University of Pittsburgh

CS 1632 Introduction

Instructor Introduction

Course Introduction

Wonsun Ahn
Department of Computer Science
School of Computing and Information

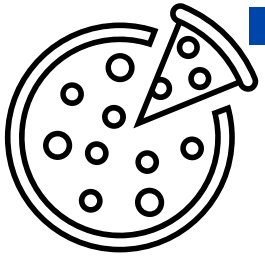
Instructor Introduction

My Technical Background

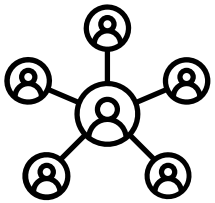
- Wonsun Ahn
  - First name is pronounced *one-sun* (if you can manage)
  - Or you can just call me Dr. Ahn (rhymes with *naan*)

- Relevant Experience
  - Bluebird Corporation (70-person startup company)
    - Manufactures industrial hand-held devices from top to bottom
    - Me: Built software stack based on Windows Embedded
  - IBM Research (thousands of people)
    - Does next-gen stuff like carbon nanotubes, quantum computers
    - Me: Built software simulators that model supercomputer designs

Zero-sum Thinking No Longer Works



- Zero-sum thinking (old way of thinking)
  - “If you get a larger slice of the pie, I get a smaller slice.”
  - Therefore, if you lose, I win (and vice versa)



- Increasingly, everything is connected
  - Pandemic: If my neighbors catch the virus, so will I
  - Environment: If disaster hits my neighbor, I feel the effects
  - Economy: Think of how the subprime mortgage crisis spread

!

- Zero-sum thinking no longer works
  - If you lose, I lose as well

Collaboration Beats Competition

- Truer in all spheres as the worlds gets more connected
- Collaboration is also happening in the IT industry
  - The *open-source* movement
  - Increasing importance of the software/hardware *ecosystem*
  - Increasing importance of the developer *community*
- Collaboration is also important for learning
  - During my undergrad years, what do I remember best?
  - Stuff that my classmates taught me
  - Stuff that I explained to my classmates

Supporting Collaborative Learning

- You will be working with a partner (on GitHub)
  - You will learn how to collaborate on a source repository
- You are a member of the CS 1632 Team
  - I encourage you to be on Teams at most times (I will too)
    - Recommend you install app on both laptop and cell phone
  - You can ask questions on the appropriate Teams channel
    - Either your classmate or your instructor will answer
  - You can also chat with any individual on the Team
    - “Manage Team” item in the “…” Team context menu

Supporting Collaborative Learning

- What to share and what not to share
  - You can freely discuss Exercises
    - Posting your code or solutions is totally fine
  - You ***cannot*** share code for Deliverables
    - Questions limited to understanding the parameters of the project
    - Once you fully understand the corresponding exercise, no need!
- Activity on Teams results in extra participation points
  - Asking questions and answering both count as activity!
  - Doesn't have to be questions. Random comments, observations, stuff you read online are welcome too.

Course Introduction

Structure of the Course

- (20% of grade) Two Midterms
- (70% of grade) Five projects
  - Manual Testing and Traceability Matrix
  - Unit Testing
  - Systems Testing a Web Application
  - Performance Testing
  - Comprehensive static & dynamic testing
- (10% of grade) Participation
  - Attendance, TopHat questions, Exercise submissions, Teams participation
- Class resources:
  - Canvas: announcements, Zoom meetings, recorded lectures
  - GitHub: syllabus, textbook PDF, lectures, exercises, deliverables
  - Tophat: in class recorded lecture questions
  - GradeScope: exercise / deliverable submission, grading and feedback
  - Microsoft Teams: Out-of-class communication

For More Details

- Please refer to the syllabus page:
  https://github.com/wonsunahn/CS1632_Fall2024/blob/main/syllabus.md

- Please follow the schedule page:
  https://github.com/wonsunahn/CS1632_Fall2024/blob/main/schedule.md

- This is a semi-flipped classroom
  - That means you will have to bring your laptops to class
  - If you do not own a laptop, please ask me for help

TODO

- Please check schedule.md page before every class.
  - I will also try to update the Canvas calendar whenever assignments are released
- Currently due on GradeScope:
  - Java Assessment Exercise
  - Partnership Contract
- TopHat, GradeScope, Panopto are accessible through Canvas

What QA techniques have you used?

What QA techniques do you want to learn?

What is Software Quality Assurance? What it's not…

- It's not something you've never done
- It's not optional
- It's not something you do after you built something
  - It's done throughout software development life cycle: requirements development, software design, writing code, integrating code, verification
- It's not finding every bug
  - It's about managing business risk from exposure to bugs
- It's not just testing
  - It's also about creating processes to correct problems
  - It's also about providing an independent view of the SW

What it is

All activities that ensure quality during software development

QA includes….

Unit testing, systems testing, acceptance testing, automated testing, requirements analysis, equivalence classes, white/grey/black box testing, verification, validation, combinatorial testing, performance testing, reliability testing, model checking, static analysis, linting, traceability matrices, defect reporting, test planning, TDD, fuzz testing, KPIs, software profiling, resource analysis, usability analysis, regression testing, smoke testing, security analysis, penetration testing....

It's an entire field of study!

Case Study: Boeing 737 MAX Crashes

Lion Air crash: Boeing 737 plane crashes in sea off Jakarta

29 October 2018 Share

ALEX DAVIES TRANSPORTATION 03.18.2019 02:47 PM

Crashed Ethiopian Air Jet Is Same Model as Lion Air Accident

An Ethiopian Airlines Boeing 737 MAX 8 crashed shortly after takeoff Sunday, evoking comparisons to an Indonesian incident in October.

Boeing & Aerospace | Business | Nation & World | Times Watchdog

Flawed analysis, failed oversight: How Boeing, FAA certified the suspect 737 MAX flight control system

March 17, 2019 at 6:00 am | *Updated March 21, 2019 at 9:46 am*

Case Study: Boeing 737 MAX Crashes

- How was Boeing 737 MAX different from previous 737 models?
- First Boeing 737 (737-100) was built in April, 1967
  - Had a low profile to ease loading/unloading of plane
    (They didn't have belt-loading baggage vehicles at that time)
- Boeing 737 MAX was built in December, 2018
  - Reused old design with larger engine for heavier load (to cut costs)
  - Engine did not fit under wing so had to bring it upwards and forwards

Case Study: Boeing 737 MAX Crashes

- New engine placement on 737 MAX led to worse aerodynamics
- Boeing did not want to retrain pilots (to cut costs)
- Boeing chose instead to write software to emulate an old 737
  - Make it "feel" like pilot was flying an old 737 instead of 737 MAX
  - Called *Maneuvering Characteristics Augmentation System* (MCAS)

- MCAS was the culprit that forced the planes into a nosedive
- MCAS had software quality issues at multiple levels
  - Requirements validation (used loopholes to skirt FAA oversight)
  - Robustness testing (skirted single-point-of-failure testing)
  - Defect reporting (catastrophic failures reported as merely hazardous)
- The root problem was that of corporate culture
  - Increased competition from Airbus since 2000 led to focus on cutting costs
  - First Boeing product to outsource critical flight software to 3rd parties