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 test design methods, as well as techniques for proving the functional correctness of sequential programs. Topics include: black-box and white-box test design strategies, incremental integration testing techniques, inspections and reviews, axiomatic verification techniques, predicate transforms, and function-based verification. Students will have the opportunity to practice the techniques presented in class via optional 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, object-oriented design principles, and discrete math is assumed.

Textbook: A collection of required readings will be available for purchase as a packet. An optional textbook, Pezze and Young’s *Software Testing and Analysis*, Wiley, 2008, is recommended for students who wish to have additional software testing and analysis reference material at their disposal.

Outline of Course Topics:

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

Examinations and Grades: Course grades will be based SOLELY on two equally weighted 90-minute exams.

Workload: Probably nominal for a non-programming upper division/graduate level course. Students who are conscientious in completing optional exercises tend to perform better on the exams.