masks is not required, however, if you are experiencing cough, sore throat, or nasal congestion it is recommended that you wear a mask and, as much as possible, keep your distance from others and advise them you may be contagious. In the case of a fever, remain at home.

8 Accommodations

For this course, we are adopting flexible assessment strategies that create greater access for all students by incorporating principles of Universal Design for Learning. As such, we have taken into consideration the variety of learner needs and barriers that students may face in this course and have designed the assessments with these considerations in mind.

8.1 Student Accessibility and Achievement

Student Accessibility and Achievement helps McGill's diverse student body achieve their academic goals and overcome barriers by providing not only accommodations for students with documented disabilities but also additional learner support for students facing barriers in university.

There may be circumstances in which some disability-related accommodations may still be needed for this course. If you feel this is the case for you, please reach out to the SAA office at `access.achieve@mcgill.ca` or (514) 398-6009 (options #1-3). They will assess the situation and coordinate with the instructor when necessary.

8.2 Pregnancy and Caregiving

Students who are pregnant and/or caring for a dependent also often may find it helpful to receive academic accommodations. McGill's guidelines for accommodations for students who are pregnant and/or caring for a dependent may be found at `https://www.mcgill.ca/study/2018-2019/university_regulations_and_resources/graduate/gi_accommodation_pregnancy_caring_dependants`

References

[Har03] Charles H Hargis. *Grades and grading practices: Obstacles to improving education and to helping at-risk students*. Charles C Thomas Publisher, 2003.

[Bur09] Sheryl Burgstahler. "Universal Design of Instruction (UDI): Definition, Principles, Guidelines, and Examples." In: *Do-It* (2009).

[Bro11] Susan M Brookhart. "Starting the Con versation About Grading". In: *Educational Leadership. Retrieved from https://www. greatschoolspartnership. org/wp-content/uploads/2016/11/Starting-the-Conversation-about-Grading-2. pdf* (2011).

[Ger16] Jennifer Gervais. "The operational definition of competency-based education". In: *The Journal of Competency-Based Education* 1.2 (2016), pp. 98–106.

[Fel19] Joe Feldman. "Beyond standards-based grading: Why equity must be part of grading reform". In: *Phi Delta Kappan* 100.8 (2019), pp. 52–55.

[Cai+22] Jeff Cain et al. "Deficiencies of traditional grading systems and recommendations for the future". In: *American Journal of Pharmaceutical Education* 86.7 (2022), p. 8850.

[Fel23] Joe Feldman. *Grading for equity: What it is, why it matters, and how it can transform schools and classrooms*. Corwin Press, 2023.

[Dec+24] Adrienne Decker et al. "Transforming Grading Practices in the Computing Education Community". In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. 2024, pp. 276–282.

[Fin24] Benjamin T Fine. "Competency and Equity Driven Grading System for Computer Science Curriculum". In: *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*. 2024, pp. 311–317.