Quality Assurance and Testing

DEV

Microsoft announces Battle Royale Mode for Visual Studio 2019

EMIL PROTALINSKI   @EPRO   JUNE 6, 2018 10:58 AM

VB Recommendations

Ctrl-labs' armband lets you control computer cursors with your mind

What Alienware has learned from 10 years of esports

Microsoft today announced Visual Studio 2019, the next version of its IDE with integrated Battle Royale mode. Release timing will be shared "in the coming months," with the company simply promising "to deliver Visual Studio 2019 quickly and iteratively." The news comes days after Microsoft's acquisition of GitHub.

Logistics: Room Swap?

- We were asked by an adjacent class (a smaller class in a larger room) to swap rooms

- This class may move to LCSIB 13**5**5 (next door)

- This will only happen if the lecture recordings can be seamlessly transferred

  - No old recordings lost, all new recordings visible

- If so, it would happen next week or later

- If so, we would give another warning

  - Like an all-class email or post

One-Slide Summary

- **Quality Assurance** maintains desired product properties through process choices.

- **Testing** involves running the program and inspecting its results or behavior. It is the dominant approach to software quality assurance. There are numerous methods of testing, such as regression testing, unit testing, and integration testing.

- **Mocking** uses simple replacement functionality to test difficult, expensive or unavailable modules or features.

3

Last Time: Metrics and Management

mcmillen
@mcmillen

Reportedly, Elon stack-ranked Twitter engineers by "lines of code written in the last year" and fired the bottom X%, which likely means he's gotten rid everyone who worked on particularly challenging things, like security, privacy, performance, & reliability. 😬

2:48 PM · 04 Nov 22 · Twitter Web App

775 Retweets   **476** Quote Tweets   **4,363** Likes
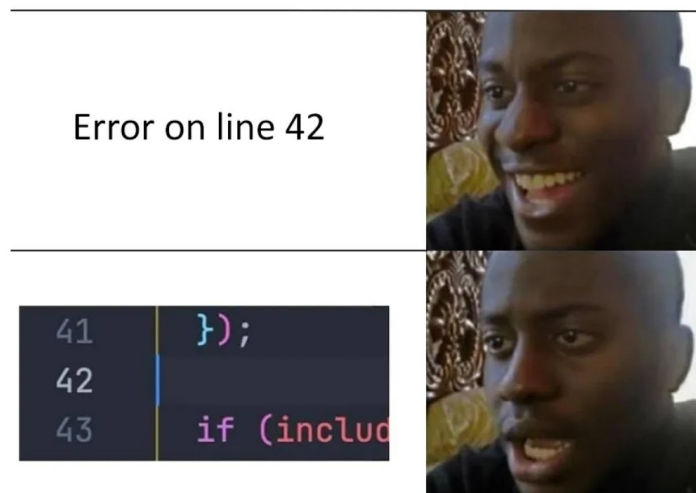
Story So Far

- We want to deliver high-quality software at a low cost. We can be more efficient if we plan and use a software development process.

- Planning requires information: we measure the world to combat uncertainty and mitigate risk.

- But how do we measure, assess or assure software quality?

Official Definition

- Quality assurance is the maintenance of a desired level of quality in a service or product, especially by means of attention to every stage of the process of delivery or production.

  - Oxford English Dictionary

That Moment...

Error on line 42

```
41    });
42
43    if (includ
```

Brenan Keller
@brenankeller

A QA engineer walks into a bar.
Orders a beer. Orders 0 beers.
Orders 99999999999 beers.
Orders a lizard. Orders -1 beers.
Orders a ueicbksjdhd.

First real customer walks in
and asks where the bathroom
is. The bar bursts into flames,
killing everyone.

1:21 PM · 30 Nov 18

6

Quality Motivation

- **External (Customer-Facing) Quality**
  - Programs should "do the right thing"
    - So that customers buy them!
- Internal (Developer-Facing) Quality
  - Programs should be readable, maintainable, etc.

Internal-Facing Quality

- If the dominant activity of software engineering is maintenance …
  - Then internal quality is mostly maintainability!
- How do we ensure maintainability?
  - Human code review
  - Code analysis tools and linters
  - Using programming idioms and design patterns
  - Following local coding standards
- *More on this in future lectures!*

External-Facing Quality

- What does "*Do The Right Thing*" Mean?

- Behave according to a specification

  - Foreshadowing: What is a good specification?

- Don't do bad things

  - Security issues, crashing, etc.

  - Some failure is inevitable. How to handle it?

- Robustness against maintenance mistakes

  - Do "fixed" bugs sneak back into code?

Doing The Right Thing

- Why don't we just write a new program X to tell us if our software Y is correct?

Carla Notarobot 🤖💻
@CarlaNotarobot

Boss: Where did you get this code?
Me: Stack Overflow
Boss: From the questions or the answers?

Pranay Pathole
@PPathole

Programming is like a "choose your own adventure game" except every path leads you to a StackOverflow question from 2013 describing the same bug, with no answer.

Doing The Right Thing

- Why don't we just write a new program X to tell us if our software Y is correct?

- The Halting Problem prevents X from giving the right answer every time

  - ~~X always gives a wrong answer~~

  - X cannot always give a right answer

- We can still approximate!

  - Type systems, linters, static analyzers, etc.

Practical Solution: Testing

Testing

- "**Software testing** is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test."

- A typical test involves input data and a comparison of the output. (More next lecture!)

- Note: unless your input domain is finite, testing does *not* prove the absence of all bugs.

- Testing gives you confidence that your implementation adheres to your specification.

Testing in UM EECS Courses (1/3)

- EECS 183 and 482

- 1 main() function == 1 test

- For each test

  - Run test against correct solution, save output

  - For each buggy solution

    - Run test against buggy solution, diff output with result from correct solution

    - If outputs differ, a bug is exposed!

Testing in UM EECS Courses (2/3)

- EECS 281

- 1 input file == 1 test

- For each test

  - Pipe input to correct solution, save output

  - For each buggy solution

    - Pipe input to buggy solution, diff output with result from correct solution

    - If outputs differ, a bug is exposed!

Testing in UM EECS Courses (3/3)

- EECS 280
- 1 function with assert()s == 1 test
- For each test
  - Run test against correct solution
    - Throw out the test if it fails
  - For each buggy solution
    - Run test against buggy solution
    - If assertion fails, a bug is exposed!

Exercise: UM EECS Testing

- With your neighbor, discuss and write down brief pros and cons of each testing method

  - If notecards are passed around, write your UM email(s) in big block letters (e.g., "`weimerw`")

    - We can't read it → we can't give you credit for it

- Recall

  - 183/482: 1 main() function == 1 test; output diff

  - 281: 1 input file == 1 test; output diff

  - 280: 1 function with assert()s == 1 test; assertion failure

Testing: Inputs and Outputs

- For 183/281/482, students write program inputs, but not expected outputs

- For 280, students write program inputs and also expected outputs

- In real life, you rarely have an already-correct implementation of your program

- Testing with random inputs (**fuzz testing**) can help detect "bad things" bugs (segfaults, memory errors, crashes, etc.)

  - But does not provide full expected outputs

Testing Concepts

- Regression Testing
- Unit Testing
- xUnit
- Test-Driven Development
- Integration Testing
- Mocking

■ Language Matters

Why English is so hard to learn

\\\ Marlene Davis

YOU think English is easy? Check out the following.

1. The bandage was wound around the wound.
2. The farm was cultivated to produce produce.
3. The dump was so full that the workers had to refuse more refuse.
4. We must polish the Polish furniture shown at the store.
5. He could lead if he would get the lead out.
6. The soldier decided to desert his tasty dessert in the desert.
7. Since there is no time like the present, he thought it was time to present the present to his girlfriend.
8. A bass was painted on the head of the bass drum.
9. When shot at, the dove dove into the bushes.
10. I did not object to the object which he showed me.
11. The insurance was invalid for the invalid in his hospital bed.
12. There was a row among the oarsmen about who would row.
13. They were too close to the door to close it.
14. The buck does funny things when the does (females) are present.
15. A seamstress and a sewer fell down into a sewer line.
16. To help with planting, the farmer taught his sow to sow.
17. The wind was too strong to wind the sail around the mast.
18. Upon seeing the tear in her painting she shed a tear.
19. I had to subject the subject to a series of tests.
20. How can I intimate this to my most intimate friend?

Heteronyms

These are brilliant. Homonyms or homographs are words of like spelling, but with more than one meaning and sound.

When pronounced differently, they are known as heteronyms.

Regression Testing (in one slide)

- Have you ever had one of those "I swear we've seen and fixed this bug before!" moments?

  - Perhaps you did, but someone else broke it again

  - This is a **regression** in the source code

- Best practice: when you fix a bug, add a test that specifically exposes that bug

  - This is called a **regression test**

  - It assesses whether future implementations still fix the bug

Regression Testing Story

```
// Dear maintainer:
//
// Once you are done trying to 'optimize' this routine,
// and have realized what a terrible mistake that was,
// please increment the following counter as a warning
// to the next guy:
//
// total_hours_wasted_here = 42
```

https://stackoverflow.com/questions/184618/what-is-the-best-comment-in-source-code-you-have-ever-encountered/482129#482129
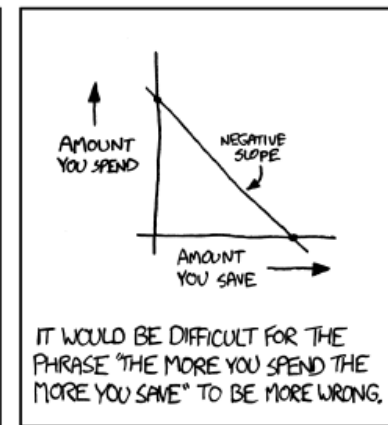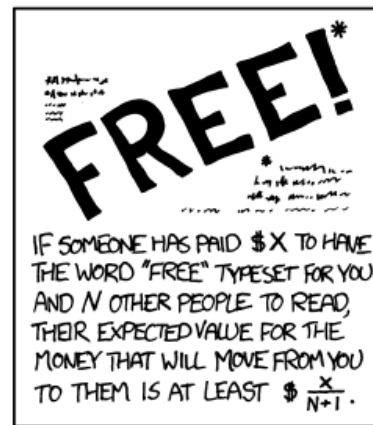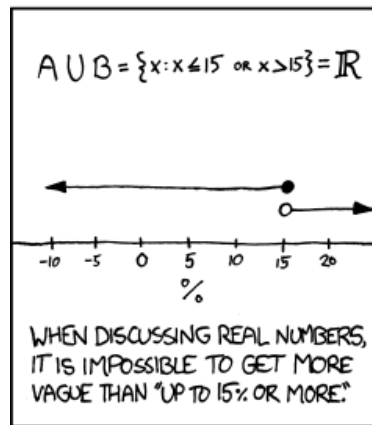
Unit Testing and Frameworks

- In **unit testing**, "individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use."

- Modern frameworks are often based on SUnit (for Smalltalk), written by Kent Beck

  - Java JUnit, Python unittest, C++ googletest, etc.

- These frameworks are collectively referred to as **xUnit**

xUnit Features

- Test cases "look like other code"

  - They are special methods written to return a boolean or raise assertion failures

- A test case discoverer finds all such tests

  - Special naming scheme, dynamic reflection, etc.

- A test case runner chooses which tests to run

MATHEMATICALLY ANNOYING ADVERTISING:

$A \cup B = \{x : x \leq 15 \text{ or } x > 15\} = \mathbb{R}$

WHEN DISCUSSING REAL NUMBERS, IT IS IMPOSSIBLE TO GET MORE VAGUE THAN "UP TO 15% OR MORE."

FREE!*

*IF SOMEONE HAS PAID $X TO HAVE THE WORD "FREE" TYPESET FOR YOU AND N OTHER PEOPLE TO READ, THEIR EXPECTED VALUE FOR THE MONEY THAT WILL MOVE FROM YOU TO THEM IS AT LEAST $ $\frac{X}{N+1}$.

AMOUNT YOU SPEND / NEGATIVE SLOPE / AMOUNT YOU SAVE

IT WOULD BE DIFFICULT FOR THE PHRASE "THE MORE YOU SPEND THE MORE YOU SAVE" TO BE MORE WRONG.

23

xUnit Definitions

- In xUnit, a test case is

  - A piece of code (usually a method) that establishes some preconditions, performs an operation, and asserts postconditions

- A **test fixture**

  - Specifies code to be run before/after each test case

  - Each test is run in a "fresh" environment

- Special assertions

  - Check postconditions, give helpful error messages

Python unittest Example

```python
import unittest
class NiceThing:
    def __init__(self, num_spams):
        self.num_spams = num_spams
    def zap(self):
        return self.num_spams + 42


class NiceThingTestCase(
        unittest.TestCase):
    def setUp(self):
        self.nice_thing = NiceThing(0)
    def test_zap(self):
        self.assertEqual(45, self.nice_thing.zap())


if __name__ == '__main__':
    unittest.main()
```

```
$ python3 unit_test_demo.py
F
======================================
FAIL: test_zap (__main__.NiceThingTestCase)
--------------------------------------
Traceback (most recent call last):
  File "unit_test_demo.py", line 11, in test_zap
    self.assertEqual(45, self.nice_thing.zap())
AssertionError: 45 != 42


--------------------------------------
Ran 1 test in 0.001s

FAILED (failures=1)
```

Python unittest Details

- Discussion Sections will provide more details

- See Python unittest documentation:

  - https://docs.python.org/3/library/unittest.html

*Python Installation Was Successful ✅
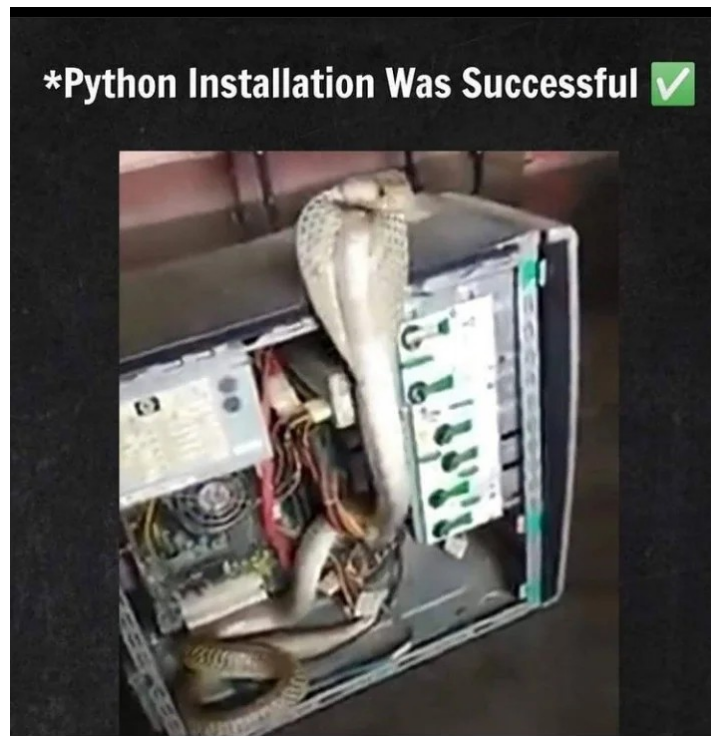
Unit Testing Advantages

- Unit testing tests features in isolation

  - In the previous example, our test for zap() tested only the zap() method

  - Advantage: when a test fails, it is easier to locate the bug

- Unit testing tests are small

  - Advantage: smaller test are easier to understand

- Unit testing tests are fast

  - Advantage: fast tests can be run frequently

EECS UM Unit Testing

- Recall the Euchre project from EECS 280

  - Card, Pack and Player classes

  - A top-level "play Euchre" application

- Suppose you wrote Card, Pack and Player without testing, and then wrote "play Euchre"

  - What do you do when you find a bug in "play Euchre"?

Test-Driven Development

- "**Test-driven development** is a software development process that relies on the repetition of a very short development cycle: requirements are turned into very specific test cases, then the software is improved so that the tests pass."

- Write a unit test for a new feature

  - When you run the test, it should fail

- Write the code that your unit test case tests

- Run all available tests

  - Fix anything that breaks; repeat until no tests fail

- Go back to step 1

Integration Testing

- Typically, any feature can be made to work in isolation

- What happens when we put our unit-tested features together into a larger program?

- Does our application work from start to finish?

  - "End-to-end" testing

- **Integration testing** combines and tests individual software modules as a group.
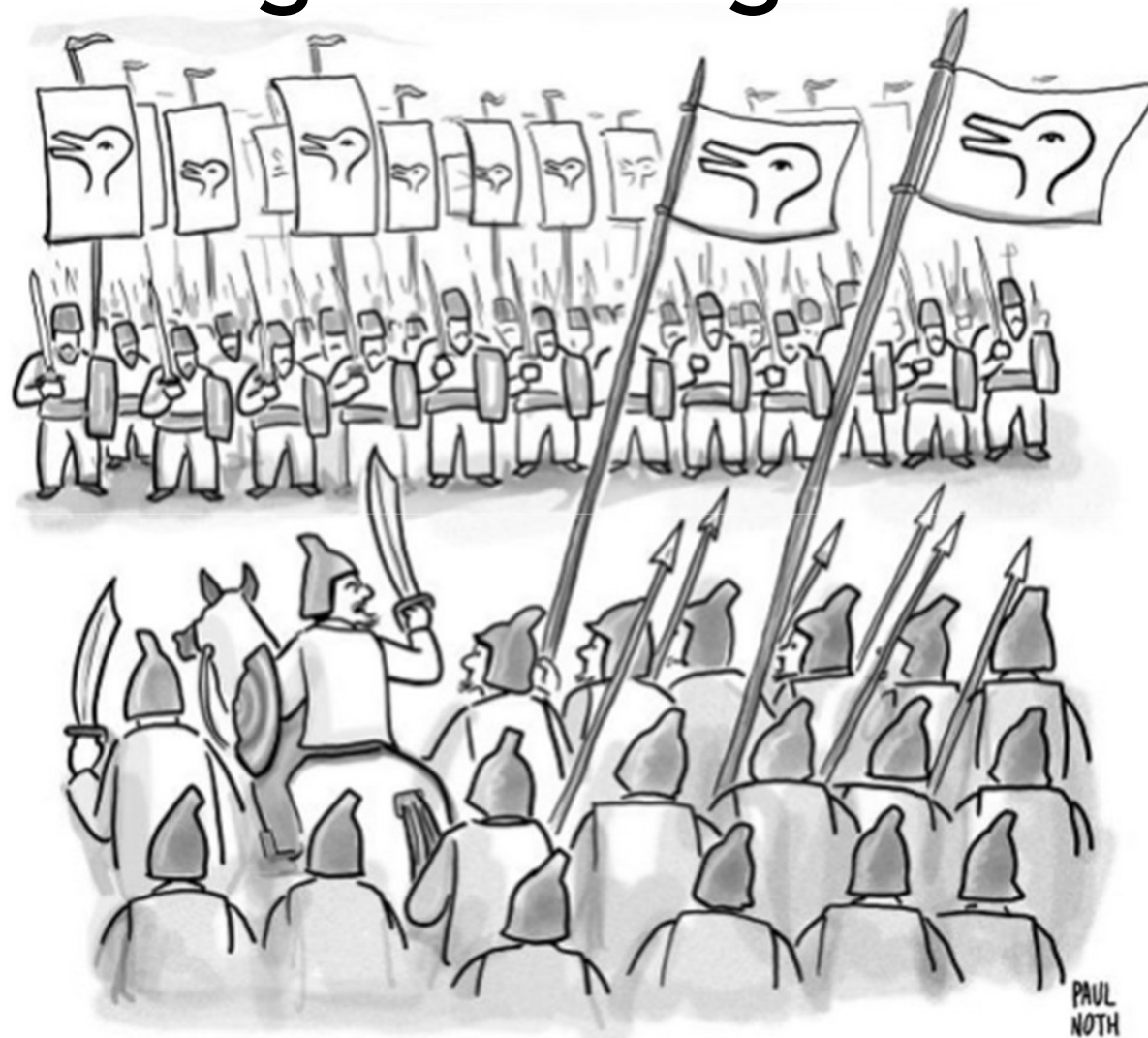
Unit Testing vs. Integration Testing

- Are those "unit tests" for Pack and Player actually integration tests?
  - Does Pack build on or use Card, for example?

Unit Testing vs. Integration Testing

"There can be no peace until they renounce their Rabbit God and accept our Duck God."

Unit and Integration Abstractions

- Once you've unit-tested an ADT, you build atop it and write unit tests for subsequent modules at a higher level of abstraction

  - This also promotes a modular, decoupled design

- Example: we already do this with Integer, etc.

  - "Does that mean that our tests that rely on integers aren't really unit tests? No. We can treat integers as a given and we do. Integers have become part of the way we think about programming." - Kent Beck

Integration Testing Examples

- Integration testing is application-specific

- EECS Classes

  - Run main program with input file, diff output

- Web and GUI Applications

  - Use a testing framework (or harness) that lets you simulate user clicks and other input

- Systems Software

  - Use a testing framework that lets you simulate disk and network failures (cf. Chaos Monkey later)

Creative Integration Testing Examples

- For video games, you might write an AI to play

  - Bayonetta
    https://www.platinumgames.com/official-blog/article/6968

  - Cloudberry Kingdom

    https://www.gamasutra.com/view/feature/170049/how_to_make_insane_procedural_.php

- Or have players use gaze-detecting goggles

  https://www.tobiipro.com/fields-of-use/user-experience-interaction/game-usability/

  "We see ... modern eye tracking technology as a future standard in modern QA teams to improve the overall quality of game experiences."

  - Markus Kassulke, CEO, HandyGames

Psychology: Backfire Effect

- Is there a difference between being *un*informed and being *mis*informed?

  - Correct factual ignorance or misperception …

- "However, individuals who receive unwelcome information may not simply resist challenges to their views. Instead, they may come to support their original opinion even more strongly – what we call a **backfire effect**."

Psychology: Backfire Effect

- Human studies of 130 + 197 participants

- Found that conservative supporters of president Bush "doubled down" when presented with evidence that there were no weapons of mass destruction in Iraq before the 2003 US invasion.

- Commonly referenced in popular press, message boards, etc.

[B Nyhan and J Reifler. (2010). When Corrections Fail: The persistence of political misperceptions. In Political Behavior 32(2):303-330.]

Psychology: Backfire Effect

- "Four experiments in which we enrolled more than 8,100 subjects and tested 36 issues of potential backfire. Across all experiments, we found only one issue capable of triggering backfire: whether WMD were found in Iraq in 2003. Even this limited case was susceptible to a survey item effect [...] Evidence of factual backfire is far more tenuous than prior research suggests. By and large, citizens heed factual information, even when such information challenges their partisan and ideological commitments." [T Wood and E Porter. (2018). The elusive backfire effect: mass attitudes' steadfast factual adherence. In Political Behavior, pp. 1-29.]

Psychology: Confirmation Bias

- **Confirmation bias** is the tendency to search for, interpret, favor, and recall information in a way that affirms one's prior beliefs or hypotheses. It includes a tendency to test ideas in a one-sided way, focusing on one possibility and ignoring alternatives.

- It is so well-established that experimental evidence is available in many flavors

[R Nickerson. (1998). Confirmation Bias: A Ubiquitous Phenomenon in Many Guises. In Review of General Psychology, 2(2):175-220.]

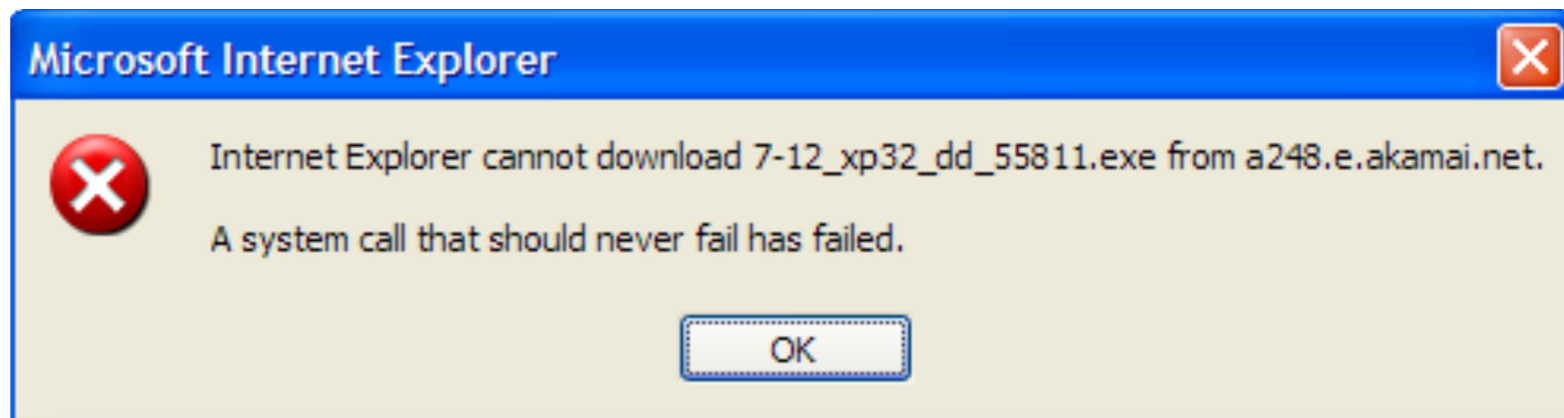# Psychology: Confirmation Bias
(each subclaim has its own studies)

- Restriction of attention to a favored hypothesis

- Preferential treatment of evidence supporting existing beliefs

- Looking only, or primarily, for positive cases

- Overweighting positive confirmatory instances

- Seeing what one is looking for

- Favoring information acquired early

Psychology: Confirmation Bias

- Implications for SE:

- **Policy Rationalization** justifies policies to which an organization has already committed. "Once a policy has been adopted and implemented, all subsequent activity becomes an effort to justify it."

- **Theory Persistence** involves holding to a favored idea long after the evidence against it has been sufficient to persuade others who lack vested interests.

- Idea or policy = any SE process decision.

Targeting Hard-To-Test Aspects

- What if we want to write unit or integration tests for some ADT, but the ADT has expensive dependencies?

- Exercise: generate three examples of things that are hard to test because of their dependencies or other expense factors.

Microsoft Internet Explorer

Internet Explorer cannot download 7-12_xp32_dd_55811.exe from a248.e.akamai.net.

A system call that should never fail has failed.

OK

Mocking

- "**Mock objects** are simulated objects that mimic the behavior of real objects in controlled ways."

- In testing, **mocking** uses a mock object to test the behavior of some other object.

  - Analogy: use a crash test dummy instead of real human to test automobiles

44

Scenario 1: Web API Dependency

- Suppose we're writing a single-page web app

- The API we'll use (e.g., Speech to Text) hasn't been implemented yet or costs money to use

- We want to be able to write our frontend (website) code without waiting on the server-side developers to implement the API and without spending money each time

- What should we do?

Mocking Dependencies

- Solution: make our own "fake" ("mock") implementation of the API

- For each method the API exposes, write a substitute for it that just returns some hard-coded data (or any other approximation)

  - Why does this work? Are there relevant concepts from 280?

- This technique was used to design and test parts of the autograder.io website

Scenario 2: Error Handling

- Suppose we're writing some code where certain kinds of errors will occur sporadically once deployed, but "never" in development

  - Out of memory, disk full, network down, etc.

- We'd like to apply the same strategy

  - Write a fake version of the function …

- But that sounds difficult to do manually

  - Because many functions would be impacted

  - Example: many functions use the disk

Mocking Libraries: Two Approaches

- Before running the program ("static")

  - Combine modularity/ecapsulation with mocking

  - Move all disk access to a wrapper API, use mocking there at that one point (coin flip → fake error)

- While running the program ("dynamic")

  - While the program is executing, have it rewrite itself and replace its existing code with fake or mocked versions

  - Let's explore this second option in detail

Dynamic Mocking Support

- Some languages provide **dynamic mocking libraries** that allow you to substitute objects and functions at runtime

  - For one test, we could use a mocking library to force another line of code inside our target function to throw an exception when reached

- This feature is available in modern dynamic languages with reflection (Python, Java, etc.)

  - googletest used to require a special base class for this sort of mocking, now it uses macros

Dynamic Mocking Example

```python
import unittest
from unittest import mock

def lowLevelOp():
  # might fail for users
  # example: no memory
  pass

def highLevelTask():
  try:
    lowLevelOp()
    return True
  except MemoryError:
    return False
```

```python
class HLTTestCase(unittest.TestCase):
  def test_LLO_no_memory(self):
    def mocked_memory_error():
      raise MemoryError('test :-(')

    with mock.patch( # look here!
            '__main__.lowLevelOp',
            mocked_memory_error ):
      self.assertFalse(highLevelTask())

if __name__ == '__main__':
  unittest.main()
```

See https://docs.python.org/3/library/unittest.mock.html

See https://docs.python.org/3/library/unittest.mock.html#patch

Dynamic Mocking Library Uses

- Track how many times a function was called and/or with what arguments ("spying")

  - How would you do this with dynamic mocking?

- Add or remove side effects

  - Exceptions are considered a side effect by mocking libraries

- Test locking in multithreaded code

  - e.g., force a thread to stall after acquiring a lock

Dynamic Mocking Disadvantages

- Test cases with dynamic mocking can be very fragile

  - What if someone moves or removes the call to lowLevelOp() that we mock.patch'd earlier?

- Dynamic mocking requires good integration tests

  - If we mock dependencies, we need to be extra careful that our ADTs play nicely together

- Dynamic mocking libraries have a learning curve

  - In Python, it can be hard to determine the correct value for 'path' in mock.patch (etc.)

  - Error messages are often cryptic (modified program)

Quality Assurance and Development Processes

- How can we assure quality before, during and after writing code?

- What if we don't have enough resources?

  - Tune in next time!

- Further Watching:

  - "So You Want To Be In QA?"

https://www.youtube.com/watch?v=ntpZt8eAvy0
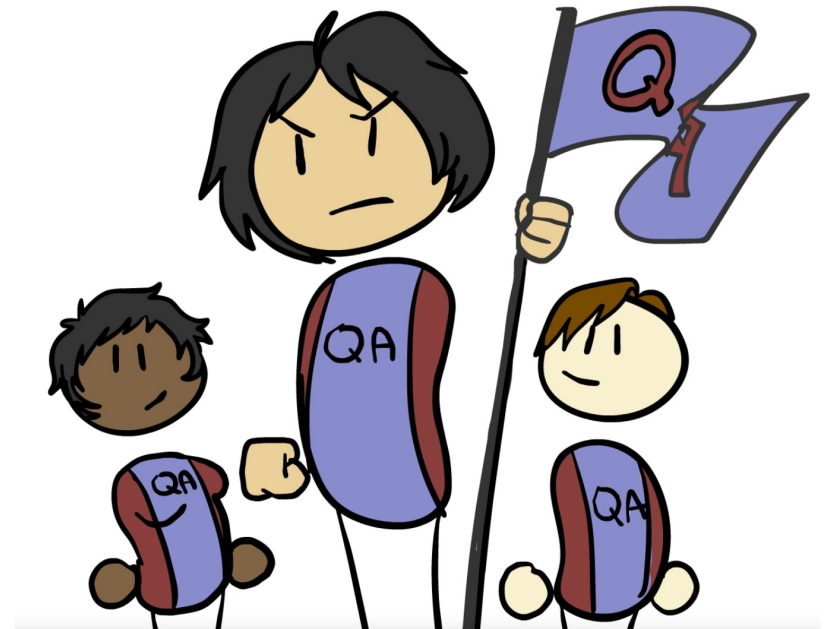
Questions?

- Next exciting episode:
  - Test Suite Quality *Metrics*