# CAS 760
# Simple Type Theory
# Winter 2026

# 0 Course Overview

William M. Farmer

Department of Computing and Software
McMaster University

December 29, 2025

# Outline

- Introductions.
- Mission of the course.
- Resources.
- Course organization.

# 1. Introductions

# Instructor: Dr. William M. Farmer

- Professor, Dept. of Computing and Software.
- B.A., University of Notre Dame.
  M.A., M.S., Ph.D., University of Wisconsin-Madison.
- P.Eng. (Licensed Professional Engineer in Ontario).
- Industrial experience. Research computer scientist for 12 years at The MITRE Corporation in Bedford, MA, USA.
- Teaching: Discrete mathematics, logic, principles of programming, software design, communication skills.
- Research: Logic, mathematical knowledge management, mechanized mathematics, formal methods.
- Software development: Developed the IMPS Interactive Mathematical Proof System with J. D. Guttman and F. J. Thayer at the MITRE Corporation (1990–1993).

# My Contact Information

- Office: ITB 163.
- Email: wmfarmer@mcmaster.ca.
- Web: http://imps.mcmaster.ca/wmfarmer/
- Office hours: To see me in person or online, please send me a note with times you are available.

# Who are You?

- I would like to get to know each of you.
- Please complete a bio sheet in PDF format for yourself.
  - Use my bio sheet on Avenue under Content / Course Information as a model.
  - Submit it as an assignment on Avenue under Assessments / Assignments.
- Please complete the CAS 760 Precourse Questionnaire on Avenue under Resources / Surveys.

## William M. Farmer

**Preferred name:** Bill.

**Pronunciation of name:** As expected.

**Pronouns:** he/him.

**Hometown:** South Bend, Indiana, USA.

**Education**

- Ph.D., mathematics, Wisconsin-Madison (1984).
- M.S., computer sciences, Wisconsin-Madison (1983).
- M.A., mathematics, Wisconsin-Madison (1980).
- B.A., mathematics, Notre Dame (1978).

**Work experience**

- Professor at McMaster University (2005–present).
- Associate Professor at McMaster University (1999–2005).
- Assistant Professor at St. Cloud State University (1997–1999).
- Lead Scientist at The MITRE Corporation (1985–1997).

**Professional information**

- *Research interests.* Logic, mathematical knowledge management, mechanized mathematics, and formal methods.
- *Released software.* W. M. Farmer, J. Guttman, and J. Thayer, *IMPS Interactive Mathematical Proof System*, developed at The MITRE Corporation (1990–1993).
- *Published textbook.* W. M. Farmer, *Simple Type Theory, Second Edition*, Birkhäuser/Springer (2025). Available at https://link.springer.com/book/10.1007/978-3-031-85352-4 .
- *Administration.* Chair (2011–2016) and Acting Chair (2025), Dept. of Computing and Software, McMaster University.

**Personal information**

- *Family.* I am married and have a son and a daughter. I am the oldest of nine children. I have 41 cousins, 18 aunts and uncles, and 20 niblings (nieces and nephews).
- *Sports.* I am an avid basketball player and greatly enjoy hiking and canoeing.
- *Film.* Classics and foreign. Favorite directors are Bergman, Hitchcock, and Kurosawa.
- *Music.* I like all kinds of music. A favorite performer and composer is Mark O'Connor.
- *Reading.* Novels, history, biographies, plays, poetry, mythology, math papers, op-eds.
- *Volunteer work.* I have been a Scout leader for 25 years.
- *Citizenship.* I am a dual American and Canadian citizen.

**In 10 years, I hope that I will:** Be retired and writing a memoir of my life.

# 2. Mission of the Course

# What is Simple Type Theory?

- Simple type theory is a classical higher-order version of predicate logic.
  - Familiar to some computer scientists but not to many mathematicians, engineers, and other scientists.
- Form of type theory.
  - Types are used to classify expressions by value and control the formation of expressions.
  - Classical: nonconstructive, 2-valued.
  - Higher order: quantification over sets.
  - Can be viewed as a weak set theory.
- Natural extension of first-order logic.
  - Based on the same principles as first-order logic.
  - Includes $n$th-order logic for all $n \geq 1$.
- Simple type theory is simpler than dependent type theory, a constructive form of type theory.

# What is Alonzo?

- Church's type theory is a version of simple type theory introduced by Alonzo Church in 1940.
  - Tailored for reasoning with functions.
  - Includes lambda notation, definite description, and a type of Boolean values.
  - A simple, elegant, highly expressive, and practical logic.
  - Underlying logic for several functional programming languages and proof assistants.
- Alonzo is a version of simple type theory based on Church's type theory that admits undefined expressions according to the traditional approach to undefinedness.
  - Undefinedness results from partial functions and definite descriptions.
- Alonzo is a practice-oriented logic that is designed to be as close to mathematical practice as possible.

# Mission

1. Introduce students to Alonzo as a practical logic for expressing and reasoning about mathematical ideas.
2. Provide a foundation for the study of higher-order logic and type theory and the use of programming languages, proof assistants, and other mathematical software systems based on higher-order logic and type theory.

# Learning Objectives: Precondition

Students taking this course are expected to have some familiarity with:

1. University-level mathematics.
2. Set theory (including ordinals and cardinals).
3. First-order logic.
4. Recursive definitions.
5. Mathematical proof (including proof by induction).
6. Decidability.

# Learning Objectives: Postcondition [1/2]

Students should know and understand:

1. The logical principles underlying predicate logics such as first-order logic and simple type theory.
2. The general and standard semantics for simple type theory.
3. What mathematical structures are and how they can be specified in a simple type theory like Alonzo.
4. How to reason with undefinedness in a logic like Alonzo that admits undefined expressions.
5. Proof systems for simple type theory.
6. The little theories method for organizing mathematical knowledge.
7. What kind of software systems support the use of simple type theory.

# Learning Objectives: Postcondition [2/2]

Students should be able to:

1. Express and reason about mathematical ideas in simple type theory.
2. Write a theory in simple type theory to specify a single mathematical structure or a collection of mathematical structures.
3. Build a mathematics library as a collection of theory developments connected by theory morphisms.

# 3. Resources

# Online Course Information

- Course Outline on Avenue.
- Announcements on Avenue.
- Discussion Forum on Avenue.

# Resources

- Course web site is on Avenue: http://avenue.mcmaster.ca/ .
- W. M. Farmer, Simple Type Theory: A Practical Logic for Expressing and Reasoning about Mathematical Ideas, Second Edition [abbreviated STT], Computer Science Foundations and Applied Logic, Birkhäuser/Springer, 2025. Available at

  https://link.springer.com/book/10.1007/978-3-031-85352-4 .
- W. M. Farmer, LaTeX for Alonzo, 2024. LaTeX macros and environments for writing types, expressions, theories, etc. in Alonzo. Available on Avenue.
- W. M. Farmer and D. Y. Zvigelsky, Monoid Theory in Alonzo: A Little Theories Formalization in Simple Type Theory, Journal of Applied Logics, 12:1853–1939, 2025. Available on Avenue.

# 4. Course Organization

# Work Plan

- Lectures. TuTh 12:00-1:30 PM.
- Bio sheet.
- Meaningfuls and memorables (M&Ms).
- Discussion forum.
- Assignments.
- Development project.
- Software project.

# Policy Statements

1. Midterm course review.
2. End-of-term course evaluation.
3. Discrimination.
4. Academic integrity.
5. Authenticity/plagiarism detection.
6. Courses with an online element.
7. Online proctoring.
8. Conduct expectations.
9. Academic accommodation.
10. Copyright and recordings.
11. Extreme circumstances.
12. Other policy statements.

# Marking Scheme

| | |
|---|---|
| Bio sheet | 2% |
| M&Ms | 8% |
| Assignments | 40% |
| Development project | 30% |
| Software project | 20% |
| **Total** | **100%** |

# Syllabus

0 Course overview.
1 Review of first-order logic.
2 Introduction to simple type theory (Chapters 1–2 STT).
3 Preliminary concepts (Chapter 3).
4 Alonzo: syntax and semantics (Chapters 4–7).
5 Proof Systems (Chapter 8, Appendices A–C).
6 Theories (Chapters 9–11).
7 Developments (Chapters 12–13)
8 Morphisms (Chapter 14).
9 Alonzo variants (Chapter 15).
10 Software support (Chapter 16).
11 Project presentations.

# The End.