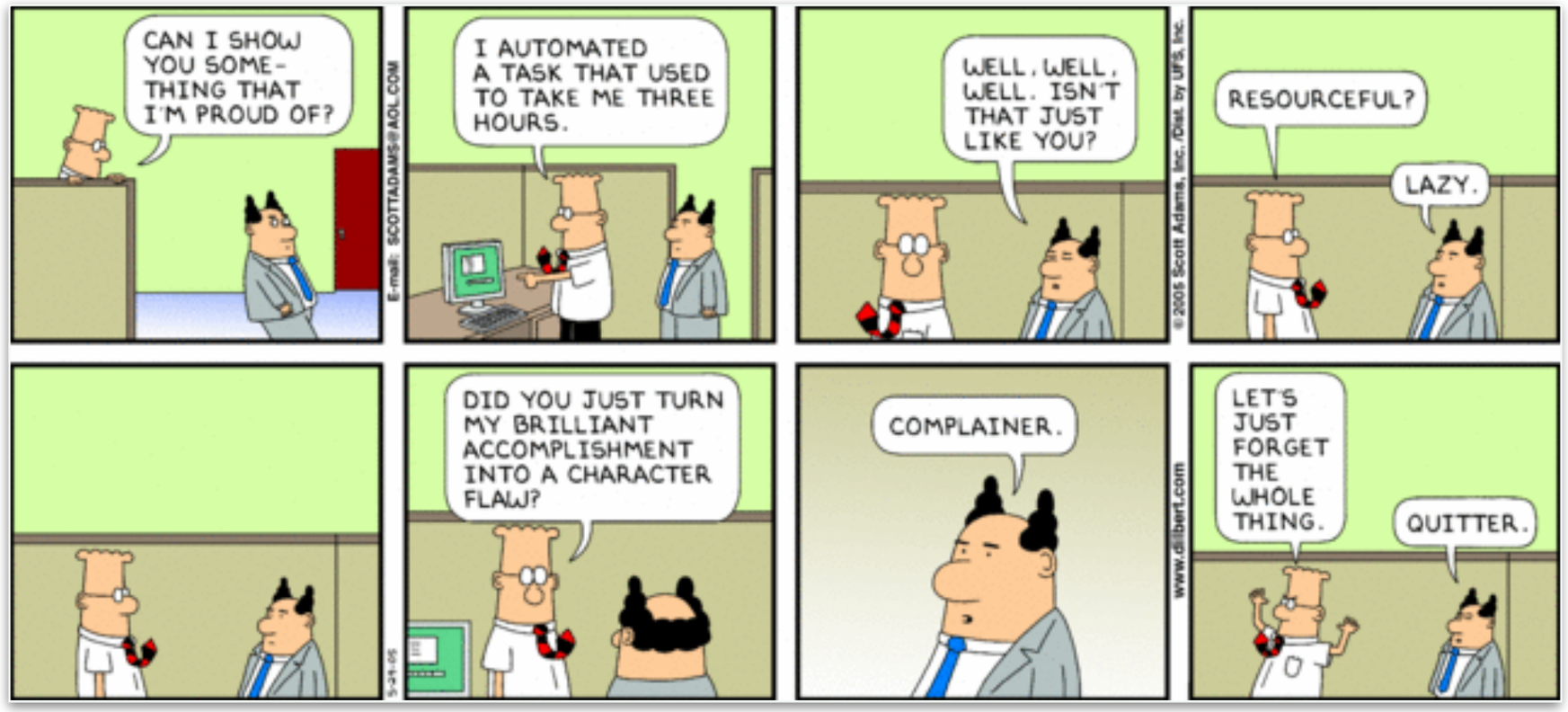

CS453: Automated Software Testing

Admins & Introduction

CS453

- Lecturer: Shin Yoo (shin.yoo@kaist.ac.uk) from COINSE (http://coinse.kaist.ac.kr)

- We focus on various concepts and techniques in automated software testing and debugging.

- With an emphasis on learning how to do meta programming.

Class Communication

- We are trying out Slack workspace as the class communication channel this semester.

- All class announcements, as well as Q&A, will take place on a dedicated workspace: **cs453-2021-kaist.slack.com**.

- **You must join!** It is strongly recommended that you install either a desktop or a mobile client, to get notifications. Invitation link has been sent in an email from KLMS.

- When you join, use “[Full Name]([STUDENT ID #])” as your username, e.g., “Shin Yoo (20201234)”.

    - #questions for questions, #teams for finding teammates, #random for jokes and memes, #general for anything else

Learning Objectives

- Know fundamental concepts, principles, activities and techniques for software validation and be able to justify appropriate use of techniques for a particular project

- Understand a range of approaches to testing that can be applied to software systems and can apply them appropriately

- Appreciate the limitations of the current tools and have insights in ongoing research topics to overcome them

Schedule

- Please refer to http://coinse.kaist.ac.kr/teaching/2021/cs453

- Spring 2021 courses will be online.
  - All my conference travels are canceled, so more time :)
  - Course has been re-designed. Major changes are:
    - Final exam, but no mid-term
    - No paper presentations

Grading

- 40% Assignments
- 30% Final Exam (2 hour written exam, open book)
- 30% Course Project

Requirements

- **Strong programming skills**: you are required to actively contribute to group and individual project, which involves serious implementation. There will be also a number of hands-on sessions where we will program together during the class.

- **Unix/Linux-savvy**: you should be familiar with the usual build tools and Unix/Linux command line environments.

- **Git-aware**: you will be required to submit a github repository as part of your project deliverable. Plus, all coursework will be handled on GitHub Classroom.

Requirements

- Ideally, CS350 Introduction to Software Engineering.
    - Lifecycle activities: requirements engineering, design, modelling, implementation, integration, testing, evolution
    - Experience of software design and programming (you will do some)
- Also, a computer, as we will do some live coding and hands-on activities.

Assignments

- All four assignments will be handled by GitHub Classroom:
  - You will be given public test cases that you can freely execute
  - There will be private test cases that we will add before the final grading
- All assignments are to be done individually.

Assignments

- Assignment 0: onboarding to GitHub Classroom - **due March 10th (no marks)**

- Assignment 1: Introduction to Metaprogramming - **due March 22nd (5%)**

- Assignment 2: Coverage Profiler - **due April 12th (10% + 5% optional bonus)**

- Assignment 3: Mutation Testing - **due May 3rd (10% + 3% optional bonus)**

- Assignment 4: Hierarchical Delta Debugging - **due May 24th (10%)**

Projects and Teams

- Project: implement, and evaluate, an automated software testing technique and report the findings

- Submission and participation: I will receive a GitHub repository as a deliverable, and I expect to see everyone contributing to the project implementation.

  - Don’t say “we worked on a single machine, which is why all commits are from X”

  - Don’t say “I just worked on documentation and presentation”

- Until the number of remaining people makes it necessary, I intend to only accept teams of four people.

Projects and Teams

- We will start talking about teams in April - after those who want to drop actually drop :)
- By then, I also hope that you will have some ideas about what to implement, and what it takes.
    - Discuss your ideas with me even before April if you have some

Literature Review Sessions

- They sort of replace the paper presentation slots.
- Main idea is that I will guide you through key papers…
- …but I am trying to think of a way to make this more interactive.

Being Online

- Some things will not be ideal, but let's be patient and optimistic :)

- I plan to do real time streaming lectures, but we have no idea whether the infrastructure will cope with the load. If it becomes too unstable:

- I will also provide recordings of the lectures on KLMS.

- As a supplementary material, there are recorded video lectures that are subset of this course on KMOOC, but they're only in Korean (search for "소프트웨어 테스팅", it is done by me and Prof. Moonzoo Kim)

Being Online: Team Projects

- This won’t be easy but:
    - For forming teams, there will be a dedicated channel on Slack (#teams) - advertise yourself there
    - For the actual collaboration for team project, *working online remotely is how we do it outside in real world anyway* - this may be a good practice! Be smart and use many available tools.

References

- Strictly recommended for your reference: we *do not teach* these books and these books *do not contain answers* to this course.

  - Paul Ammann and Jeff Offutt. Introduction to Software Testing (2nd Ed.)

  - Andreas Zeller. Why Programs Fail (2nd Ed.)

  - Y. Jia and M. Harman. An analysis and survey of the development of mutation testing. IEEE transactions on software engineering, 37(5):649–678.

  - P. McMinn. Search-based software test data generation: A survey. Software Testing, Verification and Reliability, 14(2):105–156, June 2004.