Overview: Software Testing and Verification is an advanced survey course on concepts, principles, and techniques related to software testing and formal program verification. Students will become acquainted with both the strengths and limitations of various functional and structural testing methods, as well as techniques for proving the functional correctness of sequential programs. Topics include: black-box and white-box test case design strategies, incremental integration testing techniques, inspections and reviews, axiomatic verification techniques, predicate transforms, and function-based verification. Students will practice the techniques presented in class via individual and/or group exercises.

Prerequisites: Successful completion of an upper division undergraduate or graduate-level software engineering survey course (such as CEN 3031/5035) or permission of the instructor. Familiarity with programming using a high-level language (C, C++, Java, etc.) and basic knowledge of algorithms, data structures, and discrete math is assumed.

Textbook: None. A collection of required readings will be available for purchase as a packet.

Outline of Course Topics: (subject to change!)

- Intro to V&V Techniques and Principles
- Formal Program Specification
- Requirements and Specifications
- Axiomatic Verification
- Black-Box Test Case Design Strategies
- Weak Correctness
- Partition Testing
- Rules of Inference
- Combinatorial Approaches
- Strong Correctness
- Other Strategies
- Predicate Transforms
- White-Box Test Case Design Strategies
- Proving Strong Correctness
- Logic Coverage
- Computing Weakest Pre-conditions
- Dataflow Coverage
- Functional Verification
- Path Conditions and Symbolic Evaluation
- Complete and Sufficient Correctness
- Other Strategies
- Axiom of Replacement
- Integration and Higher Level Testing
- Correctness Conditions
- Testing Object-Oriented Software
- Iteration Recursion Lemma
- Reviews and Inspections
- Revisiting Loop Invariants
- Testing Tools
- Cleanroom Software Engineering

Examinations and Grades: Course grades will be based SOLELY on two equally weighted 90-minute exams and a number of pass/fail (take-home) problem sets.

Workload: Probably nominal for a non-programming course. Students who are conscientious in completing problem sets tend to perform much better on the exams.