Software Eng. 2FO3 - Logic For Software Engineering

INSTRUCTOR:
Dr. Mark Lawford
Office: ITB 160; E-mail: lawford@mcmaster.ca
Tentative Office Hours: Friday 14:30-16:30

TEACHING ASSISTANTS:
Grad: Vera Pantelic (pantelv@cas.mcmaster.ca)
Office: ITB/205
Undergrad: Emery Finkelstein (finkelem@cas.mcmaster.ca)

LECTURES:

- Location: JHE/264
- Time: Tuesday, Wednesday & Friday 12:30-13:20

TUTORIAL/LABS:

- Tutorial: Tuesdays 10:30-11:20 for Section 1, Friday 10:30-11:20 for Section 2

MISSION:

The role of logic in software engineering is much like the role of calculus in other fields. Logical expressions can be used to describe designs and logical analysis used to analyse design documents. This course teaches logic in much the same way that early engineering calculus courses teach calculus. Rather than exploring the fundamental assumptions and methods of logic, this course teaches how logic can be used by the software developer. The course provides basic knowledge of the terminology and notation of the field and stresses applications. Issues that would concern mathematicians and logicians get relatively little time although students will know enough about those issues that they will be able to understand discussions where they are mentioned.

OBJECTIVES:

The student will learn to use mathematical logic to describe properties of program states and to verify program properties. Students will be introduces to an automated theorem prover system.

GRADING:[1]

Two marking schemes are provided. In order to have your assignments count in your final mark (Scheme A), you must pass (obtain $\geq 50\%$) on the combination of your attendance, midterm, &

[1]The instructor reserves the right to conduct deferred examinations orally.

final (Scheme B). Provided you pass by Scheme B, your final mark will be the max(Scheme A, Scheme B).

Scheme A			Scheme B	
Midterm exam	20%	or	Midterm exam	25%
Assignments/Quizzes	10%		Assignments/Quizzes	0%
Attendance/Participation	10%		Attendance/Participation	5%
Final exam	60%		Final exam	70%

NOTE: In order to pass the course you must pass the combination of the midterm and final. Also, a student may miss 4 lectures without an attendance penalty. Additional missed classes incur a $-\frac{1}{2}\%$ penalty.

TEXT:

- Michael R.A. Huth and Mark D. Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems* (2nd Edition), Cambridge University Press, 2004.

ADDITIONAL REFERENCES:

- Jean E. Rubin, *Mathematical Logic: Applications and Theory*, Saunders College Publishing, 1990.

DETAILED COURSE OUTLINE:

Why we need logic in software (and hardware) design: (slides: Intro.pdf)

- Problems in writing and checking specifications
- Problems in program documentation and inspection
- Examples: Pentium floating point bug, safety critical software - reactor shutdown systems

Propositional Logic: (slides: prop.pdf)

- Notation, truth tables
- Tautologies
- Logical equivalence, logical implication - Applications: Circuit simplification and "Building the world with NAND"
- Normal forms (DNF, CNF) - Application: Minimizing gate delays
- Axioms, rules of inference (slides: prop2.pdf)
- Deduction, resolution, duality
- Valid arguments and rules of inference, consistent assumptions
- Use of propositional logic to describe program states.
- Sequent calculus & Propositional reasoning in PVS (slides: seq.pdf)

Predicate Logic: (slides: pred.pdf)

- First-order quantification
- Interpretations, models, and validity - tautologies involving quantifiers
- Proofs by predicate logic from premises and inference rules (slides: pred2.pdf)
- Predicate logic with equality (slides: predeq.pdf)
- Use of Predicate logic to describe program states
- Proving program properties in PVS (slides: pvs4pred.pdf)
- Higher Order Logic (slides: HOL.pdf)

Mathematical Induction: (slides: induction.pdf)

- inductive definitions
- weak and strong induction
- proof by induction in PVS

The problem of partial functions and non-denoting terms in logic: (slides: partial.pdf)

Types: (slides: type.pdf)

- Ill-defined sets and paradoxes - Liar's paradox, Russell's paradox
- Sets, sorts, types and signatures
- Subtypes and dependent types
- Type checking
- Type checking in PVS
- Tabular specifications in PVS (slides: powercond.pdf)
- Treatment of undefined terms

Introduction to completeness, soundness and other issues: (slides: tba)

- Overview of completeness, soundness and decidability results for propositional and predicate logic
- Complexity of decision procedures
- Expressive power of logics

Survey of other types of logics: (slides: tba)

- Equational logic
- Intuitionist logic
- Multiple valued logics
- Temporal logics LTL and CTL

Model checking: (slides: model.pdf)

- Review of models
- Model Checking
- Verification of program safety and liveness properties
- Model checking with PVS and other tools

Program Verification: (slides: tba)

- Hoare Triples
- Partial and Total Correctness

NOTES:

Discrimination

"The Faculty of Engineering is concerned with ensuring an environment that is free of all adverse discrimination. If there is a problem that cannot be resolved by discussion among the persons concerned individuals are reminded that they should contact there Chair, the Sexual Harassment Office or the Human Rights Consultant, as soon as possible."

Academic Dishonesty

"Academic dishonesty consists of misrepresentation by deception or by other fraudulent means and can result in serious consequences, e.g. the grade of zero on an assignment, loss of credit with a notation on the transcript (notation reads: 'Grade of F assigned for academic dishonesty'), and/or suspension or expulsion from the university. It is your responsibility to understand what constitutes academic dishonesty. For information on the various kinds of academic dishonesty please refer to the Academic Integrity Policy, specifically Appendix 3, located at
`http://www.mcmaster.ca/senate/academic/ac_integrity.htm`
The following illustrates only three forms of academic dishonesty:

1. Plagiarism, e.g. the submission of work that is not one's own or for which other credit has been obtained. An example is copying all or part of someone's assignment and handing it in as your own.

2. Improper collaboration in group work.

3. Copying or using unauthorized aids in tests and examinations."

Calculators

Calculators are not needed for this course and their use will not be permitted during tests.

Bonus Marks

At the discretion of the instructor, a student will receive 1 to 2 "bonus marks" on their latest assignment for being the first person to point out a technical error in the lecture slides or assignment handout.