Why Do We Test Software?

CS 3250
Software Testing

[Ammann and Offutt, “Introduction to Software Testing,” Ch. 1]
Software is Everywhere
Software Failures

- 2019: Facebook, Instagram, WhatsApp 14 hours downtime due to Facebook News Feed issue in routine maintenance

- 2018: Hawaii Emergency Management Agency sent out a false missile alert due to no visible alterations between live alert and testing environments

- 2018: Pedestrian in Arizona was killed by an Uber car due to its self-driving software failure

- 2018: Google shut down Google+ due to the undetected fault that was present for more than two years, causing nearly 500,000 users’ data to be compromised

- 2018: TSB system upgrade causes months of online banking disruption

- 2017: Cloudflare’s major software fault led to customer sensitive data leakage

- 2017: 606 recorded software failures, impacting 3.7 billion people, 314 companies, $1.7 trillion in financial losses

- 2016: Nissan recalled 4 millions cars from the market due to software failure in the airbag sensory detectors
Software Failures (cont.)

- 2016: Info lost due to the browser back button while using TurboTax software
- 2015: Bloomberg’s trading terminal failures forced the British government to postpone $4.4 billion debt sale
- 2014: Dropbox’s outage was due to a fault in a maintenance script
- 2012: Faults in a new Knight Capital’s trading software causes $440 millions
- 2007: Symantec concluded that most security vulnerabilities are due to faulty software
- 2003: Northeast blackout due to the alarm system in the energy management system failure, affecting 40 million people in 8 US states, 10 million people in Ontario, Canada
- 1999: NASA’s Mars lander crashed due to a unit integration fault
- 1997: Ariane 5 explosion: Exception-handling bug forced self-destruct on maiden flight (64-bit to 16-bit conversion), causing $370 millions
- 1986: 3 patients were killed by Therac-25 radiation machine due to poor testing of its safety-critical software
How Important is Testing?

History of Software Testing

1960s - 1980s
Constraint

1990s
Need

2000+
Asset

What? I've done the coding and now you want to test it. Why? We haven't got time anyway.

OK, maybe you were right about testing. It looks like a nasty bug made its way into the Live environment and now customers are complaining.

Testers! you must work harder! Longer! Faster!

[link: http://ashishqa.blogspot.com/2012/12/history-of-software-testing.html]
Testing in the 21st Century

- Safety critical, real-time software
- Embedded software
- Enterprise applications
- Security
- Web
- Mobile

Software testing becomes more important

We need reliable software. Testing is one way to assess reliability and thus improve quality of software.
Cost of Late Testing

Assume $1000 unit cost, per fault, 100 faults

- Fault origin (%)
- Fault detection (%)
- Unit cost (X)

Software Engineering Institute; Carnegie Mellon University; Handbook CMU/SEI-96-HB-002

Introduction to Software Testing, Edition 2 (Ch 1) © Ammann & Offutt 25

[Chart illustrated by Ammann & Offutt
Source: Software Engineering Institute; Carnegie Mellon University; Handbook CMU/SEI-96-HB-002; page 56-58]
Wrap-up

• Testing is the most time consuming and expensive part of software development

• Not testing is even more expensive

• Having too little testing effort early increases the testing cost

• Planning for testing after development is prohibitively expensive

• A tester’s goal is to eliminate faults as early as possible

• **What’s next?**
  
  • Getting started – intro to software testing