Oregon State University

CS362, Software Engineering II

Instructor Name: Wendy Roberts
Instructor Email: roberwen@oregonstate.edu

OSU catalog course description, including pre-requisites

Introduction to the "back end" of the software engineering lifecycle implementation; verification and validation; debugging; maintenance.

Enforced Prerequisites**:** CS 261 [C]
Other: Experience with object-oriented programming and data structures (e.g., CS 161, CS 162, CS 261).
Note: CS 361 is recommended but not required.

Course Credits

This course combines approximately 120 hours of instruction, online activities, and assignments for 4 credits.

Class Expectations

You can expect to spend 8 to 10 hours a week on this course. Many of the topics we cover could be an entire course by themselves. The objective of this course is to give you an overview of each topic. You will test your understanding through assignments, projects, and exams.

Communication

The best way to contact me or your assigned TA is via email. This method will get the fastest response time.

We use Piazza and Slack in this course to communicate between students, TA's, and the Instructor. Please feel free to post all course-related questions to Slack or Piazza so the whole class may benefit from the conversation.

Please send a message (top right of Canvas) to your instructor/TA for matters of a personal nature. We will reply to course-related questions, Inbox messages, and email messages within 24-48 hours.

Course Content:

- Software verification and validation, including test plan development, test design and construction, test automation, white-box, black-box, regression testing techniques, and software inspections.
- Software maintenance including types of maintenance, configuration management, the use of configuration control tools, the use of automated product build tools, fault localization strategies, and the use of automated debugging tools

Measurable Student Learning Outcomes

At the completion of the course, students will be able to

- Apply automated tools such as make and Git in a realistic setting
- Describe the cost-benefit trade-offs inherent in the use of automated tools for building software and configuration management
- Describe several techniques for validating and measuring the quality of software
- Apply testing techniques, including black box and white box techniques, automatic testing activities, and regression testing
- Use appropriate techniques and tools, including a debugger, to locate program faults
- Describe several types of maintenance processes associated with correcting and enhancing software systems
- Participate effectively in a software inspection
- Participate effectively in a team environment

Course Calendar

Note: The schedule may be adjusted if it becomes apparent that more/less time is needed for some of the topics. Additional tasks may be assigned and graded

Week	Topic
1	**Version control systems** • Forks, Pull requests, Branching • Git, GitHub
2	**Software Testing Overview** • Thinking about Testing • Maintenance and Source Control • Builds & Static Analysis • Introduction to Software Testing: Kinds of Testing
3	**Code Coverage** • Coverage Metrics I • Coverage Metrics II
4	**Software Testing** • Lessons Learned in Software Testing: Reporting Bugs • Lessons Learned in Software Testing: Planning and Strategy • Lessons Learned in Software Testing: The Testing Role • Lessons Learned in Software Testing: Thinking Like a Tester • Lessons Learned in Software Testing: Testing Techniques
5	**Random Testing** • Random Testing Concepts • How to Write a Simple Random Tester
6	• **Introduction to Debugging** • Quick Intro to Debuggers • Introduction to open source project
7	**Causality and Localization** • Causality and Localization I • Causality and Localization II
8	**Debugging** • Agans' Rules for Debugging
9	**Software Inspections** • Integration Testing
10	**Regression Testing** • Introduction to search based software Testing (SBST) • Introduction to Symbolic Execution Testing

Evaluation of Student Performance

Scores for quizzes, assignments, and exams will be posted on Canvas as they are graded. **No late submission accepted without prior approval! Prior approval does not mean an hour or two before the time the assignment is due. You should never wait until the day a coding assignment is due to start on it.**

Assignments (35%) + Quizzes(15%) - 50%

- There are five assignments to be completed over the course of this class.
- Assignments include a mixture of written documents and code submissions.
- There are 7 quizzes. You are expected to take all quizzes.
- If you have a problem with an assignment grade, you should first contact your TA and if you do not resolve the issue please contact me.

Exams - 30% (15% each exam)

- There is one midterm exam for this course and one final exam.
- Each exam is given after completing 10-12 units.
- These exams are designed to take two hours each.
- These exams are open note, open internet, essay exams. Please do not consult your classmates or share any information with other students. These exams are NOT PROCTORED.

Final Project - 20%

- There is a final project designed to check for your cumulative understanding, which includes some of the work for assignments.

- Part-A (20%)
- Part-B (80%)

Grading Scale (round up if between whole numbers):

Grade	Average
A	Greater than 93
A-	90 - 92
B+	87 - 89
B	83 - 86
B-	80 - 82
C+	77 - 79
C	73 - 76
C-	70 - 72
D+	67 - 69
D	63 - 66
D-	60 - 62
F	less than 60

* REMINDER: A passing grade for core classes in CS is a C or above. A C-, 72 or below, is not a passing grade for CS majors.

Expectations for Student Conduct

Student conduct is governed by the university's policies, as explained in the Student Conduct Code. Students are expected to comply with all regulations pertaining to academic honesty. For further information, visit or contact the office of Student Conduct and Mediation at 541-737-3656.

Academic or Scholarly Dishonesty is defined as an act of deception in which a Student seeks to claim credit for the work or effort of another person or uses unauthorized materials or fabricated information in any academic work or research, either through the Student's own efforts or the efforts of another.

So please do your own work, be careful not to share code on Piazza or Slack, and don't share your work with your classmates unless you are working on a group project.

Resources (Optional but Recommended)

- **Lessons Learned in Software Testing**, by Cem Kaner, James Bach, and Bret Pettichord;
- **Debugging** by David J. Agans

Statement Regarding Students with Disabilities

Accommodations for students with disabilities are determined and approved by Disability Access Services (DAS). If you, as a student, believe you are eligible for accommodations but have not obtained approval please contact DAS immediately at 541-737-4098 or at http://ds.oregonstate.edu. DAS notifies students and faculty members of approved academic accommodations and coordinates implementation of those accommodations. While not required, students and faculty members are encouraged to discuss details of the implementation of individual accommodations.