California State University, Sacramento (CSUS)
Computer Science Department

CSC 179
– Software Verification and Validation –
Spring 2019

Instructor: Doan Nguyen, Ph.D.
Office: Riverside Hall 5009
E-mail: doan.nguyen@csus.edu
Office Hours: M 9:00 am – 12:00pm

Course Description: Verification and Validation (V & V) techniques to identify and resolve software problems and high-risk issues early in the software lifecycle. Application of V & V to all phases of the lifecycle process. Includes planning and reporting on the V & V effort. Topics also include software quality assurance and software testing.

Prerequisite: fully classified graduate standing in Computer Science or Software Engineering, or fully classified graduate standing in Computer Engineering and CSc131

An Overview: An often neglected, but extremely important part of the software development process involves testing the product to ensure that high quality product is released. When done correctly, this entails developing a strategy for testing the software, preparing a plan for implementing the strategy, generating test cases, and executing test cases in an attempt to detect and eliminate errors. The objective of the testing efforts is to "release a high quality product into the field" and maintain a competitive edge in the market.

Textbook: Introduction to Software Testing (Authors: by Paul Ammann, Jeff Offutt ISBN-13: 978-1107172012)

Extensive lecture notes, research papers, handouts, and other reading materials will be assigned.

Course Goals: Upon successful completion of this course you will be able to:

- Understand the basic concepts of software verification and validation
- Investigate and understand the prominent software testing techniques and tools.
- Develop a software verification and validation plan
- Perform effective & efficient structural testing of software under test
- Perform effective & efficient functional testing of software under test
- Integrate & test the various units & components of software under test
- Plan, track & control the software testing effort

Academic Honesty: All students are expected to maintain high standards of academic integrity. All work you submit should be your own. All suspected cases of academic dishonesty would be reported and pursed.

Attendance policy: All students are expected to attend all classes. Unexcused absences from quiz, midterm exam, final exam, or project presentations will result in zero grades for what the student missed. All assignments are due at the beginning of class on the due date, unless otherwise specified. Late assignments will be accepted with 5% deducted for each day late. No late work to be accepted after one week of due date. Attendance quizzes cannot be made up.

In-Class Computers and Communication:

Phone calls, text messages, instant messages, email, and general web surfing is not allowed during class time. Computers may only be used during assigned time.

Grading:

Final Exam	25%
Attendance, participation, quizzes	15%
Project, presentation, assignments	40%
Midterm	20%

Class Participation: All students are expected to make a regular contribution to the class discussions. You should be prepared to offer your feedback, analysis and comments regarding material presented or assigned.

Quality documentation is critical for the success of the projects and counts toward your final and project grade.** ***All work submitted must be typed.

Professional Societies:

Many technical disciplines have a number of professional societies and publications that exist to provide forum of presenting professional views and advancements in the state-of-the-art. As a computer scientist or software engineering, you should belong to and be active in at least one professional society.

Grading Scale:

Range	Letter Grade
94-100	A
90-93	A-
87-89	B+
84-86	B
80-83	B-
77-79	C+

74-76	C
70-73	C-
67-69	D+
64-66	D
60-63	D-
59 or Less	F

Weekly Tentative Schedule:

Week	Topic
1	Software quality assurance, Verification & validation
2	Introduction to software testing (Unit, Module, Subsystem, and System Testing), Testing Software Specification, Testing tools
3	Model Driven Test Design (MDTD)
4	Black box testing Test Automation
5	White box testing
6	White box testing (cont)
7	Introduction to Test-Driven Development (TDD) Web application testing, Usability testing
8	Code Inspection Midterm examination
9	Testing the documentation
10	Configuration Management (CM). Introduction to Git/GitHub software.
11	Testing for security
12	Mutation testing (advanced testing)
13	Software testing metrics
14	Project presentations I
15	Project presentations II
16	Review and Final Examination