DESCRIPTION: This course introduces the general topic of quality engineering, which encompasses software testing, quality assurance and process improvement. We will examine concepts of quality management across the lifecycle, the intellectual challenges, research approaches, and current industry practices. Organizational infrastructure, automated tools and practices to support quality engineering—walkthroughs and inspections, software testing, problem tracking and reporting, defect causal analysis. Contemporary practices of extreme programming, exploratory testing, test-driven development. Static/dynamic analysis techniques—model checking, model based testing.


REFERENCES:

RESEARCH PAPERS: A reading list will be provided.

OBJECTIVES: The purpose of this course is manifold: to expose students to the breadth of the subject; to give students experiences performing a variety of quality engineering tasks; to introduce students to research in verification and validation; and to expose students to industry practice and career opportunities in quality engineering. This experience, combined with lectures, assigned research papers, and student investigations, will give the student an appreciation for the technical, management, and economics of quality engineering. A theme: achieving cost-effectiveness by leveraging technology.

TEACHING APPROACH: The following types of activities are used.
- Assigned reading and lectures from textbook.
- Assigned research papers/projects with in-class discussions.
- Student led discussions of outside readings (textbook and research articles)
- Individual "industry practices" investigations and presentations (guests identified by student)
- Individual and group V&V task assignments
- Term team V&V project on existing software applications
- Online, in-class and/or take-home examinations
- A course lessons learned experience report or research paper.
GRADING:
Percentages:  20% Quizzes  20-30% Exams (2)  40-50% Projects  20% Research
Scale:   A = 90-100  B = 80-89  C = 70-79  D = 60-69  F = below 60

Conceptual Objectives: The student successfully completing this course the student shall understand and be able to:
• read, assimilate and critique technical papers from journals and conferences
• state the influence of the requirements problem on validation
• state approaches to software validation across the software life cycle
• distinguish between verification and validation
• describe the major concepts exploratory testing and test-driven development
• identify metrics relevant to validation & verification
• define and critique software inspection processes
• explain and distinguish concepts of reliability, dependability and safety
• understand professional careers in software quality
• describe the use of formal and semi-formal methods in verification
• describe the test lifecycle and its deliverables
• explain coverage criteria for white-box testing
• articulate differences in testing approaches for web-based, GUI-based, component based software
• identify existing and emerging standards for software quality.

Performance Objectives: After successful completion of this course, the student shall be able to do the following:
• Implement elementary test automation for code units, GUIs
• analyze and document testing requirements given software requirements
• apply a systematic technique to derive test cases, given a software specification
• utilize specific specification techniques such as decision tables, finite-state modeling
• develop white-box test cases using control-flow and data-flow adequacy criteria
• develop automated black-box tests for functions, classes and application
• organize, staff and conduct a software inspection
• implement tests using stored test data and automated test scripts.