# Teaching Portfolio

## 1 Biographical Sketch

## 2 Teaching Philosophy

## 3 Teaching Responsibilities

- 3.1 Course Instruction
- 3.2 Guest Lecturing
- 3.3 Instructional Apprenticeships
- 3.4 Teaching Assistantships
- 3.5 Curriculum Development
- 3.6 Committee Service

## 4 Evidence of Teaching Effectiveness

- 4.1 Summary
- 4.2 Representative Feedback from Students
- 4.3 Representative Feedback from Colleagues
- 4.4 Honours and Awards

## 5 Professional Development

- 5.1 Professional Development in University Teaching and Learning Program, Queen’s University
- 5.2 Certificate in Professional Development, Queen’s University
- 5.3 Fundamentals of University Teaching Program, University of Waterloo

## A Course Descriptions

## B Course Evaluations

## C Sample Course Outline

Last updated:
May 4, 2020
1 Biographical Sketch

I am a PhD candidate in the School of Computing at Queen’s University. I am a member of the Formal Languages and Automata Theory group. I received my Bachelor of Science degree from the University of Western Ontario in 2015 and my Master of Mathematics degree from the University of Waterloo in 2017.

Generally, I teach and assist with courses on theoretical computer science covering topics such as formal language theory, automata theory, complexity theory, data structures, and algorithm analysis/design. I have also taught and assisted with courses on discrete mathematics and abstract mathematics. Appendix C contains a sample course outline of a recent course I have taught.

I am fully qualified to teach courses at the undergraduate level in any of the above topics. I also have considerable experience with formal logic and programming language theory, and I would be able to teach courses at the undergraduate level in those areas.

I am also willing to teach graduate-level courses relating to my area of research. Such courses may cover advanced topics in formal language theory, automata theory, or complexity theory.

2 Teaching Philosophy

...the best theory is inspired by practice and the best practice is inspired by theory.

— DONALD KNUTH

Theory and Practice (1991)

Teaching theoretical computer science presents a unique challenge: typically, at the level of instruction when most students are introduced to theory, all of their previous courses have been application-focused and, hence, directly applicable to what they expect to see when they enter the workplace. In a theoretical computer science course, then, where the concepts taught cannot be immediately connected to any real-world application, it is not unusual to hear the question “why do we care about this?” Indeed, having taught and assisted with courses that span the spectrum of theoretical computer science—algorithms, computability and complexity, data structures, discrete mathematics, formal languages and automata—this one question has been a universal constant. Students want to know why they are investing their time and energy into learning something that is too often presented in the abstract.

My attempt to guide students toward finding an answer to this question is best summarized by the above quote by Donald Knuth: I use practice to inspire theory and, in turn, I strive to have students’ future practices inspired by this theory. As hard as some theorists may work to keep their field “pure”, it is an undeniable truth that theory and practice are inextricably intertwined components of computer science. Rather than resisting this association, I propose that we embrace it: teach students from the very beginning about the deep and meaningful connections between theoretical computer science and other areas, so that they have a better understanding not only of theory, but of the field as a whole. My goal as an educator is to bring meaning to theory, and I make progress toward that goal by drawing three connections: between theory and application, between theory and the bigger picture, and between students and their education.

Connecting Theory to Application. One great pitfall of testing a student’s understanding of a theoretical subject lies in asking “so what?” questions. These questions are often designed to test knowledge of definitions or theorems; for instance, “Show that the function $42n^8 + 20n^6$ is $O(n^{10})$.” Answering questions in this vein provides an extrinsic motivation for, say, students preparing to write an exam. However, there is no intrinsic motivation for those same students to care about the solution they obtain.

I aim to design problems that require students to apply concepts taught in the classroom to a situation found in the real world. Often, I select applications I have not discussed in the classroom, leaving students to discover the link on their own and solidifying in their minds both the idea itself and how they might use the idea.
In terms of course design, I tend to allocate a higher percentage of marks to assignments and projects rather than to time-constrained assessments such as exams. I believe exams have their merits, but the fact that most exams are held at a point in the term when students no longer have an opportunity to improve their understanding of material severely limits the use of an exam as a learning tool. Placing a greater weight on assignments and projects encourages students to put more work into crafting a submission they can be proud of while, in the process, gaining a greater understanding of how the material they have learned can be applied to novel problems.

**Connecting Theory to the Bigger Picture.** When I teach concepts in theoretical computer science, I make an effort to connect the concept to its broader context. I often do so by outlining the history and development of the concept, or by highlighting a few areas of application where this concept may appear. For example, when I introduce students to Greibach normal form, I do not restrict the lesson simply to talking about context-free grammars. Rather, I introduce students to Sheila Greibach, how she developed her idea, and why compiler designers care about transforming grammars into the form bearing her name.

I include this broader context within my lectures not only to pique interest in otherwise dry topics, but also to imbue within students the sense that these topics exist outside of the realm of mathematics. Framing the introduction of a lesson through the lens of the broader arts and sciences—through biography, history, chemistry, or otherwise—both provides for a gentler introduction to an abstract concept and allows students to emerge from a course as more well-rounded academics.

Providing a historical perspective within lectures also comes with the added benefit of introducing students to the people behind the ideas, thus revealing that these ideas were not simply gifted unto us centuries ago by brilliant minds from atop the ivory tower. I feel that this approach encourages students to attempt their own discoveries, as the barrier to entry vanishes once students realize that the people who developed the very ideas being taught today were once just like them. This approach was influenced by texts such as Rosen’s *Discrete Mathematics and Its Applications*, which includes biographical sidebars of important mathematical figures throughout each chapter.

**Connecting Students to Their Education.** When I was an undergraduate student, the theoretical computer science course in which I was enrolled was taught through PowerPoint. While this may work for some subjects, I staunchly believe that PowerPoint (and, indeed, any static presentation format) has no place in theoretical computer science. At its core, theoretical computer science is applied mathematics, and so much like a course in mathematics, instructors should employ a fluid, discovery-based approach to teaching theory.

To promote interest in a course, I invite students to participate in the development and presentation of course material during each lecture, rather than flicking through slides and beaming contextless information at the class. I write lecture material on the chalkboard, giving me the ability to amend and add to my planned notes as students ask questions, extend ideas, and even offer corrections—after all, no instructor is infallible! For the same reason, I engage students in worked examples, solving problems as a group in real time and highlighting potential mistakes as they arise. Finally, to accommodate different learning styles, I bring props into the classroom: for example, sharing a collection of dice prior to a lecture on sample spaces and outcomes. The guided discovery process facilitated by students handling props establishes a foundation upon which I can build a memorable lecture. These approaches incorporate a dimension of flexibility into each lecture, allowing me to mould the content to the students rather than the other way around.

I believe my approaches have a positive effect on student learning, as evidenced by both the objective numeric ratings and the subjective student comments I have received. In turn, this feedback provides reassurance that I am achieving my goal of bringing meaning to theory. That being said, I still seek to improve various aspects of my lecture delivery. Currently, my highest priority is to identify areas where I can increase the use of technology in my lectures, either through using software tools to illustrate concepts (e.g., using JFLAP to construct automata) or through leading some form of live-coding session during algorithm-focused lectures. In past courses, I have introduced students to the \LaTeX document preparation system with the intent that they typeset their assignment submissions using that software, and this was generally received well.
3 Teaching Responsibilities

The following lists outline all courses for which I had some kind of teaching responsibility, including as an instructor or guest lecturer, as an instructional apprentice (IA) or teaching assistant (TA), or as a curriculum developer. Appendix A contains course descriptions for all courses listed here.

All courses from Fall 2017 to present were held at Queen’s University.
All courses from Fall 2015 to Spring 2017 were held at the University of Waterloo.

3.1 Course Instruction

Primary responsibilities include planning and delivering lectures, writing lecture notes, creating assignments/exams, holding office hours, coordinating teaching assistants, and handling administrative matters.

Winter 2019  CISC 203: Discrete Mathematics for Computing II
              49 students, 1 instructor, 2 TAs
              – Based on existing course design with development of new material.

Spring 2017  CS 240: Data Structures and Data Management
              340 students, 3 instructors, 1 IA, 7 TAs
              – Based on existing course design.

3.2 Guest Lecturing

Primary responsibilities include planning and delivering self-contained lectures relevant to course material.

Fall 2019  CISC 203: Discrete Mathematics for Computing II
           246 students, three 1-hour lectures
           – Delivered lectures on graph embeddings and planar graphs.

3.3 Instructional Apprenticeships

The University of Waterloo created the role of instructional apprentice in recognition of the advanced skills of some graduate students. Primary responsibilities include coordinating teaching assistants, creating solution sets and marking schemes, assisting in labs, and leading tutorials.

Fall 2016  CS 234: Data Types and Structures

Spring 2016  CS 240: Data Structures and Data Management
              121 students, two 1-hour tutorial sections per week
              – Presented and worked through example problems related to course material.

3.4 Teaching Assistantships

Primary responsibilities include holding office hours, proctoring midterm/final exams, and marking assignments/exams.

Winter 2020  CISC/CMPE 223: Software Specifications

Fall 2019  CISC 203: Discrete Mathematics for Computing II

Fall 2018  CISC 462: Computability and Complexity

Winter 2018  CISC/CMPE 223: Software Specifications

Fall 2017  CISC 462: Computability and Complexity
3.5 Curriculum Development

Winter 2019  CISC 203: Discrete Mathematics for Computing II
– Wrote ~110 pages of lecture notes with illustrations, proofs, and worked examples.

3.6 Committee Service

2014 – 2015  Curriculum Committee, Department of Computer Science, University of Western Ontario
– Primarily tasked with undertaking a curriculum mapping process to align undergraduate course learning outcomes with CIPS CSAC graduate attributes.

4 Evidence of Teaching Effectiveness

4.1 Summary
The following table summarizes my “effectiveness rating” as an instructor for past courses. Department means are provided when available. Additional data are provided in Appendix B.

<table>
<thead>
<tr>
<th>Term</th>
<th>Course</th>
<th>Effectiveness Rating</th>
<th>Department Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winter 2019</td>
<td>CISC 203</td>
<td>4.7/5.0</td>
<td>3.7/5.0</td>
</tr>
<tr>
<td>Spring 2017</td>
<td>CS 240</td>
<td>4.3/5.0</td>
<td>—</td>
</tr>
</tbody>
</table>

4.2 Representative Feedback from Students

My name is [...]. I was a Discrete Mathematics II student of yours this past semester. Your enthusiasm for the subject matter was infectious, and inspired me to consider Post-Graduate research in Pure Mathematics. I am reaching out to thank you, and wish you the best of luck, whatever your future may hold.
(Email from former CISC 203 student, Winter 2019)

Prof. Smith is very diligent and clear when he teaches. He is organized, and his lectures have a flow/story to them that make the material easy to understand. The assignment problems stimulate my creative-thinking skills. He is also very helpful during office hours. This course could be very boring and difficult without him.
(CISC 203 feedback, Winter 2019)

I liked how genuinely interested Prof. Smith was in teaching the course. His enthusiasm makes coming to this class the highlight of my day. His lecture notes both in-person and online are in-depth and I know that if I have to miss a class, the online notes will prepare me for the next assignment.
(CISC 203 feedback, Winter 2019)

I was originally skeptical about having a grad student as an instructor, but Taylor ended up being one of the better instructors I’ve had. It may be a symptom of him sympathizing more with us undergrads than some tenured profs do, but he targeted his explanations at precisely the right level, and actually answered low-level questions when students were having a tough time understanding the content, never blowing off questions with the equivalent of “I shouldn’t have to explain that, you should be smart enough to figure that out,” which I’ve seen from several professors before.
(CS 240 feedback, Spring 2017)
Very clear and engaging, with small bits of humour thrown in to really keep lectures interesting. And he was always really encouraging, which was especially nice around the midterm.  
(CS 240 feedback, Spring 2017)

4.3 Representative Feedback from Colleagues

Taylor was my choice for head TA for the CISC 223 course. I made this choice when I heard that he has had stellar feedback and standing ovations from students in CISC 203 in Winter 2019. I run my courses by democratic council to avoid power dynamics as well as a redundancy layer for important choices. Taylor has been vital in this process. He has provided great insight for some choices, including midterm and final exam design. He is available for discussion, and I have heard great feedback from students that go to office hours. He also volunteered to do a test run of the midterm; it was a welcome surprise.  
(CISC 203 instructor, Winter 2020)

4.4 Honours and Awards

2020  
Teaching Assistant Excellence Award, Queen’s Society of Graduate & Professional Students  
– University-level award given to recognize the outstanding contributions of a teaching assistant to the SGPS and/or the Queen’s community. One recipient per year.  
– First student from the School of Computing to receive this award.

2018  
Excellence in Teaching Assistance Award, School of Computing, Queen’s University  
– Department-level award given to a teaching assistant in a computing course who went above and beyond in their duties. One recipient per year.

5 Professional Development

5.1 Professional Development in University Teaching and Learning Program, Queen’s University

The Queen’s University Professional Development in University Teaching and Learning program is intended for students and fellows who have an interest in developing themselves in areas of university teaching and learning. The program consists of five components that are specific to each of the areas of university teaching and learning.

| Workshops | Foundations in Teaching and Learning |
| Workshops | Practical Experience |
| Workshops | Educational Leadership |
| Workshops | Scholarship in Teaching and Learning |
| Workshops | Accessible Teaching and Learning |

5.2 Certificate in Professional Development, Queen’s University

The Queen’s University Certificate in Professional Development consists of a series of workshops and seminars to support the academic, personal, and professional success of graduate students and postdoctoral fellows. Participants complete workshops in the areas of Health, Wellness, and Community; Research Skills; Communication; Management and Leadership Development; Career Building; and Setting Ideas in Motion.

| Relevant workshops | Principles of Teaching and Learning |
| Relevant workshops |Preparing a Teaching Dossier |
| Relevant workshops |Leadership 1 |
| Relevant workshops |Leadership 2 |
5.3 Fundamentals of University Teaching Program, University of Waterloo

The University of Waterloo Fundamentals of University Teaching program supports graduate students in developing their knowledge and skills as university TAs and instructors. Participants attend a minimum of six teaching workshops and lead three small-group microteaching sessions.

<table>
<thead>
<tr>
<th>Workshops completed</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective Lesson Plans</td>
<td></td>
</tr>
<tr>
<td>Giving and Receiving Feedback</td>
<td></td>
</tr>
<tr>
<td>Teaching Online – Basic Skills</td>
<td></td>
</tr>
<tr>
<td>Teaching Online – Advanced Skills</td>
<td></td>
</tr>
<tr>
<td>Classroom Delivery Skills</td>
<td></td>
</tr>
<tr>
<td>Teaching Methods</td>
<td></td>
</tr>
<tr>
<td>Teaching STEM Tutorials</td>
<td></td>
</tr>
</tbody>
</table>
A Course Descriptions

CS 234: Data Types and Structures
Top-down design of data structures. Using representation-independent data types. Introduction to commonly used data types, including lists, sets, mappings, and trees. Selection of data representation.

CS 240: Data Structures and Data Management
Introduction to widely used and effective methods of data organization, focusing on data structures, their algorithms, and the performance of these algorithms. Specific topics include priority queues, sorting, dictionaries, data structures for text processing.

CS 462/662: Formal Languages and Parsing

CISC 203: Discrete Mathematics for Computing II

CISC/CMPE 223: Software Specifications
Introduction to techniques for specifying the behaviour of software, with applications of these techniques to design, verification and construction of software. Logic-based techniques such as loop invariants and class invariants. Automata and grammar-based techniques, with applications to scanners, parsers, user-interface dialogs and embedded systems. Computability issues in software specifications.

CISC 462: Computability and Complexity
B  Course Evaluations

Winter 2019 – CISC 203: Discrete Mathematics for Computing II

Response rate: 35 / 49 (71.4%)

Instructor is an effective teacher:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>18</td>
<td>14</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
| This is an excellent course:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>25</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Instructor was sensitive to student needs/interests:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>22</td>
<td>7</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>
| Student interest in subject was stimulated by course:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>17</td>
<td>11</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Instructor presented material clearly:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>22</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
| Instructor was available outside of class:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>27</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Instructor showed genuine concern for students:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>24</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
| Course was well-organized:

<table>
<thead>
<tr>
<th></th>
<th>Strong agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strong disagree</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong agree</td>
<td>22</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Spring 2017 – CS 240: Data Structures and Data Management

Response rate: 64 / 140 (45.7%)

<table>
<thead>
<tr>
<th>Overall effectiveness of instructor:</th>
<th>Organization and coherence of lectures:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excellent: 30</td>
<td>Excellent: 27</td>
</tr>
<tr>
<td>Good: 26</td>
<td>Good: 28</td>
</tr>
<tr>
<td>Satisfactory: 3</td>
<td>Satisfactory: 6</td>
</tr>
<tr>
<td>Unsatisfactory: 1</td>
<td>Unsatisfactory: 0</td>
</tr>
<tr>
<td>Very poor: 1</td>
<td>Very poor: 1</td>
</tr>
<tr>
<td>No opinion: 0</td>
<td>No opinion: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Level of instructor’s explanations:</th>
<th>Instructor’s treatment of student questions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Too high: 2</td>
<td>Excellent: 29</td>
</tr>
<tr>
<td>Somewhat high: 8</td>
<td>Good: 25</td>
</tr>
<tr>
<td>Just right: 44</td>
<td>Satisfactory: 5</td>
</tr>
<tr>
<td>Somewhat low: 5</td>
<td>Unsatisfactory: 0</td>
</tr>
<tr>
<td>Too low: 2</td>
<td>Very poor: 1</td>
</tr>
<tr>
<td>No opinion: 0</td>
<td>No opinion: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Effectiveness of instructor’s visual presentation:</th>
<th>Effectiveness of instructor’s oral presentation:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excellent: 30</td>
<td>Excellent: 34</td>
</tr>
<tr>
<td>Good: 23</td>
<td>Good: 23</td>
</tr>
<tr>
<td>Satisfactory: 5</td>
<td>Satisfactory: 1</td>
</tr>
<tr>
<td>Unsatisfactory: 1</td>
<td>Unsatisfactory: 1</td>
</tr>
<tr>
<td>Very poor: 1</td>
<td>Very poor: 1</td>
</tr>
<tr>
<td>No opinion: 1</td>
<td>No opinion: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Instructor availability outside of class:</th>
<th>Interest in the course:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available: 17</td>
<td>Very interesting: 22</td>
</tr>
<tr>
<td>Usually: 13</td>
<td>Interesting: 35</td>
</tr>
<tr>
<td>Sometimes: 2</td>
<td>Not interesting: 3</td>
</tr>
<tr>
<td>Rarely: 0</td>
<td>No opinion: 2</td>
</tr>
<tr>
<td>Unavailable: 0</td>
<td></td>
</tr>
</tbody>
</table>
1 Course Description


2 Learning Outcomes

A course learning outcome is a brief statement of a skill, competency, or attitude a successful student will achieve by the end of a course. The following list of learning outcomes for CISC 203 is provided by the School of Computing. (http://www.cs.queensu.ca/students/undergraduate/outcomes/CLO.php)

- Critique and construct moderately sophisticated mathematical arguments such as proof by contradiction, proof by induction, proof by minimal counterexample, counting arguments, and recognition of orderings.
- Apply discrete mathematical tools and models such as graph theory, probability, group theory and modular arithmetic to problems such as modelling relational data and networks, scheduling and resource allocation, network design, predicting expected performance, motion planning, cryptography.
- Apply basic discrete probability techniques to computational tasks.
- Build a foundation for further learning by exposure to multiple computer languages, development tools, and methodologies.

3 Personnel

3.1 Instructor

Taylor J. Smith
- Email: tsmith@cs.queensu.ca
- Office: Goodwin Hall, room 235
- Office hours: Tuesday, 1:30pm–2:30pm; Thursday, 12:30pm–1:30pm

3.2 Teaching Assistants

Xavier McMaster-Hubner
- Email: 17xjem@queensu.ca
- Office: Goodwin Hall, room 241
- Office hours: Monday, 10:30am–11:30am

Leonard Zhao
- Email: 16lz1@queensu.ca
- Office: Goodwin Hall, room 241
- Office hours: Wednesday, 10:30am–11:30am
4 Course Details

4.1 Lecture Time/Place

- Tuesday, 12:30pm–1:30pm
- Thursday, 11:30am–12:30pm
- Friday, 1:30pm–2:30pm

All lectures are held in Goodwin Hall, room 254.

4.2 Textbook


4.3 Prerequisites

This course requires [CISC 102 (Discrete Mathematics for Computing I) or MATH 110 (Linear Algebra)] and CISC 121 (Introduction to Computing Science I) as prerequisites, each with a minimum mark of C-.

This course is a prerequisite for CISC 235 (Data Structures), CISC 322 (Software Architecture), CISC 333 (Introduction to Data Mining), and CISC 365 (Algorithms I).

5 Evaluations

5.1 Marking Scheme

- 50% assignments (12.5% per assignment)
- 25% midterm
- 25% final exam

You must pass the final exam to pass the course.

The marking scheme will not be adjusted for individual students, with the exception of students who miss an evaluation due to illness or other extenuating circumstances.

5.2 Assignments

There will be four written assignments in this course. Assignments will be due at the beginning of class on the due date. Only paper assignments will be accepted; email copies will not be accepted.

Assignment solutions must be based on individual work. See Section 6 for more information about academic integrity.

Late assignments will be accepted up to the beginning of the first class following the due date. Assignments submitted later than this time will not be accepted. Late assignments are subject to a penalty of 10% deducted from the earned mark.

If you must miss an assignment due to illness or other extenuating circumstances, please contact the instructor prior to the assignment due date/time. See Section 5.5 for more information about academic considerations and accommodations.

If you have questions about assignment marking, please contact the teaching assistant within one week from the time you received the marked assignment. All assignment marks are considered final after one week has passed from the time you received the marked assignment.
5.3 Midterm and Final Exams

The midterm exam will be held in class during the usual lecture time.

If you have questions about midterm exam marking, please contact the instructor within one week from the time you received the marked midterm exam. All midterm exam marks are considered final after one week has passed from the time you received the marked midterm exam.

The final exam will be held during the examination period in April. It will be scheduled by the Examinations Office. More details are listed in Section 5.4.

If you require accommodations for the midterm or final exam, please follow the procedures listed in Section 5.5.

5.4 Location and Timing of Final Exams

Arts and Science Regulation 8.2.1 states

The final examination in any class offered in a term or session (including Summer Term) must be written on the campus on which it was taken, at the end of the appropriate term or session at the time scheduled by the Examinations Office.

The exam period is listed in the key dates prior to the start of the academic year in the Faculty of Arts and Science Academic Calendar and on the Office of the University Registrar’s webpage. A detailed exam schedule for the Fall Term is posted before the Thanksgiving holiday; for the Winter Term it is posted the Friday before Reading Week, and for the Summer Term the window of dates is noted on the Arts and Science Online syllabus prior to the start of the course. Students should delay finalizing any travel plans until after the examination schedule has been posted. Exams will not be moved or deferred to accommodate employment, travel/holiday plans or flight reservations.

5.5 Academic Considerations and Accommodations

Academic considerations and accommodations are two different mechanisms for helping students in extenuating circumstances. If you have extenuating circumstances for missing a midterm or assignment deadline, or for long-term issues, see the Student Wellness website.

Queen’s University is committed to achieving full accessibility for persons with disabilities. Part of this commitment includes arranging academic accommodations for students with disabilities to ensure they have an equitable opportunity to participate in all of their academic activities. If you are a student with a disability and think you may need accommodations, you are strongly encouraged to contact Student Wellness Services (SWS) and register as early as possible. For more information, including important deadlines, please visit the Student Wellness website.

The Senate Policy on Academic Consideration for Students in Extenuating Circumstances was approved in April 2017. Queen’s University is committed to providing academic consideration to students experiencing extenuating circumstances that are beyond their control and which have a direct and substantial impact on their ability to meet essential academic requirements. Each Faculty has developed a protocol to provide a consistent and equitable approach in dealing with requests for academic consideration for students facing extenuating circumstances. Arts and Science undergraduate students please consult the Faculty of Arts and Science protocol and the portal where they submit a request. Students in other Faculties and Schools should refer to the protocol for their home Faculty.
5.5.1 Academic Considerations and Accommodations for Tests

Some of your instructors may participate in central administration of special tests (including midterms and quizzes) for students with accommodations (other than “use of computer”, which is handled by the exams office); the same process will apply for make-up tests requested via the Arts and Science academic considerations portal. Although we will strive to ensure you are accommodated to the standards in your form, if we do not have, at minimum, 10 working days’ notice, we can’t guarantee that your accommodation needs will be fully met. Shorter notice may be possible with academic considerations for students who don’t have accommodations that complicate scheduling, but we can’t guarantee it.

Students’ accommodation tests will be booked either at the same time as the rest of the class, or as soon after as possible, according to their accommodation requirements. Make-up tests for academic considerations will similarly be booked as soon as possible. Your class schedule will be taken into account when making the booking; due to the expected volume of requests and bookings, we will not be able to book according to the student’s preference. Once we have finalized the booking and arranged for a proctor, you will be contacted by email.

IMPORTANT: Should you elect to write your test with the rest of the class, instead of writing the accommodated test that was booked for you, we require two working days’ notice so that we can cancel your room booking, and notify the proctor that was hired to invigilate your test.

All computer accommodations are handled by the Exams Office. They require 10 working days’ notice to make these arrangements.

If you have questions or concerns, or would like to discuss your case, please feel free to contact us at accommodation@cs.queensu.ca, 613-533-6050, or drop by the School of Computing main office in Goodwin Hall, room 557 (8am–12pm and 1pm–4pm).

6 Academic Integrity

Queen’s students, faculty, administrators and staff all have responsibilities for supporting and upholding the fundamental values of academic integrity. Academic integrity is constituted by the five core fundamental values of honesty, trust, fairness, respect and responsibility (see www.academicintegrity.org) and by the quality of courage. These values and qualities are central to the building, nurturing and sustaining of an academic community in which all members of the community will thrive. Adherence to the values expressed through academic integrity forms a foundation for the “freedom of inquiry and exchange of ideas” essential to the intellectual life of the University.

Students are responsible for familiarizing themselves with and adhering to the regulations concerning academic integrity. General information on academic integrity is available at Integrity@Queen’s University, along with Faculty or School specific information. Departures from academic integrity include, but are not limited to, plagiarism, use of unauthorized materials, facilitation, forgery and falsification. Actions which contravene the regulation on academic integrity carry sanctions that can range from a warning, to loss of grades on an assignment, to failure of a course, to requirement to withdraw from the university.
6.1 Academic Integrity in the School of Computing

Within the School of Computing, in addition to the general concerns about academic integrity, there are a few specific situations.

- **Group work.** Teamwork is an essential part of certain courses. Failure to carry out your own fair portion of the group work may be considered a departure from academic integrity.

- **Collaboration on individual assignments.** In some courses it is permitted to discuss the overall approach to a problem, but not specifics of the solution. With coding exercises, for example, many instructors will consider that you have gone too far if you look at someone’s specific code. It is much safer to consult your TAs or the instructor, who can help you work on your programs without giving away details that cross the line into departures from academic integrity.

- **Internet solutions.** Finding and using solutions on websites is usually a departure from academic integrity.