



Intro to Testing

Trust me, it's worth it

Unit Testing

01

Integration Testing

02

TODAY'S AGENDA

03

Other types of testing

04

Conclusion

What is unit testing?

- Test functionality of a small piece of code in isolation
 - Should not depend on external software, such as database, network etc
 - *REPRODUCIBILITY!*
- Tests shouldn't depend on *how* the code is implemented
 - Benefit: you can rewrite and change your code with confidence: if you have tests, make sure they pass after you make changes

Unit testing - substring search

- We want to search a large string T for a substring P . What do we test?

Unit testing - substring search test cases

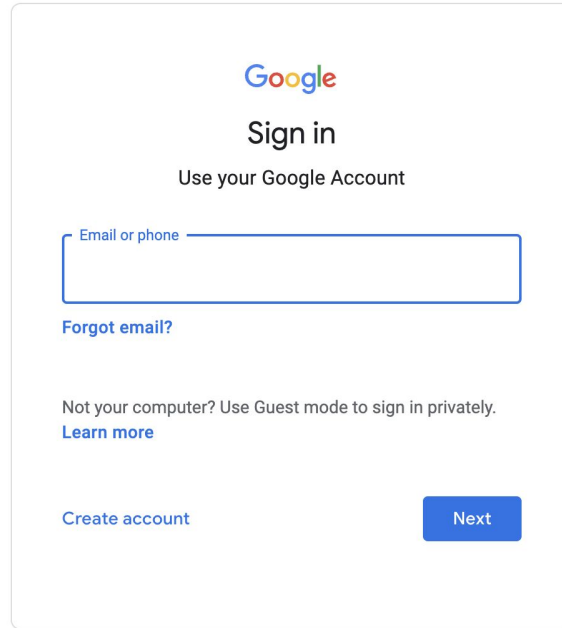
- Pattern in text
 - T = "hello" P = "el"
- Pattern not in text
 - T = "hello" P = "elll"
- Pattern empty
 - T = "hello" P = ""
- Text empty
 - T = "" P = "hello"
- Multiple occurrences
 - T = "hello hello" P = "hello"
- Test cases cover "edge" inputs - those which are nonstandard and are more likely to lead to errors

What is integration testing?

- Test functionality of how many pieces work together
 - May depend on external software, such as a database
 - Tests mirror behavior your application exposes to the user
 - *REPRODUCIBILITY still important! Reset external dependencies before running!*
- Tests shouldn't depend on *how* the code is implemented
 - Benefit: you can rewrite and change your code with confidence: if you have tests, make sure they pass after you make changes

Integration testing - login page

What would you test?



The image shows a screenshot of the Google Sign in page. At the top center is the Google logo. Below it, the text "Sign in" is displayed in a large, bold font, followed by "Use your Google Account" in a smaller font. A text input field is present with the placeholder text "Email or phone". Below the input field is a link for "Forgot email?". Further down, there is a message: "Not your computer? Use Guest mode to sign in privately." with a "Learn more" link. At the bottom left, there is a "Create account" link, and at the bottom right, there is a blue "Next" button.

English (United States) ▼

[Help](#)

[Privacy](#)

[Terms](#)

Integration testing - login page test cases

- User can log in with existing account
- User can create account and then log in
- User gets an error when their username or password are incorrect
- User can reset their password through forgot password

- Note that the above don't mention specific *units*, modules, etc in your code: they're just actions that should stay the same no matter what!

Integration testing vs unit testing

- Participation activity: open the Testing activity on Canvas, and answer the question provided: do you think integration or unit testing is more important? Why? What type should most of your tests be?

Integration testing vs unit testing

- Participation activity: open the Testing activity on Canvas, and answer the question provided: do you think integration or unit testing is more important? Why? What type should most of your tests be?
- Answer: it depends! If you *have* to pick one, it should probably be integration tests, since they're more like what the user sees (but they can be harder to write)
- Balance of integration and unit tests depends on your application
 - For example, a self-driving car should have lots of unit *and* integration tests, but a Discord Bot might be fine with only integration tests

Other ways to test - property based testing

- Ensures that a piece of code satisfies a *mathematical* property
 - For example, consider implementing a function that finds the greatest common divisor of two integers: you want to verify that any output is positive, that there is no greater divisor, etc
- Property based testing libraries automatically worry about edge cases: you just specify the properties of the output that you want to be true!
- Great to use as a partial/complete replacement for unit tests
- Now available in a language near you: Hypothesis library for Python, fast-check for JavaScript, etc

Other ways to test - fuzzing

- Generates random/adversarial inputs to a program and tries to crash it and/or get unexpected results
- Good complement to integration testing
 - For example, with fuzzing you might be able to auto-generate adversarial inputs to the login page, such as non-unicode emails, extremely long emails, random clicks, etc

TLDR

- Four great ways to test your application
 - **Unit testing:** handwritten tests of small parts such as functions, manually covering edge cases
 - **Property based testing:** formal specification of how functions should behave, automatically covering edge cases
 - **Integration testing:** test how components of your application work together
 - **Fuzzing:** randomly generate inputs to try and break your code
- In CS222 we ask that you write a few tests for each of your PRs - they can be any type of test you want!

