CSC410: Software Testing and Verification

Course Instructor: Logan Murphy

Fall 2025

E-mail: `lmurphy@cs.toronto.edu`	Lecture Hours: Wed 3-5pm
Course Forum: Piazza	Lecture Room: ES B149
Instructor Office Hours: Tue 4-5pm, BA 2270, Table #6	Tutorial Hours: Fri 3-4pm
TA Office Hours: Fri 4-5pm (In Tutorial Room)	Tutorial Room: ES B149

Course Description

In creating any sufficiently complicated piece of software, errors are inevitably made. To improve the quality of software, developers can attempt to identify bugs by observing executions of the program (testing), or mathematically prove that the program satisfies certain properties (verification). This course explores a variety of testing and verification techniques, both their theoretical foundations and their application using state-of-the-art tools. Topics covered in the course include: automated reasoning with SAT and SMT solvers; software testing and test suite evaluation; deductive program verification; temporal logic and software model checking.

Prerequisites

This course places a heavy emphasis on logic and formal reasoning. You will be expected to be proficient in propositional logic, first-order logic, and discrete mathematics, e.g., graph theory, set theory. You will also need programming experience, with at least a working familiarity with Java, C, and Python, and familiarity with common data structures.

Grading Scheme

- 50% across **four** assignments (each worth 12.5%)
- 15% midterm **Friday, Oct 10th** (in-person, during tutorial slot)
- 35% final exam (in-person, date TBD). You must get **at least 40%** on the final to pass the course.
- **Bonus marks**: up to 5% for active participation in lecture and tutorial, answering questions on the course forum, and participating in the (optional) logic review quiz on Sept 5th.[1]

[1] If you wish to write the quiz but are unable to because of, e.g., accessibility needs or late enrollment, you may reach out to the instructor for an alternative arrangement.

Learning Objectives

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

1. Automated Theorem Proving
   (a) Describe and apply the algorithms underlying modern SAT and SMT solvers, such as DPLL, CDCL, and DPLL(T).
   (b) Use Z3 to solve a variety of problems through reduction to SAT/SMT.
2. Software Testing
   (a) Apply functional (black-box) testing methods to identify software faults: domain partitioning, boundary value testing, and fuzzing.
   (b) Evaluate the quality of test suites using various structural criteria (e.g., decision coverage, path coverage), dataflow criteria, and mutation analysis.
   (c) Reason about program behaviours symbolically and combine symbolic and concrete program analysis.
3. Deductive Verification
   (a) Prove program correctness using Hoare Logic.
   (b) Automate program verification via the inductive assertion method.
   (c) Prove program termination by identifying ranking functions.
   (d) Use Dafny to automatically verify imperative and recursive programs.
4. Model Checking
   (a) Formalize system properties in Linear Temporal Logic (LTL) and Computation Tree Logic (CTL), and reason about the semantics of such specifications.
   (b) Describe and apply algorithms underlying software model checkers.

Course Materials

This course has no official textbook. Since we cover a wide range of topics, there is no single text which covers all the course material. Instead, we will mostly assign readings of selected chapters from the following references, all of which are either freely available online or have electronic versions accessible through the university library system:

- *Decision Procedures* (DP) by Daniel Kroening and Ofer Strichman
- *Software Testing and Analysis* (STA) by Mauro Pezze and Michal Young
- *The Calculus of Computation* (CC) by Aaron R. Bradley and Zohar Manna
- *Principles of Model Checking* (PMC) by Christel Baier and Joost-Pieter Katoen

Other course materials (papers, slides, code examples, etc.) will be posted on Quercus.

Course Outline

The schedule is tentative and subject to change based on the pace of discussion in lecture.

- **Week 1**. Lecture on Sept 3.
  Course Introduction & Logic Review
  Sept 5: (Optional) Review Quiz
- **Week 2**. Lecture on Sept 10
  SAT Solving
- **Week 3**. Lecture on Sept 17
  Satisfiability Modulo Theories
- **Week 4**. Lecture on Sept 24
  Introduction to Testing, Black-Box Testing Methods
- **Week 5**. Lecture on Oct 1
  White-Box Testing Methods
- **Week 6**. Lecture on Oct 8
  Symbolic Execution & Concolic Testing; Midterm Review
- **Week 7**. Lecture on Oct 15
  Introduction to Deductive Verification, Hoare Logic
- **Week 8**. Lecture on Oct 22
  Inductive Assertion Method, Program Termination
- **Week 9**. Lecture on Nov 5
  Introduction to Model Checking, Linear Temporal Logic
- **Week 10**. Lecture on Nov 12
  Computation Tree Logic
- **Week 11**. Lecture on Nov 19
  Model Checking Algorithms
- **Week 12**. Lecture on Nov 26
  Course Wrap-up & Exam Review

Important Dates

Assignment releases and due dates subject to change. Bold dates are fixed.

- **First Lecture: September 3**
  Optional Logic Quiz: **September 5**
- Assignment 1 Released September 10
- Assignment 1 Due September 24
- Assignment 2 Released September 24
- Assignment 2 Due October 8
- **Midterm: October 10**
- Assignment 3 Released October 15
- **Reading Week: October 27–31**
- Assignment 3 Due November 5
- Assignment 4 Released November 5
- **Course Drop Deadline : November 11**
- Assignment 4 Due November 19

Teaching Assistants (TAs)

Ruotong Cheng, Maite Kramarz, Eric Liu

Lectures

Lectures will be run on UofT time (i.e., starting 10 minutes after the hour). Attendance in lecture is expected for all students throughout the semester. Lecture materials (slides or notes) will be posted on Quercus, but they may not capture the entirety of the discussion. Lectures will not be recorded. Students are not permitted to take photos/video during lecture, except with the explicit permission of the instructor.

Tutorials

The "tutorial" slot (Fridays 3-5PM) will serve multiple functions. For most weeks, the first hour will be focused on working through problems with a TA, while the second hour will be used for TA office hours. The tutorial slot will also be used for the midterm (see the course calendar). On special occasion, we may also use this slot for guest lectures from faculty or senior graduate students to discuss contemporary research relevant to the course topics.

Like lectures, tutorials will be run on UofT time.

Assignments

The assignment problems will be primarily *hands-on*, in that you will often be asked to apply the concepts discussed in lecture by implementing some program or using some state-of-the-art tool. All assignments may be completed by **groups of up to 3 students**. Only one submission needs to be made by each group. The team format is meant to provide you with a local support group to quickly learn the necessities together. You do not have to stick with the same group throughout the term – feel free to switch groups.

Each assignment will be released at the beginning of a course "module". The problems in the assignment will align with the topics covered in each week of that module. For example, if a module starts in week i and the assignment covers two week's worth of material from that module, you can expect some problems based on material from week i, others from week $i+1$, and the full assignment will be due in week $i+2$.

Put another way: if the assignment is due in week n, then we will have covered all the relevant material needed to complete it no later than week $n-1$. But, it is strongly recommended to begin working on assignments *early*, so that you can incrementally work through the problems as we progress through the material in class.

Assignments will be submitted on MarkUs.

Everything is due at 11:59:59 pm (Toronto time) on its due date.

No questions about an assignment (on Piazza or via e-mail) will be answered by the course instructor or the TAs during the last 24 hours before the assignment is due. Plan accordingly!

Late Submission Policy

Assignment submissions will remain open on MarkUs for 12 hours after the deadline. A 4% deduction will be automatically applied each hour after the submission deadline up to a maximum of 12 hours.

Remark Requests

All remarking requests must be received within **two weeks** of the date when the assignment was returned. It is your responsibility to check for your posted grade or your returned assignment. Remark requests must be submitted through MarkUs.

Note that there is no reconsideration for auto-graded code assignments. Yes, it is unfair to lose all marks due to a small programming mistake. But you are computer scientists now; small coding mistakes can have big consequences in the real world. Learn not to make (many of) them!

If your request is about requesting a higher grade, be sure to explain in your remark request in detail. Note that your mark may *decrease* if, upon reviewing your work, we see that you were incorrectly awarded too high a mark. You should expect a turnaround time for remark requests of about two weeks.

Practice Problems

In addition to the assignment problems, we will also periodically release some practice problems. These will be more theory-oriented and are intended to be used to check your understanding of the course material, as well as to practice for tests. These will not be graded, they are only for your own practice.

Midterm and Final

We will hold a midterm worth 15% of the course grade on **October 10th**, and a final exam worth 35% of the course grade (the date and time of which will be determined by the Faculty of Arts and Science later in the term and communicated to the class). The midterm will be based on material up to and including Week 5, and the final exam will be cumulative, i.e., will be based on material from the whole term. Remark requests for the midterm are subject to the same policies as for assignments.
See the “Special Considerations” section below for the course policy regarding student absence from scheduled tests and examinations.

Academic Integrity

Do not submit someone else’s words, symbols, or code without giving them credit. This is called plagiarism and it is an academic offence.

All suspected cases of academic dishonesty will be investigated following procedures outlined in the Code of Behaviour on Academic Matters. If you have questions or concerns about what constitutes appropriate academic behaviour, please reach out to me. Note that you are expected to seek out additional information on academic integrity from me or from other institutional resources (for example, the University of Toronto website on Academic Integrity).

Statement on Artificial Intelligence Tools

Generative AI is not required to complete any aspect of this course, and I caution you against relying on AI tools as a learning aid. Current AI tools remain unreliable when it comes to many of the topics and techniques we study in this course. Ultimately, you (and not any AI tool) are responsible for your own learning in this course, and for all the work you submit for credit. Over-reliance on generative AI may give you a false sense of how much you've actually learned, which can lead to poor performance on the midterm test or final exam.

Nevertheless, I know that some students find AI tools useful to help clarify certain aspects of lecture materials or assignment questions. So, the expectation in this course is that you will treat interactions with AI tools the same way you treat interactions with other students and TAs. It is normal to ask your peers and TAs to help clarify something mentioned in lecture, or (at a very high level) how one might approach a problem. But you (presumably) would not ask another student or a TA to complete your assignment for you. In the same way, you may ask an AI tool to clarify something from lecture, or suggest a high-level approach to a problem, but **you cannot ask an AI tool to complete an assignment for you.** This will be considered cheating.

If you *do* choose to use generative AI tools while working on any of the assignments, you are required to disclose which tool was used, in what capacity it was used, and provide transcripts of your interactions with such tools if any exist. This is to ensure that any help you get from an AI tool remains at a "high level", similar to the help you might get from a peer or TA. Steps for disclosing this information will be provided on the assignment handouts. Submissions which were completed using the assistance of generative AI tools (in any capacity) *without* appropriate disclosure and documentation may be considered as instances of an academic offence.

Communication

Your recourse for *technical* questions (i.e., questions about course material), outside of lecture, are (1) office hours and (2) the Piazza forum. Asking questions on the Piazza forum is strongly encouraged, because then other students are able to benefit from the conversation. Obviously, you should not post details about your solutions to assignments on Piazza. If you want to ask detailed questions about *your* approach to an assignment problem, you should do so during office hours.

You should e-mail the instructor (`lmurphy@cs.toronto.edu`) for *personal* matters, or to ask for *special considerations* (see the policy on special considerations below). You should expect a reply within two days (excluding weekends). If you don't receive a response in that time, send a follow-up e-mail.

Always sign your full name (as it appears on university records) at the bottom of your e-mails. Always include "CSC410" in the subject line of your e-mails. If your question is related to MarkUs, include your MarkUs handle in the e-mail.

Communications which do not adhere to the above guidelines will be ignored.

Do not e-mail your TAs. Answering student emails is not a part of their contract.
Course-wide announcements will be made via Quercus.

Special Considerations

If you are unable to complete an assignment due to major illness or other circumstances completely outside of your control, please contact your instructor immediately in order to receive special consideration. Note that special consideration will be considered on an individual basis and will not be given automatically—in other words, you risk getting a mark of zero for missed work unless you contact your instructor promptly. Unless you are physically incapable of doing so due to extreme illness, contact your instructor before the due date and ask for an extension.

If you are unable to write the midterm due to major illness or other circumstances completely outside of your control, please contact your instructor as soon as possible. You will be asked to provide appropriate documentation (e.g., by filling out the absence declaration form). The weight of the missed midterm will be moved to the final exam (so the final will be worth 50% of your grade).

If you are unable to write the final exam due to major illness or other circumstances completely outside of your control, you may submit a deferred exam petition.