

CS 5154

Software Testing

Owolabi Legunsen

Spring 2021

Ch -

On the state of software quality

The New York Times *Airline Blames Bad Software in San Francisco Crash*



GOOGLE SELF-DRIVING CAR CAUSED FREEWAY CRASH AFTER ENGINEER MODIFIED ITS SOFTWARE

BY JASON MURDOCK ON 10/17/18 AT 11:34 AM

Newsweek



~9% of 2017
US GDP

Report: Software failure caused \$1.7 trillion in financial losses in 2017



Software testing company Tricentis found that retail and consumer technology were the areas most affected, while software failures in public service and healthcare were down from the previous year.

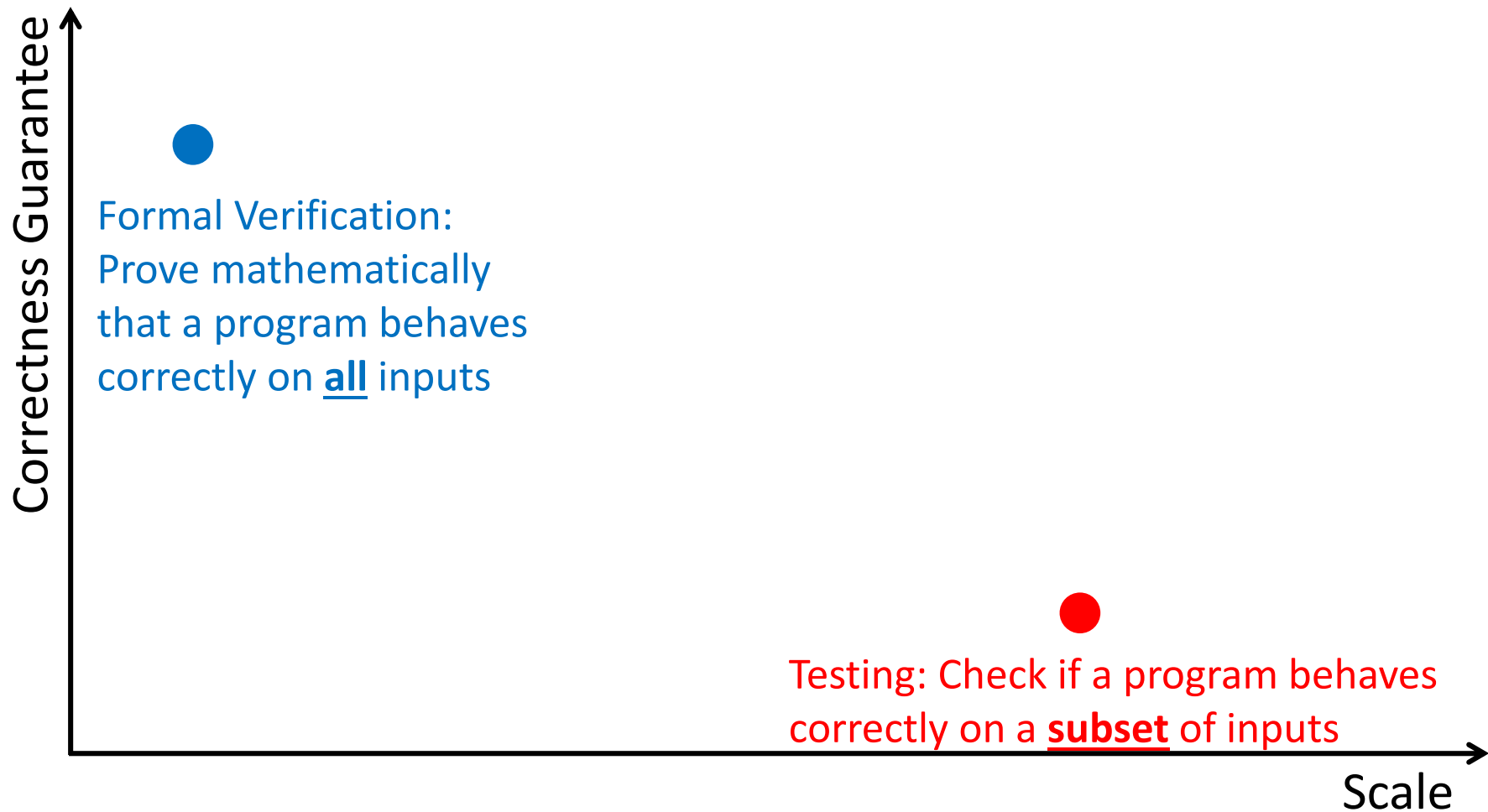
By Scott Matteson  | January 26, 2018, 7:54 AM PST

Hard Questions Raised When A Software 'Glitch' Takes Down An Airliner

Intro to Software Testing

- Testing is usually the last line of defense against bugs
- Testing is widely used for detecting bugs in practice
- **Does software behave correctly for given inputs?**

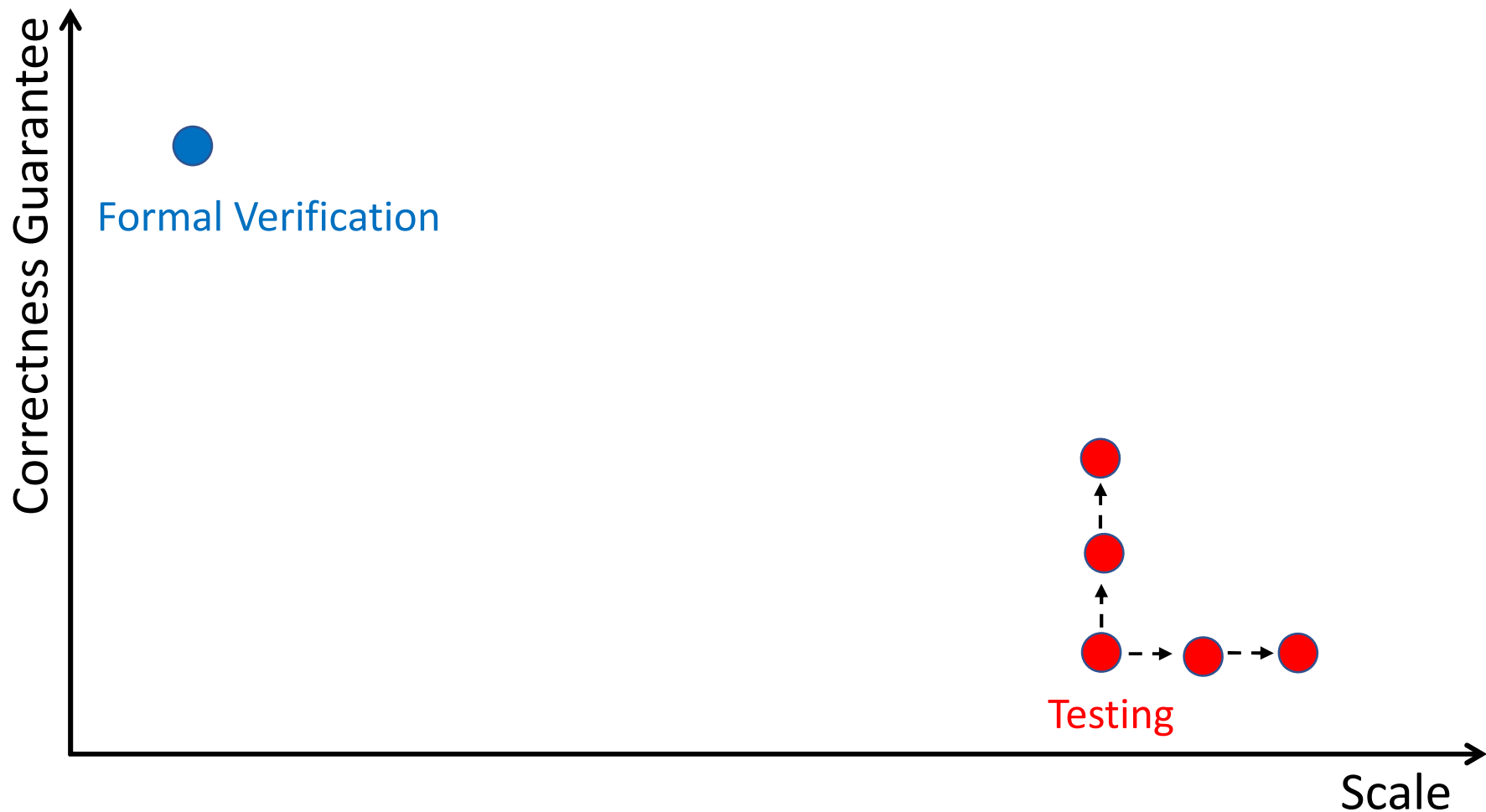
One reason testing is widely used



About me

- I work on integrating lightweight formal methods with software testing
 - side note: my research helped find over 500 bugs
- I received my PhD from UIUC in 2019
 - I used to think that Illinois winter was snowy 😊
- I became interested in software engineering research while I was working as a developer

What this course is about



How can we improve the guarantees that testing provides?

How can we make testing scale even better?

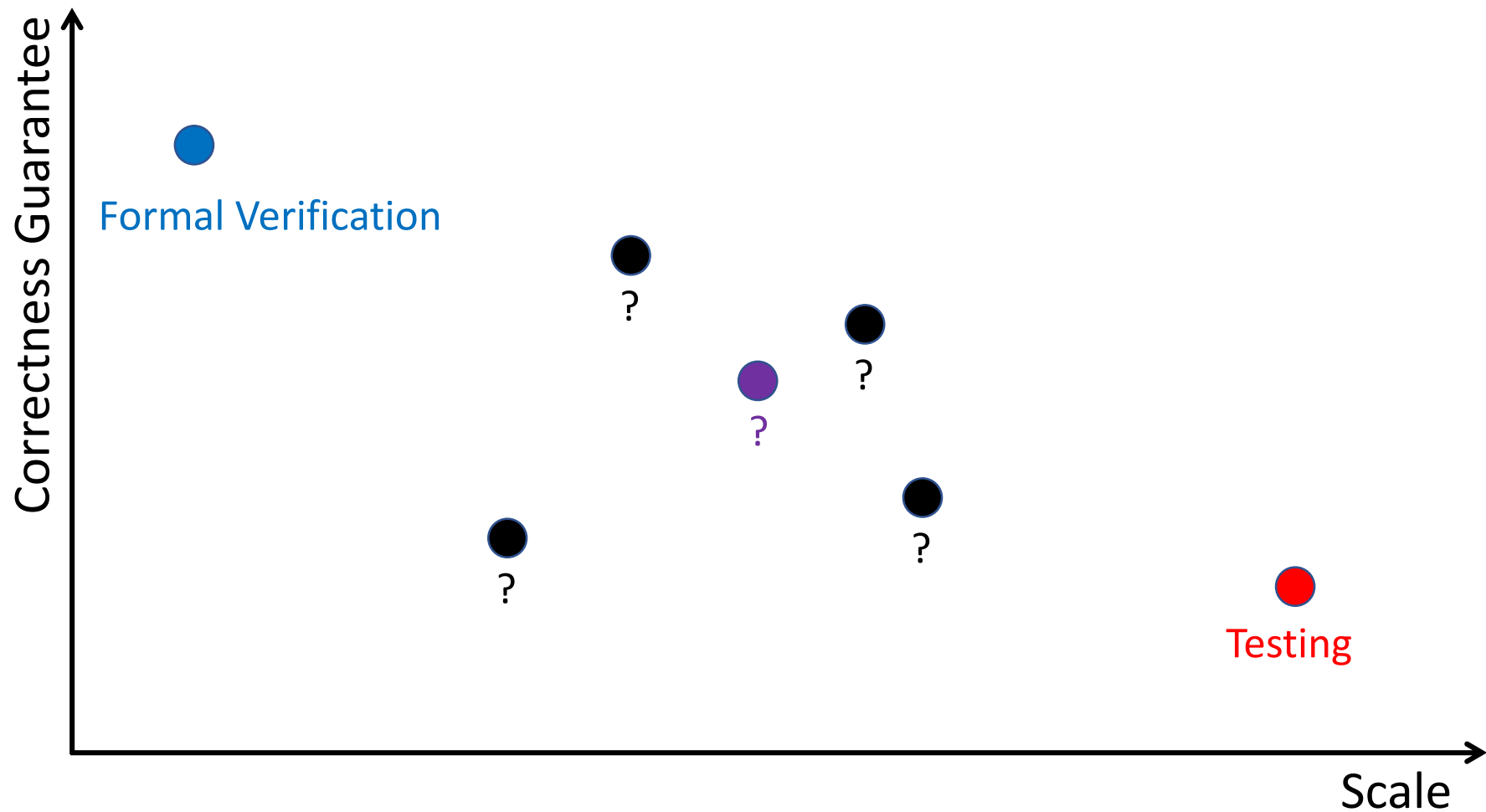
What this course is about (2)

- Systematic, organized approaches to testing
 - Learn how to obtain high quality tests
 - Learn cutting-edge testing techniques and tools
 - Project: apply techniques learned to real software
 - Bonus: figure out if software testing is an area of (research) interest for you

What this course is **not** about

- Developing software for clients
 - Covered in CS 5150
- Basic software engineering knowledge and skills
 - That's a prerequisite

Your turn: other QA approaches?



Small group discussion (10 mins)

- Introduce yourself to people in your group
- Make sure you type in your netid under your group no.
- Question: What other QA approaches did you use or hear about?
 - What are the advantages and disadvantages of each?
- Tabulate your group discussion in the Google sheet

What did your group discuss?

Code review

Beta testing

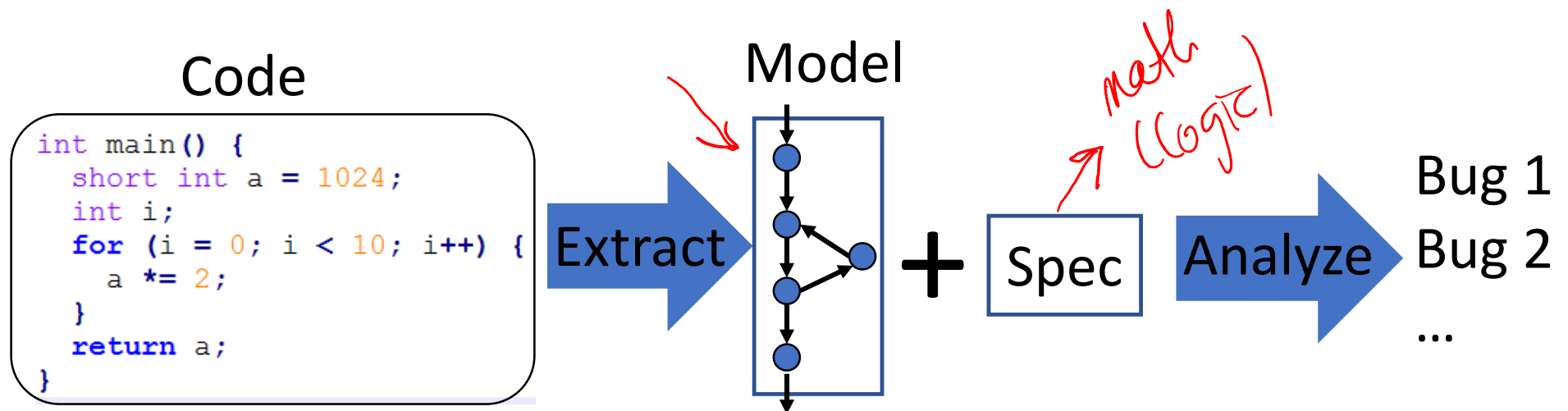
Design DRY : Do not repeat yourself
Encapsulation

Now that we broke the ice...

- Feel free to unmute yourself and ask questions
- Or you can post your question in the zoom chat
- At the very least, feel free to use zoom “raise hand”

Formal (static) verification

- E.g., model checking, static analysis



Pros	Cons
Good code coverage	Errors in modeling
Applied early in development	False positives
Mature and well studied	Does not scale

Software testing



Output == expected output

Pros	Cons
Easier for most developers	Low code coverage
Scales well in practice	Oracle generation is hard
Leverages developer insights	High maintenance costs, e.g., obsolete tests

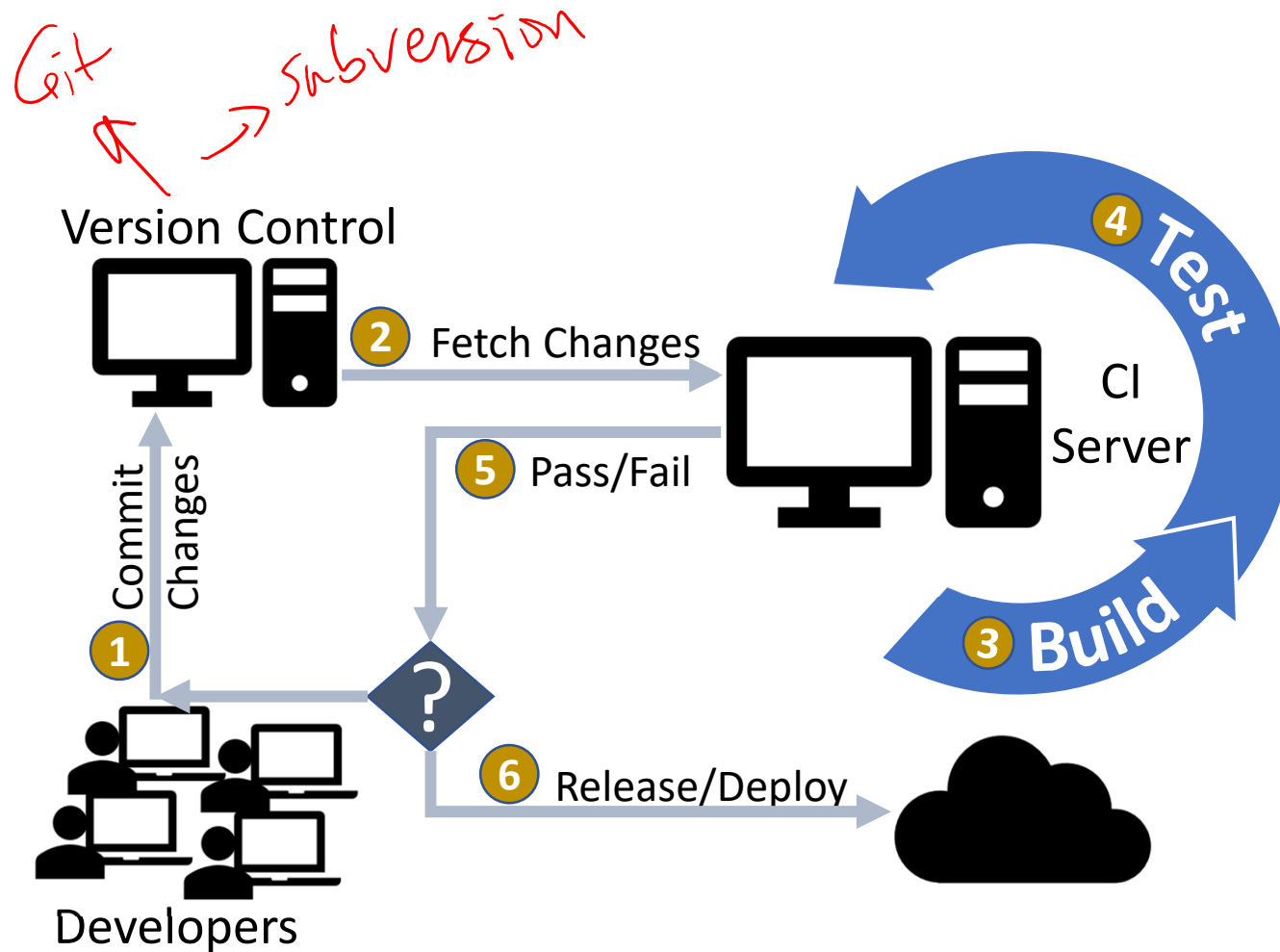
What you'll learn in this class

- Topics are organized around six themes/questions:
 1. How to automate the execution of tests?
 2. How to write high-quality tests?
 3. How to measure the quality of tests?
 4. How to automate the generation of tests?
 5. How to reduce the costs of executing existing tests?
 6. How to deal with bugs that tests reveal?

Theme 1: test automation

- The xUnit paradigm and the JUnit framework
- Parameterized Unit Tests
- Test Theories
- The Maven build system
- Continuous Integration (??)

Intro to Continuous Integration (CI)



Theme 2: writing high-quality tests

- Goal: systematically derive tests to exercise code in ways that increase the chance to reveal bugs
- We'll learn about criteria-based test design
 - Input-space partitioning
 - Graph coverage
 - Logic coverage
 - Syntax-based testing

Teaser: input-space partitioning?

```
public double computeAverage(int a, int b, int c) {...} ←
```

- How many possible values does each variable have on a 32-bit machine?

$$2^{32} = \sim 4 \text{ billion values}$$

- How many possible input values does the method have?

$$(2^{32})^3 = \sim 80 \text{ octillion values}$$

- If the method always runs in 1 sec, how long will it take to run all possible values?

$$((2^{32})^3) / 31557600 = \sim 2.5 \text{ sextillion years}$$

Teaser: input-space partitioning?

```
public double computeAverage(int a, int b, int c) {...}
```



How to systematically select values from the input domain such that each selected value exercises the program in a (possibly) different way?

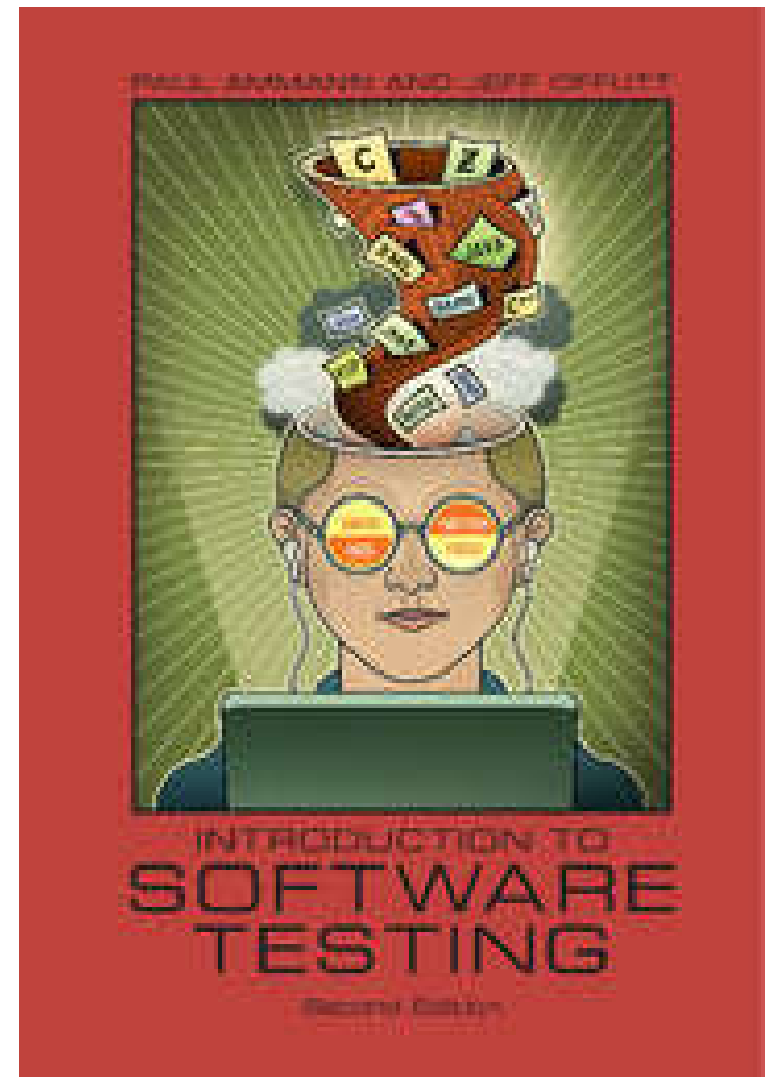
How to combine those values across multiple input variables parameters (e.g., a, b, and c for computeAverage)?

Theme 3: checking test quality

- Main idea: a good set of tests should catch artificially-seeded bugs
- Also helps find tests that need to be added
- We'll learn about mutation testing and use a mutation testing tool

Optional Text for Themes 1 – 3

- Classes will be self-contained
- Read the book if you want to know more about testing
- Readings will not be assigned from the book



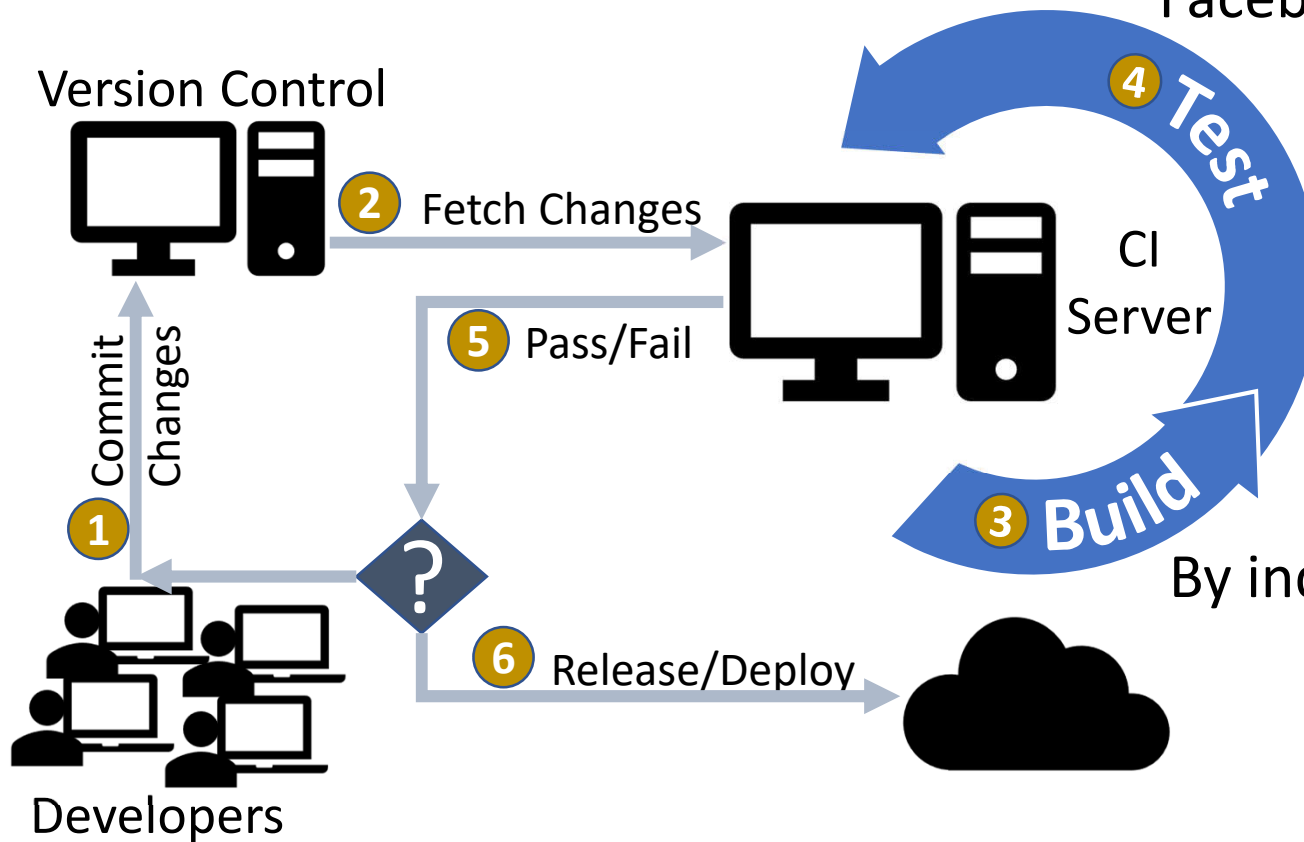
Theme 4: automatic test generation

- Problem: writing tests is very expensive
 - 75% of development effort at Microsoft (??)
 - Constrain: “customers pay for features, not tests” ☹️
- Goal: learn about cutting-edge automated test-generation techniques and tools
 - Random generation of test cases
 - Search-based generation of test cases
 - Fuzzing (??)

(input, expected output)

Quiz: How many CI cycles per day?

At companies like Microsoft, Facebook, Google?

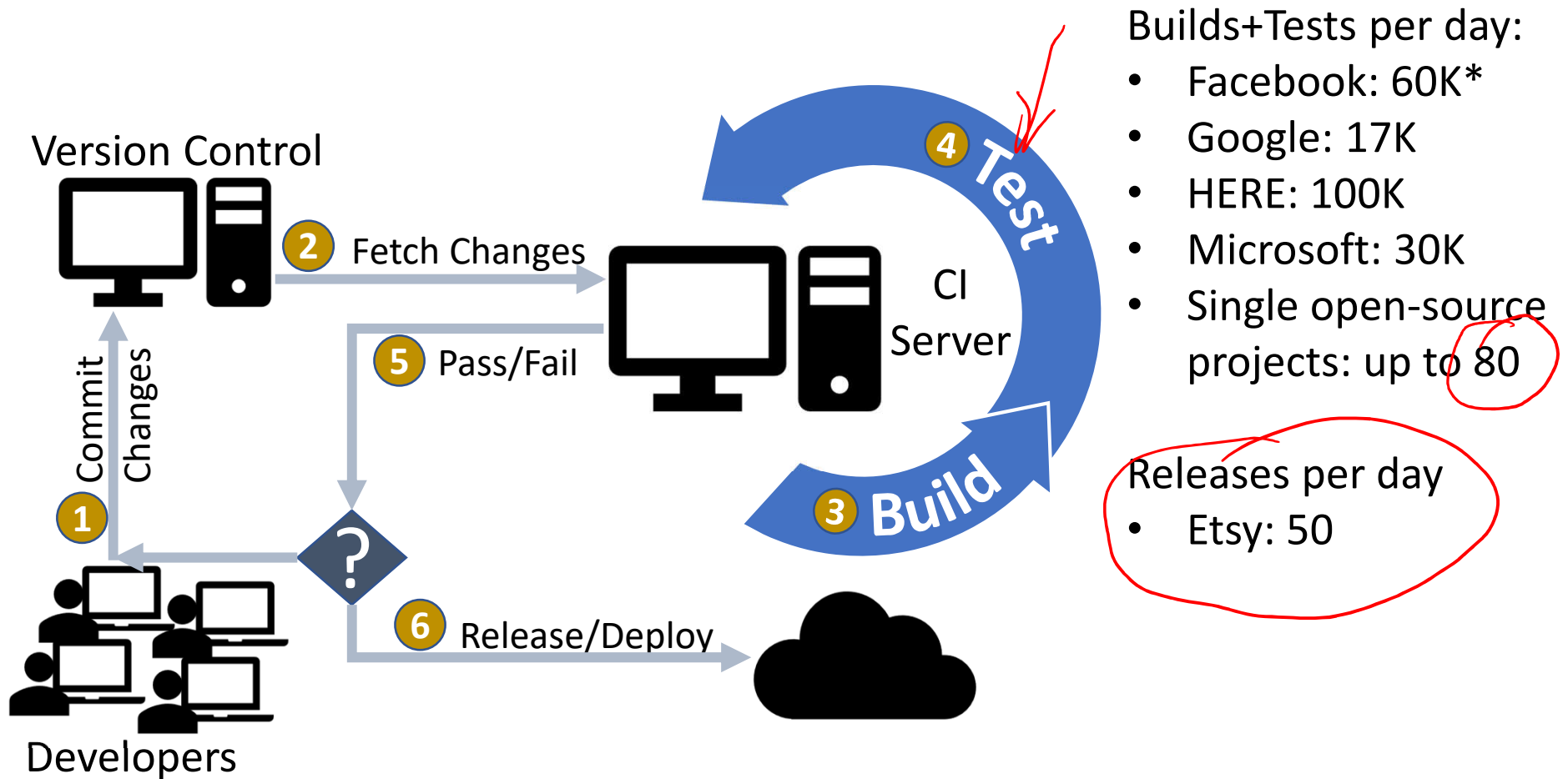


10K
50K
100K

By individual developers?

2-3

Tests are re-run very frequently



* Android only; Facebook: <https://bit.ly/2CAPvN9> ; Google: <https://bit.ly/2SYY4rR> ;
HERE: <https://oreil.ly/2T0EyeK> ; Microsoft: <https://bit.ly/2HgiUpw> ; Etsy: <https://bit.ly/2liSOJP> ;







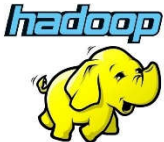
How long do tests take to run?

- Question: what's the longest your tests ever took to run?

45-50 minutes (NLP)
5 minutes (Server)

- Question: what do you typically do if your tests take too long to run?

Re-running tests is very costly

	test execution time	number of tests
	~5min	1667
	~10min	641534
	~45min	1296
	~45min	361
	~45min	631
	~4h	4975
	~17h	8663

Run many times each day

Re-running tests is very costly (2)



* linear increase in the number of revisions per day
linear increase in the number of tests per revision

=> **quadratic increase in test execution time**

75+ million tests run per day

20+ revisions per minute

Theme 5: regression testing

- Regression testing: rerunning tests after code changes
- Goal: learn about cutting-edge techniques for making regression testing more efficient and effective
 - Regression test selection
 - Dealing with flaky tests
 - Combining regression testing with runtime verification

Theme 6: dealing with bugs

- Context: your tests revealed a bug. Now what?
- If time permits, we'll learn about
 - Debugging
 - Bug Advocacy: best practices for getting others to fix the bugs that you found

Why should you learn about testing?

- Philosophy 1: each developer is responsible for testing their own code
 - e.g., at Google^[1], Facebook^[1], open-source
- Philosophy 2: developers create features, testers check those features
 - e.g., at Microsoft in the old days^[2]
- Philosophy agnostic:
 - Many well-paying jobs require software testing expertise
 - Society needs testers to reduce bug-induced catastrophes
 - Gain knowledge that can strengthen your CV

[1] <https://bit.ly/2OnWlTv> ; [2] <https://bit.ly/3rBagQo>;

Questions on course content?

?

Logistics

Software Testing

Spring 2021

Software testing is widely used for detecting flaws in software. Systematic and organized approaches to testing will be discussed, including test adequacy criteria, manual and automatic generation of test inputs, regression testing, debugging, and at least one dynamic analysis for detecting known classes of errors. Students will learn how to design and automate the execution of high-quality software tests. Students will also learn how to generate test suites that meet coverage and other adequacy criteria.

Prerequisites. Graduate standing (Ph.D, MS, or MEng) in CS, or CS majors who have taken CS 3110 or CS 4120, or permission of instructor required. Experience with Java will be helpful for programming assignments.

This course is in Beta. CS 5154 is a brand new course. Everything might change. Nothing is certain.

Logistics: CS 5154 in β is not...



You



Me

Logistics: CS 5154 in β should be



CS5154 information

- Instructor: Owolabi Legunsen
 - Web: <https://www.cs.cornell.edu/~legunsen>
 - Email: legunsen@cornell.edu
 - Office Hours: Right after class on Zoom
- TA: Daniel Martin
 - Email: dm839@cornell.edu, Office Hours: TBD
- Course web page (with in-progress schedule)
 - <https://www.cs.cornell.edu/courses/cs5154/2021sp>
 - **Take some time to go through the web page this week**
 - Check the news section frequently for announcements

You are expected to...

- Complete 5 – 6 homework assignments
- Conduct a research project on software testing
 - Semester-long, split into phases
 - Each phase may not build on the previous one
- Work effectively in teams
 - Software engineering (including testing) is a team sport

Your grade will be based on...

Homework assignments	50%
Course project	40%
Participation and Subjective factors	10%

Homework assignments

- 5 – 6 homework throughout the semester
- Goals:
 - Assess your understanding of reading and lectures
 - Give you opportunity to reinforce what you learned
 - Get practice working in a team
- Content:
 - mix of written solutions, programming, readings

Course Project

- Work in small teams to test open-source software
- Goals:
 - Gain deeper knowledge and expertise than we can cover in class and homework
 - Learn how to test (parts of) existing and large software
 - Learn cutting-edge tools techniques and tools and apply them to open-source code

Open-source code you may test

- Apache commons-math
 - <https://github.com/apache/commons-math>

Apache Commons Math

build passing coverage 91% maven central 3.6.1 license apache

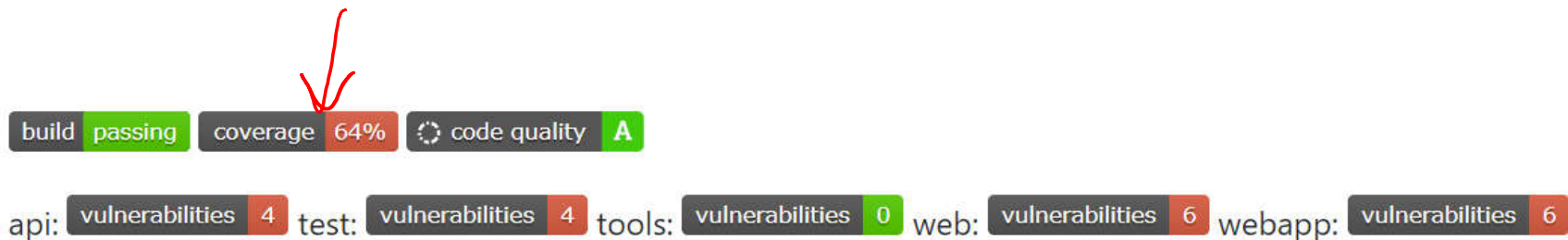
The Apache Commons Math project is a library of lightweight, self-contained mathematics and statistics components addressing the most common practical problems not immediately available in the Java programming language or commons-lang.

Some code that you may test (2)

- OpenMRS core
 - <https://github.com/openmrs/openmrs-core>



OpenMRS
MEDICAL RECORD SYSTEM



OpenMRS is a patient-based medical record system focusing on giving providers a free customizable electronic medical record system (EMR).

The mission of OpenMRS is to improve health care delivery in resource-constrained environments by coordinating a global community that creates a robust, scalable, user-driven, open source medical record system platform.

Participation

- Come to class, participate, and ask questions
 - Or watch video and ask questions on Ed Discussions
- Find bugs in homework, projects, etc.
- Be an effective teammate

Tentative timeline for course project

- Detailed plans: TBD
- Project kick-off date: 2/24/2021
- Final project report due: 5/14/2021
 - At least three other reports will be required during the semester

Questions on content or logistics?

?

Videos : Canvas

Before next class (pre-homework)

- Read the course webpage
 - <https://www.cs.cornell.edu/courses/cs5154/2021sp>

Next class...

- Start with the basics: testing costs, concepts,
- Homework 0 is assigned
 - Due by 11:59pm EST Tuesday 2/16/2021
 - To be completed individually
 - We will only answer clarification questions about the instructions

A review of today's class

- High-level introduction to Software Testing
- Comparison of testing with other QA approaches
- Whirlwind tour of how this course is organized
- Learning outcomes, course content, and logistics