UNIVERSITY OF LINCOLN

When software smells!

Dr John C. Murray

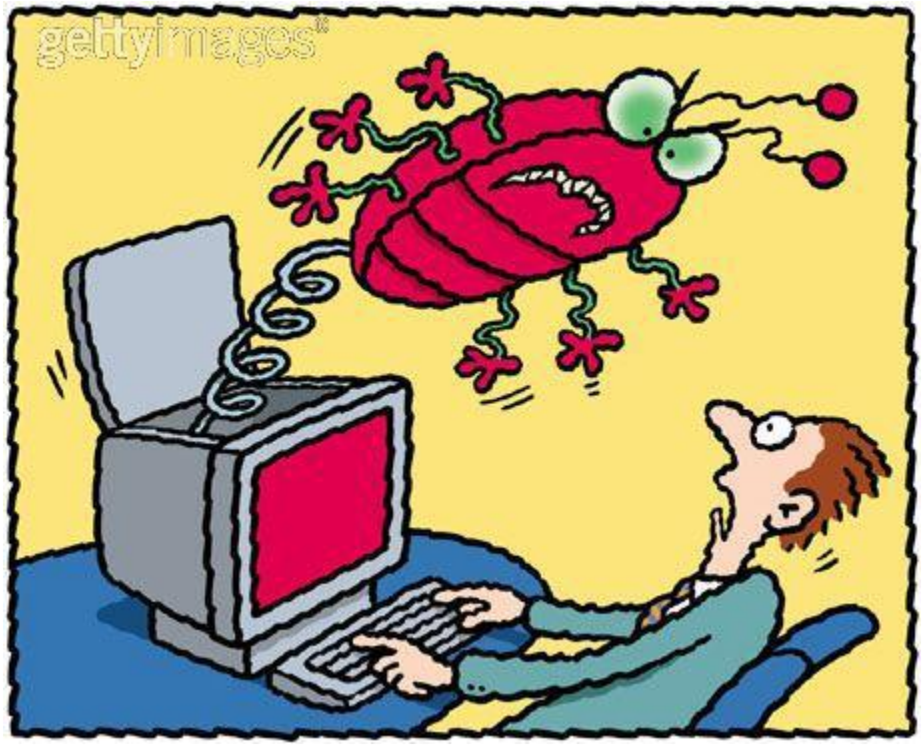
17th November 2016

Assessment Support

- Monday 21st
  - 1pm – MC3107
  - 2pm – Lab B
- Monday 28th
  - 12pm – MC3107
  - 1pm – Lab B

Computer Bugs

- What is a Computer Bug?



Computer Bug

- What was the first computer bug?
- In 1947 Harvard University was operating a room-sized computer called the Mark II.
  - Consisted of mechanical relays
  - vacuum tubes
- A moth flew into the computer and was zapped by the high voltage when it landed on a relay.
  - Hence, the first computer bug!

https://en.wikipedia.org/wiki/Harvard_Mark_II

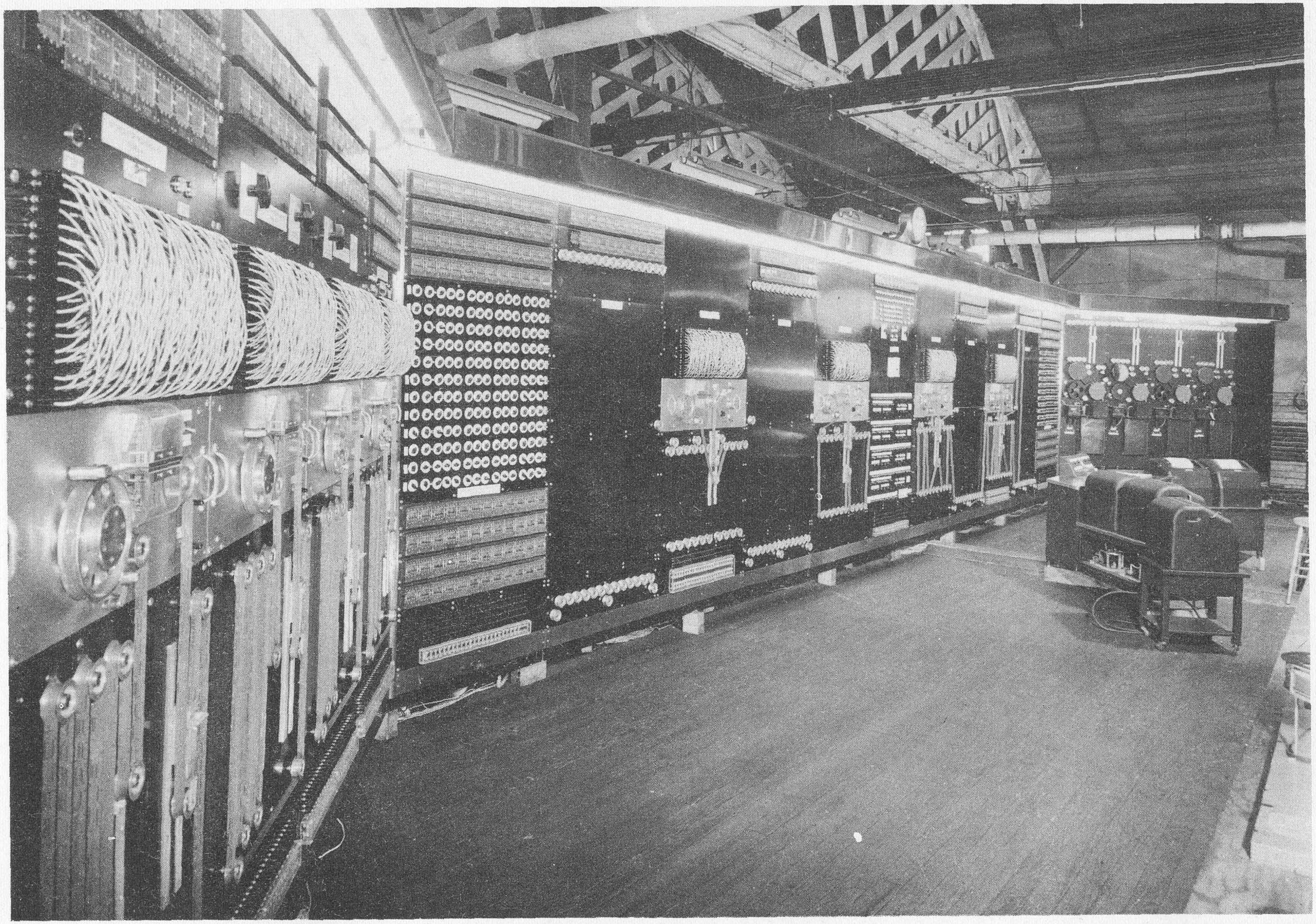


Plate I Main Control Board and Wings

1st Computer Bug

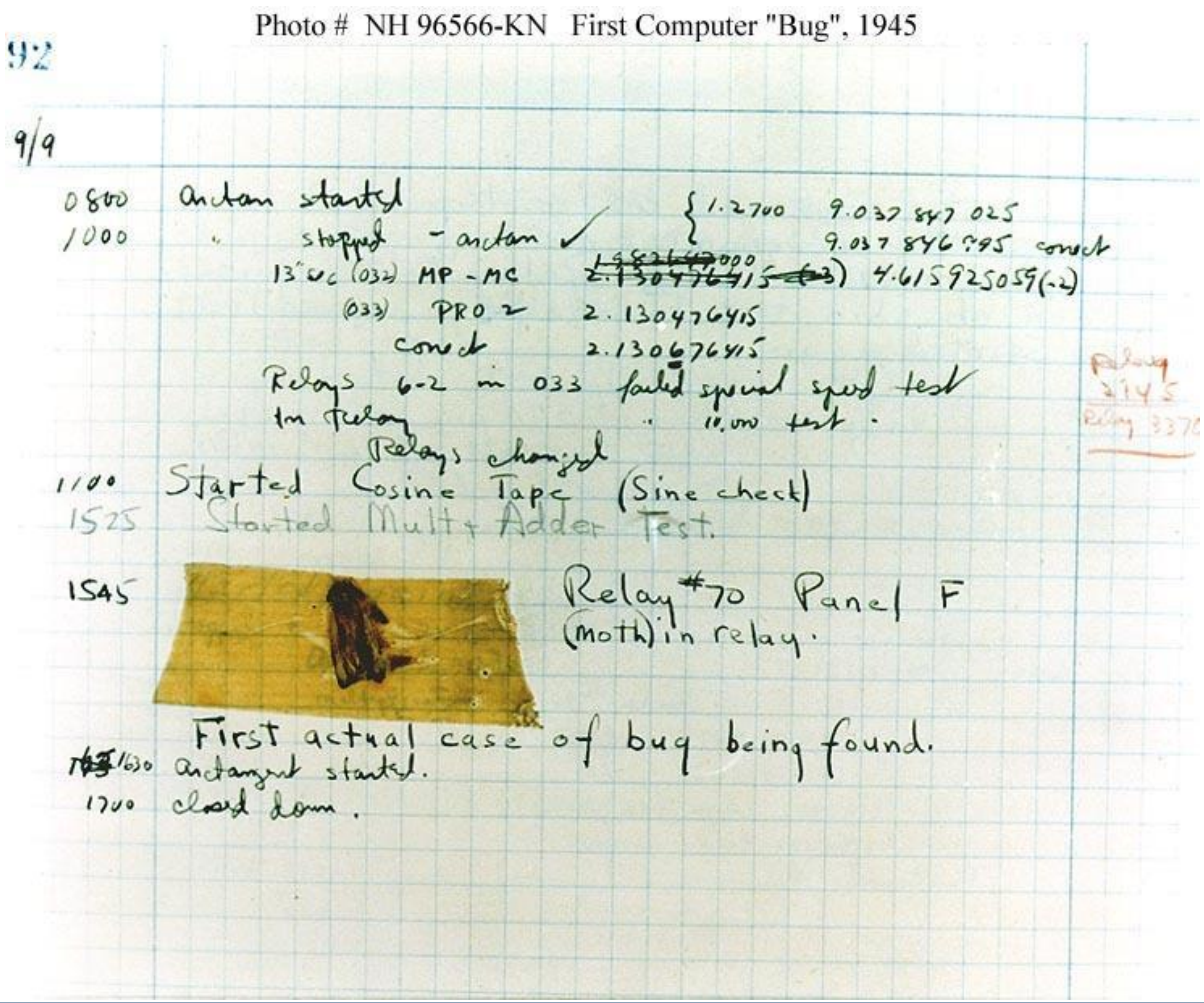


Photo # NH 96566-KN First Computer "Bug", 1945

92

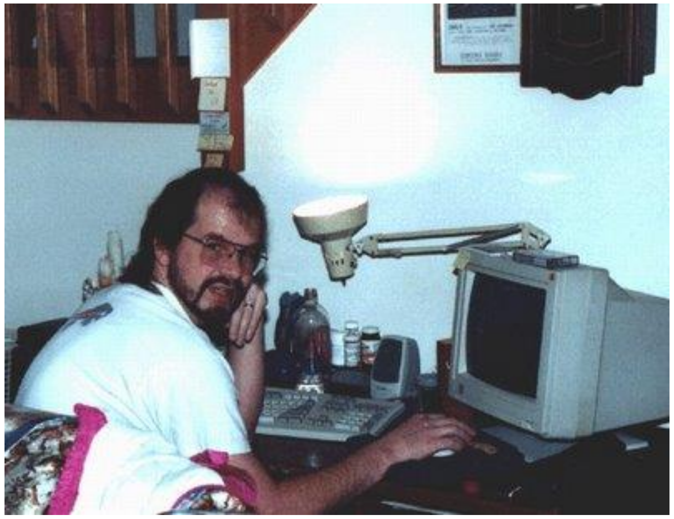
9/9

0800 antan started
1000 " stopped - antan ✓ {1.2700 9.037 847 025
9.037 846 995 conect
13" uc (032) MP - MC ~~1.982647000~~ ~~2.130476415 (-3)~~ 4.615925059(-2)
(033) PRO 2 2.130476415
conect 2.130676415
Relays 6-2 in 033 failed special speed test
In relay " 11.000 test .
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Mult + Adder Test.
1545 Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1630 antangent started.
1700 closed down.

Relay 2145
Relay 3370

Computer Bug

- Now referred to more recently as ‘Software Bugs’
  - Transition in terminology happened automatically
- What causes Computer/Software Bugs?



- Progr
- Anyo

Modern Computer Bugs

- Computer Bugs today can relate to a number of things
  - Defect, Fault
  - Problem, Error
  - Incident, Anomaly
  - Variance, Failure, Inconsistency
  - Product Anomaly
  - And more

Defective Software

- We develop programs that contain defects!
  - How many?
  - We can never know this

- What kind?
  - Again, we don't know, all we can do is perform tests

- Only exhaustive testing can show a program is free from defects....
  - Only exhaustive testing is impossible
    - Why?

Sources of Problems

- What causes software bugs? In context of Software Development?
- **Requirements Definition**
  - Erroneous, incomplete, inconsistent requirements.
- **Design**
  - Fundamental design flaws in the software.
- **Implementation**
  - Mistakes in chip fabrication, wiring, programming faults, malicious code.
- **Support Systems**
  - Poor programming languages, faulty compilers and debuggers, misleading development tools.

Sources of Problems

- **Inadequate Testing of Software**
  - Incomplete testing, poor verification, mistakes in debugging.
- **Evolution**
  - Sloppy redevelopment or maintenance, introduction of new flaws in attempts to fix old flaws, incremental escalation to inordinate complexity.

Are these a problem?

- Are bugs in our system a problem?
  - Why?

PUT MORE BUGS IN THE SOFTWARE! I'M MAKING A FORTUNE OUT HERE!

- We have to consider the implications of the use of the software
  - What is it controlling
  - What are the consequences if it fails?

Case Study - Therac-25 Radiation

- In Texas, 1986, a man received between 16,500-25,000 rads in less than 1 sec, over an area of about 1 cm.
  - He lost his left arm, and died of complications 5 months later.

- In Texas, 1986, a man received at least 4,000 rads in the right temporal lobe of his brain.
  - The patient eventually died as a result of the overdose.

Case-Study – Therac-25

- http://cs.stackexchange.com/questions/30393/asking-for-help-with-this-therac-25-bugged-code-i-dont-understand-the-explanat
- https://en.wikipedia.org/wiki/Therac-25

- Problem was down to software interlocks replacing previous hardware locks.

- If manual data entered a ‘race condition’ could occur
  - Resulting in a failing of the software interlocks

- Race conditions are very difficult to test for
  - What is a Race Condition?

Race Condition Example

- Happens in Multi-Threaded systems
  - Trying to access shared data / attributes

Race Condition Example

- Expected Operation

Thread 1	Thread 2		Integer Value
			0
Read value		←	0
Increment value			0
Write value		→	1
	Read value	←	1
	Increment value		1
	Write value	→	2

Race Condition Example

- Un-Expected Operation

Thread 1	Thread 2		Integer Value
			0
Read value		←	0
	Read value	←	0
Increment value			
	Increment value		
Write value		→	1
	Write value	→	1

Case Study - Bank Generosity

- A software flaw caused a bank in the UK to duplicate every transfer payment request for half an hour.
  - The bank lost £2 billion!
  - The bank eventually recovered the funds but lost half a million pounds in potential interest.

Who is responsible?

- http://yro.slashdot.org/story/10/05/13/0034203/UK-Court-Rules-Company-Liable-For-Software-Defects?from=rss&utm_source=twitterfeed&utm_medium=twitter&utm_campaign=Feed%3A+Slashdot%2Fslashdot+(Slashdot)

- http://www.linuxjournal.com/content/should-software-developers-be-liable-their-code

- "Digital content is not a tangible good and should not be subject to the same liability rules as toasters," BSA director of public policy Francisco Mingorance told ZDNet UK on Thursday. "Unlike tangible goods, creators of digital content cannot predict with a high degree of certainty both the product's anticipated uses and its potential performance."

Specification

- You have to know what your product is before you can say if it has a bug.
- A *specification* defines the product being created and includes:
  - Functional requirements that describes the features the product will support. E.g., on a word processor
    - Save, print, check spelling, change font, …
  - Non-functional requirements are constraints on the product.
    - Security, reliability, user friendliness, platform, …

Questions

?

UNIVERSITY OF LINCOLN