EECE 5132-001 Software Test & QA

EECE 6032-001 Software Test & QA

Fall Semester 2018

Instructors: Ken A. Baker

Contact information:
Email: baker2kk@mail.uc.edu

Office Hours: Presently there are no office hours. We can meet by appointment via Webex, or Skype if need be.

Credit Hours: 3

Course Description:

Course Timeframes and location:

Type	Days & Times	Room	Meeting Dates
Lecture	Online		08/27/2018 – 12/15/2018

Subject to change with notice on Blackboard. May have many revisions of the syllabus so please check for updates.

Course Textbooks:

Optional but used for the first lecture:

Tian, Jeff (2005)Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement ISBN: 978-0-471-71345-6

The above book is available in its entirety on slideshare.net at:

https://www.slideshare.net/Shani729/book-software-quality-engineering-jeff-tian?qid=60473293-9233-4ae4-b4f0-d9eb5a6f30bf&v=&b=&from_search=3

The book below is the primary textbook. **Note the 1st edition WILL NOT WORK!** There have been significant revisions completed to produce the second edition.

Ammann, P., & Offutt, J. (2016). *Introduction to software testing* (Edition 2. ed.). Cambridge, United Kingdom; New York, NY, USA: Cambridge University Press. ISBN: 9781107172012 eBook ISBN: 9781316774366

and

Gregory, J., & Crispin, L. (2015). More agile testing: learning journeys for the whole team. Upper Saddle River, NJ: Addison-Wesley. ISBN: 9780321967053 eBook: 9780133749564

Course Learning Objectives:

- Fundamentals of software quality, quality assurance, verification and validation perspectives. Software testing types and concepts, test planning and management, software inspection. Formal methods, quantifiable improvement, software reliability, and global software engineering.
- Course Goals: The goals of this course are for each student to understand the basic principles of software testing and quality assurance, and their role in contemporary software engineering. Students will examine research areas in software testing and quality assurance, and be prepared to conduct research in software engineering in general

COURSE REQUIREMENTS

- Students are required to view online and actively participate in each discussion. Class participation is part of grading. Class participation is positively correlated to student grade: the more a student participates, the higher the grade will be achieved.
- The concepts from the lecture and discussions will be tested in exams, as well as be applied and reinforced through individual assignments. Various documents (models, presentations, etc.) and working programs
  will be the deliverables.

GRADING

Mid-term – 20%
Final – 20%
Discussions – 30%
Participation – 20% total, 10% critiques and feedback, 10 participation in the course itself.
Quizzes – 10%

 Letter grades are assigned as follows:
A [92, 100]
A- [88, 92)
B+ [84, 88)
B [80, 84)
B- [76, 80)
C+ [72, 76) F [0, 52)
C [68, 72)
C- [64, 68)
D+ [60, 64)
D [56, 60)
D- [52, 56)
F [0, 52)

COMMUNICATION POLICY

- Official announcements will be made on the course website, and Blackboard shall be used for communication regarding this course.
- Email the instructor by only using the UC account, i.e., from the "mail.uc.edu" or "uc.edu" domain. Please make sure to include "EECE5132 or EECE6032" in the subject of the email.

ACADEMIC INTEGRITY

The University Rules, including the Student Code of Conduct, and other documented policies of the department, college, and university related to academic integrity will be enforced. Any violation of these regulations, including acts of plagiarism or cheating, will be dealt with on an individual basis according to the severity of the misconduct. The Student Code of Conduct document can be accessed at: http://www.uc.edu/conduct/Code_of_Conduct.html.

ATTENDANCE, ADD/DROP/WITHDRAW, AND OTHER POLICIES

- Class attendance is ***mandatory***. Since this course is online, it will be determined by your participation each week in activities in the different modules. Class participation is worth 10% of the total grade. In-class exercises are used as a means to track attendance. Other means will be used during the lectures.

- Students who are auditing this course must attend at least 75% of the classes to avoid a grade of F.
- The ADD/DROP/WITHDRAW policy is at http://www.uc.edu/registrar/policies_and_procedures/add_drop_withdrawal.html.
- Students are responsible for reviewing this policy and complying.
- Information on Mental Health Services and Title IX can be found on the course website in Blackboard.
- UC policies and procedures can be found at http://www.uc.edu/registrar/policies_and_procedures.html.

SPECIAL NEEDS STATEMENT

If you have any special needs related to your participation in this course, including identified visual impairment, hearing impairment, physical impairment, communication disorder, and/or specific learning disability that may influence your performance in this course, you shall meet with the instructor in the beginning of the semester to arrange for reasonable provisions to ensure an equitable opportunity to meet all the requirements of this course. At the discretion of the instructor, some accommodations may require prior approval by Disability Services (http://www.uc.edu/aess/disability.html).

Background Knowledge:

- Reviewing standard textbooks on mathematics and statistics covering the following topics would be useful if you are unfamiliar with some of them:
- Basic concepts of relations, algebra, and set theory: Used throughout the book, and especially in the following:
    - Sets, subsets, partitions, basic types of relations, and equivalence classes in
    - Use of algebraic equations to define boundaries.
    - Precedence and dependency relations for control-flow and data-flow testing.

  - Cause-effect relations for hazard analysis and safety assurance, and for defect analysis.
- Logic, particularly Boolean logic, and related formalisms: Used throughout the book, and especially in the following:
  - Boolean logic for predicate and decision testing.
  - Mathematical logic and formalisms for formal verification of program correctness.
- Some basic concepts of graph theory: Used throughout the book, and especially in the following:
  - Decision trees for operational profiles used in statistical testing.
  - Graph elements for finite-state machines (FSMs) and related testing
  - Flow-chart like situations for control-flow testing.
  - Data dependency graphs (a tree-structured graph) for data-flow testing.
  - Trees in fault-tree analysis and event-tree analysis for hazard analysis and safety assurance.
  - Tree-based models for risk identification in Chapter 21 and for reliability analysis.
- Basic concepts of probability and statistics: Particularly important to the following topics:
  - Usage-based testing.
  - Defect classification and distribution analysis.
- Basic concepts of statistical analysis and modeling:
  - General analysis and modeling techniques.
  - Various specific types of analyses for risk identification.
  - Stochastic process and analysis for software reliability modeling.

Course Outline

Module #1 Week of August 27th	Lecture: What is Software Quality? Quality Assurance and Quality Engineering (source Tian chapters 2,3, and 5) 2 What Is Software Quality? 2.1 Quality: Perspectives and Expectations 2.2 Quality Frameworks and ISO-9126 2.3 Correctness and Defects: Definitions, Properties, and Measurements 2.4 A Historical Perspective of Quality 2.5 So, What Is Software Quality? Problems 3 Quality Assurance 3.1 Classification: QA as Dealing with Defects 3.2 Defect Prevention 3.2.1 Education and training 3.2.2 Formal method 3.2.3 Other defect prevention techniques 3.3 Defect Reduction 3.3.1 Inspection: Direct fault detection and removal 3.3.2 Testing: Failure observation and fault removal 3.3.3 Other techniques and risk identification 3.4 Defect Containment 3.4.1 Software fault tolerance 3.4.2 Safety assurance and defect containment 3.5 Concluding Remarks Problems 5 Quality Engineering 5.1 Quality Engineering: Activities and Process 5.2 Quality Planning: Goal Setting and Strategy Formation 5.3 Quality Assessment and Improvement 5.4 Quality Engineering in Software Processes 5.5 Concluding Remarks Problems Gregory & Crispin Chapter 1: How Agile Testing Has Evolved
	Quiz 1 Discussion Question Postings

Module #2 Week of September 3rd	Lecture: Ammann and Offutt: 1. Why Do We Test Software? - When Software Goes Bad - Goals of Testing Software 2. Model-Driven Test Design - Software Testing Foundations - Software Testing Activities - Testing Levels Based on Software Activity - Coverage Criteria - Model-Driven Test Design - Test Design - Test Automation - Test Execution - Test Evaluation - Test Personnel and Abstraction - Why MDTD Matters - Bibliographic Notes Gregory and Crispin: The Importance of Organizational Culture *(Gregory & Cripsin) Chapter 2* - Investing Time - The Importance of a Learning Culture - Fostering a Learning Culture - Transparency and Feedback Loops - Educating the Organization - Managing Testers *Chapter 3* - Roles and Competencies (Gregory & Crispin) Chapter 3 - Competencies versus Roles - T-Shaped Skill Set - Generalizing Specialists - Hiring the Right People - Onboarding Testers
	Quiz 2 Discussion Question Postings

Module #3 Week of September 10th	Lecture: Amman & Offutt: 3. Test Automation - Software Testability - Components of a Test Case - A Test Automation Framework o The JUnit Test Framework o Data-Driven Tests o Adding Parameters to Unit Tests o JUnit from the Command Line - Beyond Test Automation Gregory & Crispin: - Thinking Skills for Testing *Chapter 4* - Facilitating - Solving Problems - Giving and Receiving Feedback - Learning the Business Domain - Coaching and Listening Skills - Thinking Differently - Organizing - Collaborating Technical Awareness *Chapter 5* - Guiding Development with Examples - Automation and Coding Skills - General Technical Skills - Development Environments - Test Environments - Continuous Integration and Source Code Control - Systems - Testing Quality Attributes - Test Design Techniques
	Quiz 3 Discussion Question Postings

<table>
<tr><td rowspan="2">Module #4
Week of
September 17th</td><td>Lecture: Amman & Offutt:
4. Putting Testing First
- Taming the Cost-of-Change Curve
o Is the Curve Really Tamed?
- The Test Harness as Guardian
o Continuous Integration
o System Tests in Agile Methods
o Adding Tests to Legacy Systems
o Weaknesses in Agile Methods for Testing

Gregory & Crispin:
How to Learn Chapter 6
- Learning Styles
- Learning Resources
- Conferences, Courses, Meet-ups, and Collaborating
- Publications, Podcasts, and Online Communities
- Time for Learning
- Helping Others Learn
Levels of Precision for Planning Chapter 7
- Different Points of View
- Product Release Level
- Feature Level
- Story Level
- Task Level
- Planning for Regression Testing
- Visualize What You Are Testing</td></tr>
<tr><td>Quiz 4
Discussion Question Postings</td></tr>
</table>

Module #5 Week of September 24th	Lecture: Amman & Offutt: 5. Criteria Based Test Design - Coverage Criteria Defined - Infeasibility and Subsumption - Advantages of Using Coverage Criteria - Next Up Gregory & Crispin: Using Models to Help Plan *Chapter 8* - Agile Testing Quadrants - Planning for Quadrant 1 Testing - Planning for Quadrant 2 Testing - Planning for Quadrant 3 Testing - Planning for Quadrant 4 Testing - Challenging the Quadrants - Using Other Influences for Planning - Planning for Test Automation Are we building the right thing? *Chapter 9* - Start with "Why" - Tools for Customer Engagement - Impact Mapping - Story Mapping - The 7 Product Dimensions - More Tools or Techniques for Exploring Early - Invest to Build the Right Thing
	Quiz 5 Discussion Question Postings

Module #6
Week of
October 1st
	Lecture: Amman & Offutt:
	6. Input Space Partitioning
	- Input Domain Modeling
	  ○ Interface-Based Input Domain Modeling
	  ○ Functionality-Based Input Domain Modeling
	  ○ Designing Characteristics
	  ○ hoosing Blocks and Values
	  ○ Checking the Input Domain Model
	- Combination Strategies Criteria
	- Handling Constraints Among Characteristics
	- Extended Example: Deriving an IDM from JavaDoc
	  ○ Tasks in Designing IDM-Based Tests
	  ○ Designing IDM-Based Tests for Iterator
	
	Gregory & Crispin:
	The expanding Testers Mindset: Is this my Job? *Chapter 10*
	- Whose Job Is This Anyway?
	- Business Analysis Skills
	- UX Design Skills
	- Documentation Skills
	- Take the Initiative
	Getting Examples *(Gregory & Crispin) Chapter 11*
	The Power of Using Examples
	- Guiding Development with Examples
	- ATDD
	- BDD
	- SBE
	- Where to Get Examples
	- Benefits of Using Examples
	- Potential Pitfalls of Using Examples
	- Getting Bogged Down in the Details
	- Lacking Buy-in
	- Too Many Regression Tests
	- Not Enough Is Known Yet
	- The Mechanics of Using Examples to Guide Coding

	Quiz 6 Discussion Question Postings

Module #7
Week of
October 8th
	Lecture: Amman & Offutt:
	7. Graph Coverage
	- Overview
	- Graph Coverage Criteria
	  o Structural Coverage Criteria
	  o Touring, Sidetrips, and Detours
	  o Data Flow Criteria
	  o Subsumption Relationships Among Graph Coverage Criteria
	- Graph Coverage for Source Code
	  o Structural Graph Coverage for Source Code
	  o Data Flow Graph Coverage for Source Code
	- Graph Coverage for Design Elements
	  o Structural Graph Coverage for Design Elements
	  o Data Flow Graph Coverage for Design Elements
	- Graph Coverage for Specifications
	  o Testing Sequencing Constraints
	  o Testing State Behavior of Software
	- Graph Coverage for Use Cases
	  o Use Case Scenarios
	
	Gregory & Crispin:
	Exploratory Testing *Chapter 12*
	- Creating Test Charters
	- Generating Test Charter Ideas
	- Exploring with Personas
	- Exploring with Tours
	- Other Ideas
	- Managing Test Charters
	- Session-Based Test Management
	- Thread-Based Test Management
	- Exploring in Groups
	- Recording Results for Exploratory Test Sessions
	- Where Exploratory Testing Fits into Agile Testing
	Other Types of Testing *(Gregory & Crispin) Chapter 13*
	- So Many Testing Needs
	- Concurrency Testing
	- Internationalization and Localization
	- Regression Testing Challenges
	- User Acceptance Testing

	- A/B Testing - User Experience Testing
	Quiz 7 Discussion Question Postings

Module #8 Week of October 15th	Lecture: Amman & Offutt: 8. Logic Coverage - Semantic Logic Coverage Criteria (Active) o Simple Logic Expression Coverage Criteria o Active Clause Coverage o Inactive Clause Coverage o Infeasibility and Subsumption o Making a Clause Determine a Predicate o Finding Satisfying Values Gregory & Crispin: Technical Debt in Testing *(Gregory & Crispin) Chapter 14* - Make It Visible - Work on the Biggest Problem—and Get the Whole Team Involved Pyramids of Automation *(Gregory & Crispin) Chapter 15* - The Original Pyramid - Alternate Forms of the Pyramid - The Dangers of Putting Off Test Automation - Using the Pyramid to Show Different Dimensions
	Quiz 8 Discussion Question Postings

Module #9 Week of October 22nd	**Mid-Term**

Module #10 Week of October 29th	Lecture: Amman & Offutt: 8. Logic Coverage - Syntactic Logic Coverage Criteria (DNF) o Implicant Coverage o Minimal DNF o The MUMCUT Coverage Criterion o Karnaugh Maps Gregory & Crispin: Test Automation Design Patterns and Approaches *Chapter 16* - Involve the Whole Team - Starting Off Right - Design Principles and Patterns - Testing through the API (at the Service Level) - Testing through the UI - Test Maintenance Selecting Test Automation Solutions *Chapter 17* - Solutions for Teams in Transition - Meeting New Automation Challenges with the Whole Team - Achieving Team Consensus for Automation Solutions - How Much Automation Is Enough? - Collaborative Solutions for Choosing Tools - Scaling Automation to Large Organizations - Other Automation Considerations
	Quiz 9 Discussion Question Postings

Module #11 Week of November 5th	Lecture: Amman & Offutt: 8. Logic Coverage Structural Logic Coverage of Programs - Satisfying Predicate Coverage - Satisfying Clause Coverage - Satisfying Active Clause Coverage - Predicate Transformation Issues - Side Effects in Predicates Specification-Based Logic Coverage Logic Coverage of Finite State Machines Gregory & Crispin: Agile Testing in the Enterprise *Chapter 18* - What Do We Mean by "Enterprise"? - "Scaling" Agile Testing - Dealing with Organizational Controls - Coordinating Multiple Teams - A System Test Team and Environment - Consistent Tooling - Coordination through CI - Version Control Approaches - Test Coverage - Managing Dependencies - Working with Third Parties - Involving Customers in Large Organizations - Advantages of Reaching Out beyond the Delivery Team Agile Testing on Distributed Teams *Chapter 19* - Why Not Co-locate? - Common Challenges - Cultural Issues - Language - Time Zones - Dependencies - Planning - Strategies for Coping - Integrating Teams - Communication and Collaboration - Collaborating through Tests - Offshore Testing - Tool Ideas for Distributed Teams

	- Communication Tools
	Quiz 10 Discussion Question Postings

Module #12 Week of November 12th	Lecture: Amman & Offutt: 9. Syntax-Based Testing - Syntax-Based Coverage Criteria o Grammar-Based Coverage Criteria o Mutation Testing - Program-Based Grammars o BNF Grammars for Compilers o Program-Based Mutation - Integration and Object-Oriented Testing o BNF Integration Testing o Integration Mutation - Specification-Based Grammars o BNF Grammars o Specification-Based Mutation - Input Space Grammars o BNF Grammars o Mutating Input Grammars Gregory & Crispin: Agile Testing for Mobile and Embedded Systems *Chapter 20* - Similar, Yet Different - Testing Is Critical - Agile Approaches Agile Testing in Regulated Environments *Chapter 21* - The "Lack of Documentation" Myth - Agile and Compliance
	Quiz 11 Discussion Question Postings

Module #13 Week of November 19th	Lecture: Amman & Offutt: 10. Managing the Test Process - Overview - Requirements Analysis and Specification - System and Software Design - Intermediate Design - Detailed Design - Implementation - Integration - System Deployment - Operation and Maintenance - Implementing the Test Process Gregory & Crispin: Testing for Data Warehouses and Business Intelligence Systems *Chapter 22* - What Is Unique about Testing BI/DW? - Using Agile Principles - Data—the Critical Asset - Big Data Testing and DevOps *Chapter 23* - A Short Introduction to DevOps - DevOps and Quality - How Testers Add DevOps Value - Summary
	Quiz 12 Discussion Question Postings

Module #14 Week of November 26th	Lecture: Amman & Offutt: 11. Writing Test Plans - Level Test Plan Example Template 12. Test Implementation - Integration Order - Test Doubles o Stubs and Mocks: Variations of Test Doubles o Using Test Doubles to Replace Components 13. Regression Testing for Evolving Software 14. Writing Effective Test Oracles - What Should Be Checked? - Determining Correct Values o Specification-Based Direct Verification of Outputs o Redundant Computations o Consistency Checks o Metamorphic Testing Gregory & Crispin: Visualize Your Testing *Chapter 24* - Communicating the Importance of Testing - Visualize for Continuous Improvement - Visibility into Tests and Test Results - Summary Putting It All Together *Chapter 25* - Confidence-Building Practices - Create a Shared Vision - Summary
	Quiz 13 Discussion Question Postings

Module #15 Week of December 3rd	- Performance Testing and Quality Assurance
	Quiz 14 Discussion Question Postings

Module #16 December 10th	**Final Exam**

I, the undersigned, acknowledge that I have received a copy of the course syllabus for this class. I have read the syllabus for this course, and I verify that I understand the syllabus. I have had all of my questions answered about this document. I understand that the course syllabus will be a guideline for the class and I agree to abide by the rules and regulations for the course outlined in the syllabus. I also agree that not only will I be required to follow the syllabus, but I must also follow all college rules and regulations while enrolled in this course.

I agree to comply with any and all due dates for reading, Readings assignments, projects, and any other assignments outlined in the syllabus. I agree to maintain a professional standard in this course, which includes adherence to the guidelines regarding tardiness and attendance.

Date: 03/21/96 UCID #: M12998592

Student Name Suthirr Preethum Balasubramanian

Home # Cell # 5133285253

Student Printed Name: Suthirr Preethum Balasubramanian

Student Signature: B. Suthirr