CSSE 490 - Debugging: From Art to Science

General Course Syllabus

Course Description

Introduction to systematic debugging. Systematic approaches to solving problems covered in this class includes: defect classification, search space, instrumentation, tool usage, engineering/ scientific process. Specific topics include failure reproduction, bug tracking and software debugging tools, problem isolation and size reduction. There are several in-class debugging "labs" and programming assignments and several larger out-of-class debugging projects with a final project to reproduce and fix a bug in open-source software projects.

Course Prerequisites

- Required: CSSE 230; CSSE 132
- Good to have: CSSE 332; CSSE 333

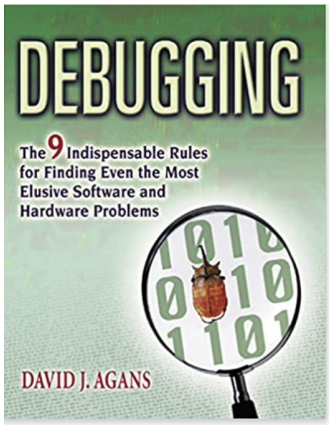
Technical Prerequisites

Should be able to write and understand programs in C, Java and Python; must also have a fair ability to read and understand assembly

Required Resources

Book: **Debugging** by David J. Agans

AMACOM, 2002



<u>In addition to the book above, the following tools are required for the assignments</u> (see the Getting Started section of Moodle page):

- Visual Studio Code or CLion with C and C++ support (you may use the IDE of your choice, but only information on importing etc. will be provided for the listed IDEs)
- Java (any version is fine, recommend 8 or newer) and Eclipse or IntelliJ
- Access to a Linux installation with GDB and RR (this has been done for you with a VM, to access, see Getting Started section of Moodle page)

EXTREMELY Recommended Resources

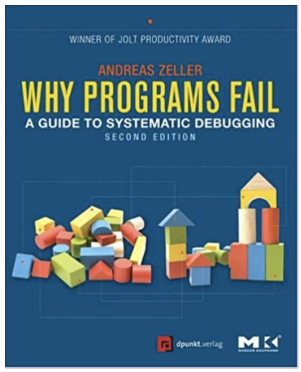
Book: https://www.debuggingbook.org

(free online book)

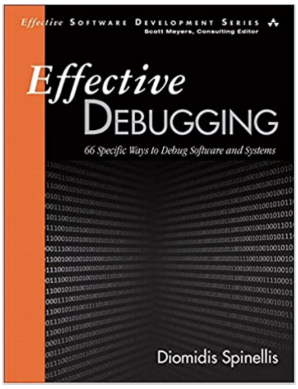
Online Resource: https://github.com/jlevy/the-art-of-command-line

Book: **Why Programs Fail, 2nd edition** by Andreas Zeller

Morgan Kaufman, 2009

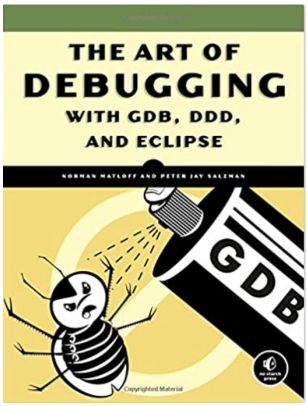


Book: **Effective Debugging** by Diomidis Spinellis
Addison-Wesley, 2017



Book: **The Art of Debugging**, by Norman Matloff and Peter Jay Salzman

No Starch Press, 2008



Course Objectives

The following is what you will learn through this course.

- Describe how software defects come to exist
- Classify bugs based on reproducibility and determinism
- Examine and understand large software projects
- Track bugs efficiently and with all necessary detail
    - appropriately classify bug severity and priority
    - summarize problem reports
    - tracking software
    - “Silver Platter Principle”
    - identify duplicate bugs with tracking
    - reduce input to smallest possible to recreate failure
- Describe process needed to reproduce/force software failures
    - write unit tests for any cases possible
    - utilize further automation whenever possible
- Apply the Scientific Method to bug search
    - use instrumentation to gather data
    - eliminate code that works from search
    - eliminate irrelevant variables from search
    - hypothesize the bug cause
    - formulate and apply correction to test hypothesis
    - find bugs systematically
    - determine “infection chain” and follow up to defect
    - automate process when possible
- Efficiently utilize appropriate debugging tool to find defects
    - find logic bugs with program state
    - find regressions with Continuous Integration

        - static analyzers
        - debuggers
            - GDB
            - RR
            - IDE and/or language specific
        - statistical analyzers static and dynamic
    - Apply cause and effect rules to repair software defect and prove fix corrects failed state
    - Judge and plan appropriate design techniques and patterns to improve/simplify future debugging efforts in software under development
        - refactor as necessary
        - plan for automation and replay of program runs
        - model-view-controller design pattern

Out of Class Assignments

Reading Assignments

There will be assigned readings each week from the required text. The reading should be completed **before** the corresponding course section where there will be a discussion on the reading

Projects (debugging @home)

There will be several debugging projects (large and small). You will be required to track/log every bug in this course. You will be required to keep a debugging logbook throughout this course. You will be required to turn in the relevant portions of your logbook with every project/ assignment along with corresponding code changes.

Bug Tracking

For every bug you encounter in this course, you will be required to "file" or "track" the bug (after that material has been covered). The requested information will be similar to other bug-tracking databases (see Jira or Bugzilla). This exercise will be completed on Moodle. You can see "Filing a Bug" in the Moodle page for more information.

Debugging Logbook

Every great "debugger" has a logbook of bugs (whether stored physically or mentally). You will be required to keep a logbook of all the bugs (we will discuss this in class). You can keep this on paper or digitally (I recommend digitally). For every project, you will be asked to turn in the relevant parts of your log after you have completed the exercise. You can see the "Keeping a Logbook" page for more information.

In-class Assignments

Discussions of readings

A portion of a class each week will be discussing the content of the readings. You must participate in the weekly discussions to earn points for this assignment.

Quizzes

Accompanying *most* class days will be a quiz to be completed on Moodle. These quizzes and their due dates are set up so that you can work through them in whatever best fits your learning style. The instructor recommends attempting to complete as much of a quiz during the corresponding lecture as possible (answers to all/most questions will be discussed in the lecture). However, if this does not fit your learning preference, you may complete the quizzes later, outside of class.

All quizzes in a given week are due by the following Sunday at 11:59 PM. For example, in Week 1 (e.g., March 4-8), you have 4 quizzes. These quizzes are due (must be *completed*) by Sunday March 10 at 11:59 PM. The quizzes will then close. Take note that 15% of your grade comes from these quizzes and these are not generally difficult questions. **Quizzes cannot be made up later, so be careful to complete them!** NOTE: Most weeks, Day 4 will have a "Discussion" quiz. Discussion quizzes should be filled out in class as the discussion takes place as much as possible. The Discussion quizzes are also still due the following Sunday at 11:59 PM.

Labs

There are several labs that must be completed in class. Lab assignments will be available on the Moodle page (and in your VMs). Completed labs are submitted via Moodle.

Final Project

In the last weeks of the course, there will be a large debugging project done by teams of 2. For this project, you will be asked to find an open/current bug in open-source software. A debugging log and the correcting code changes must be submitted for the project. To view some open-source projects, you can go to: https://www.openhub.net; https://en.wikipedia.org/wiki/List_of_free_and_open-source_software_packages; https://rocket.chat/blog/open-source-projects

Incentive

In this course, you will be asked to go above and beyond the material assigned. By the end of the quarter, you will need to have attained 100 points of Incentive work (work not assigned directly) to receive full credit for this portion of your grade. Note, however, that there is no cap on on the number of points you can receive.

Please read the Incentive Details page to get all the information. It is very important you fully read this page!

Note: Particularly for this course, you could also learn a new tool for debugging (that is not a required one in the course) and demonstrate what you learned.

There might even be extra bugs in assignments and if you fix those too, there can be incentive in it for you. The possibilities are nearly endless!

Bug of the Day

Starting in Week 2, we will have a "Bug of the Day" presentation by one student in the first 5 minutes of every class. Every student will eventually present a bug of the day in class. For every bug of the day you present, you will be awarded 10 incentive points. The more current the bug is, the better. But older bugs can also be presented when they are more "interesting." No two students may present the same one (bugs in the wild, they are a' plentiful, so this won't be difficult).

Final Oral Exam

During the final week of the course or during finals week, each student will meet with me individually for a small oral examination. The intent is to have a discussion of course topics, so there will be questions asking about the higher level concepts of the class. Any of the material covered in the course may be fair play for the discussion, but fine details will not be required. Simple questions such as "Why do we use X?" and "How can Y help in a particular situation?" are likely to be covered.

Grading

Assignment	Weight
Readings, Quizzes and Discussions	15%
Labs and (HW) Projects	45%
Incentive	10%
Final Project	20%
Final Oral Exam	10%

Collaboration

Collaboration is encouraged on quizzes and, of course, discussions. For homework, projects and labs, each assignment will explicitly state whether it is an individual or group assignment (if it is not explicitly stated, you should assume by default that the assignment is an individual one). Collaboration is prohibited on every assignment that explicitly states it is an individual assignment. Most group assignments will be paired assignments (no more than two usually). You should consider working with different students in the course. When you collaborate, you must **properly credit your collaborators and clearly indicate the extent of the collaboration**. In all cases, each individual is responsible for understanding and writing out the entire solution and turning in the relevant materials to Moodle. For example, on homework, this means that once a group solution has been achieved, each collaborator must rework the problem and write up the solution independently. Each participant is responsible for his/her own logbook entries as well, which must be turned in. Logbooks should ALWAYS be kept in the owner's own words. **Copying assignments, solutions or logbook entries is not collaboration.**

In general, any actions that reduces your or your peer's ability or opportunity to learn can be considered academic misconduct. You are here to learn!

ChatGPT

The use of ChatGPT in this course should remain highly/extremely limited. You may NOT use it for assignments (homework and/or labs). You may use ChatGPT in a limited capacity for research purposes. However, you should NEVER directly copy and paste any content directly found when using the tool. Even if using ChatGPT for Incentive credit, directly copying and pasting any content will be considered Academic Misconduct. If it is discovered that you improperly used ChatGPT, you will receive penalty grading and the incident will be reported to Student Affairs.

When using ChatGPT in the proper setting, you MUST always cite that you used it and in what capacity. Failing to cite your usage of ChatGPT may result in penalty grading and reporting the incident to Student Affairs.

Concerns or Feedback

If you have any concerns about any course matters, you should find someone to contact. You can email your instructor, speak to your instructor, or leave anonymous feedback on the course page. You can also contact the CSSE department head, Student Affairs, or another Rose employee you feel comfortable with.