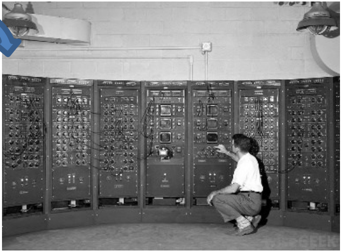
# Software Bugs and Detections

Bowen Zhang

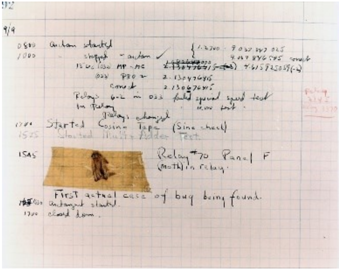
Instructor: KY Wu

April 28 2022

# Why do we call it “bug”?



On 1947, a moth flied into a computer device…
Then the computer couldn’t work…



92

9/9

0800 Arctan started
1000 " stopped - arctan ✓ {1.2700 9.037 847 025
9.037 846 995 correct
13" UC (032) MP - MC ~~1.982147000~~ 2.130476415 ~~(-3)~~ 4.615925059(-2)
(033) PRO 2 2.130476415
correct 2.130676415
Relays 6-2 in 033 failed special speed test
In relay 10,000 test.
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multi Adder Test.
1545 Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1630 Arctangent started.
1700 closed down.

Relay 2145
Relay 3370

The researchers recorded it…
It’s the first “actual” bug being found

# Importance of catching bugs

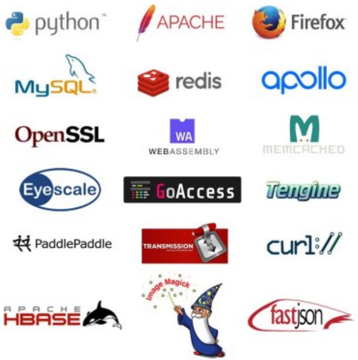
Sometimes your program works well even with a bug

The crash of ARIANE 5:
It's because of an integer overflow bug in the launching program.

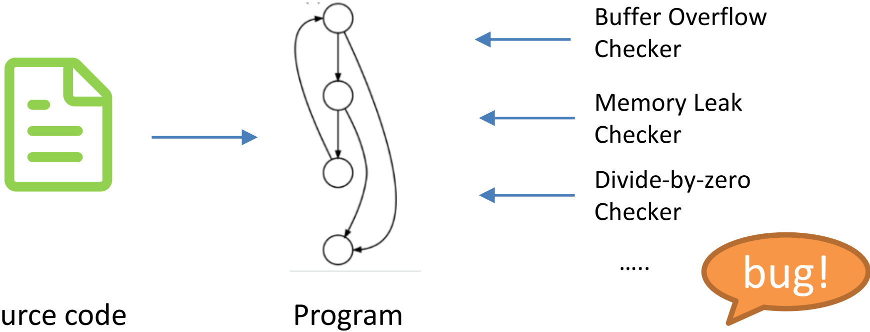
# Static Program Analysis

We analyze the program to discover potential vulnerabilities, without really executing it.

**Pinpoint** static analyzer by our research group.
It has found bugs in:



# A glimpse of Static Analysis: workflow



Buffer Overflow Checker

Memory Leak Checker

Divide-by-zero Checker

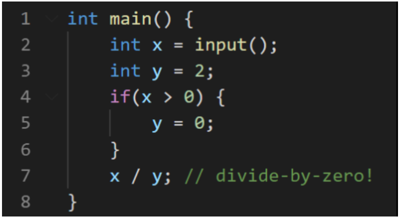
.....

Source code of Program

Program Representation

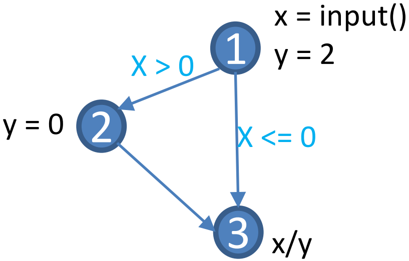
Bug Checkers

# Hunting divide-by-zero bug!



```
int main() {
    int x = input();
    int y = 2;
    if(x > 0) {
        y = 0;
    }
    x / y; // divide-by-zero!
}
```

1. Source code



2. Graph Representation

1 -> 3 safe!

1 -> 2 -> 3 unsafe! (y could be zero)

3. Checker

# My work: Program Representation

Python

C++

Java

Different languages have different representations!

…<br>
C++14<br>
C++17<br>
C++20<br>
…

Different versions could lead to different representations!

**A unified Representation:**
can express different languages
with different versions!

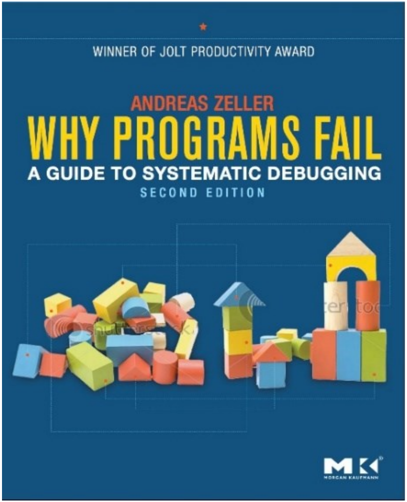
# Take away ideas

How to write less bugs?

1. Don't write all the code before you run it. Instead write code incrementally.
2. Write comments
   - ➢ // this line is for bla bla…
3. When a bug appears, do not try to fix it by randomly changing some code you think is wrong. Instead read through your code and think.

# Thank you

Q & A


WINNER OF JOLT PRODUCTIVITY AWARD
ANDREAS ZELLER
WHY PROGRAMS FAIL
A GUIDE TO SYSTEMATIC DEBUGGING
SECOND EDITION
MK
MORGAN KAUFMANN

*A recommended book about "debug"*