Debugging

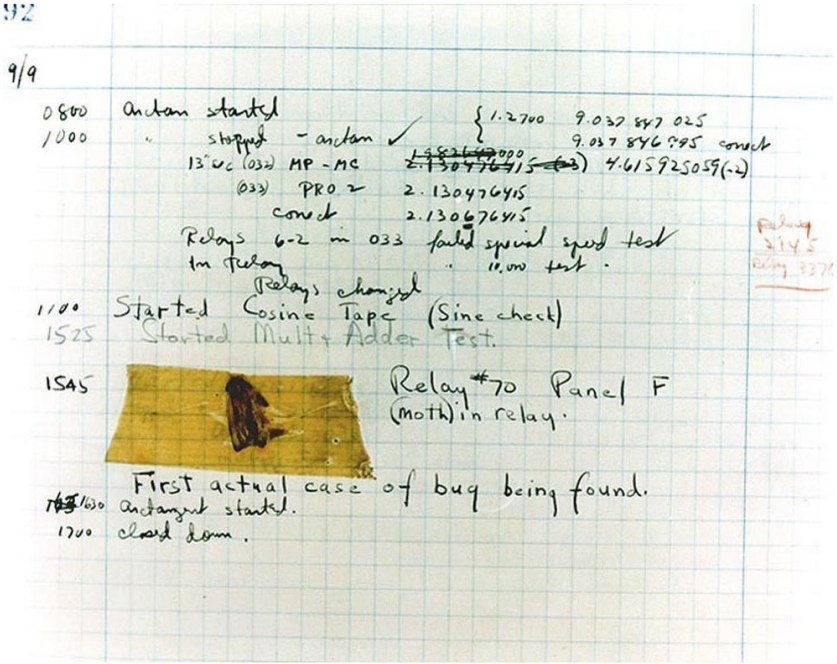
Fall 2023
Mayank Goel

Inspired by content from Michael Hilton, Andreas Zeller, and Stuart Halloway

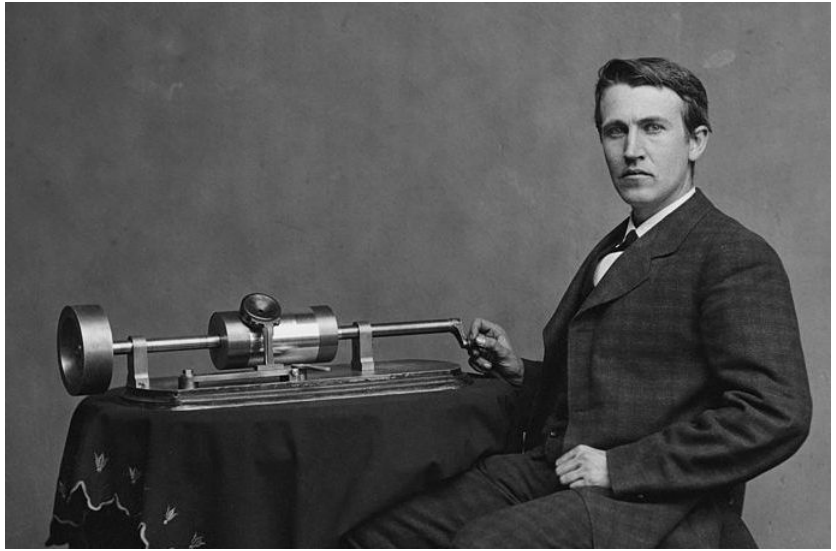
History of Bugs

Dr. Grace Hopper

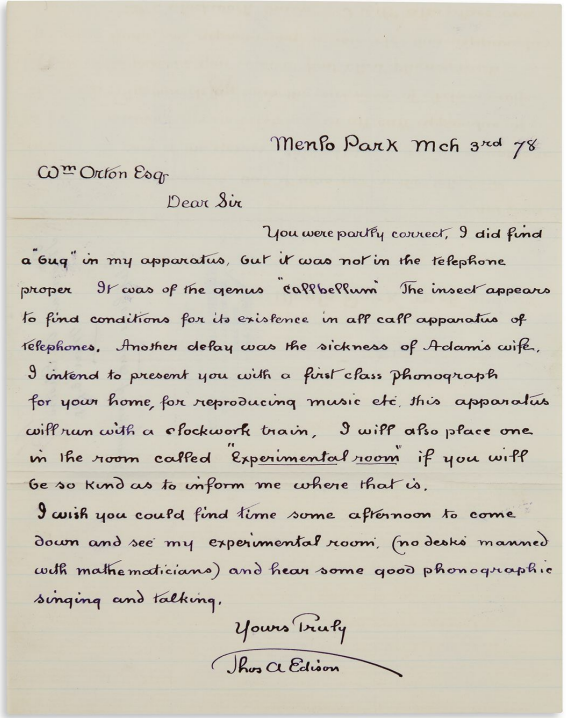
Photo # NH 96566-KN (Color) First Computer "Bug", 1947



History of Bugs



Thomas Edison



Menlo Park Mch 3rd 78

Wm Orton Esq

Dear Sir

You were partly correct, I did find a "bug" in my apparatus, but it was not in the telephone proper. It was of the genus "callbellum" The insect appears to find conditions for its existence in all call apparatus of telephones. Another delay was the sickness of Adam's wife. I intend to present you with a first class phonograph for your home, for reproducing music etc. this apparatus will run with a clockwork train, I will also place one in the room called "Experimental room" if you will be so kind as to inform me where that is.

I wish you could find time some afternoon to come down and see my experimental room, (no desks manned with mathematicians) and hear some good phonographic singing and talking.

Yours Truly

Thos A Edison

Bugs can be Very Hard to Find

tanh(mmalex) @mmalex · 23 Nov 2018

I feel your pain! We had a 'fun' one on LittleBigPlanet 1: 2 weeks to gold, a Japanese QA tester started reliably crashing the game by leaving it on over night. We could not repro. Like you, days of confirmation of identical environment, os, hardware, etc; each attempt took /

4 551 1.6K

tanh(mmalex) @mmalex · 23 Nov 2018

Over 24h, plus time differences, and still no repro. Eventually we realised they had an eye toy plugged in, and set to record audio (that took 2 days of iterating) still no joy. Finally we noticed the crash was always around 4am. Why? What happened only in Japan at 4am? We begged

1 24 373

tanh(mmalex)
@mmalex

Follow

Replying to @mmalex @AeornFlippout

To find out. Eventually the answer came: cleaners arrived. They were more thorough than our cleaners! One hour of vacuuming near the eye toy- white noise- caused the in game chat audio compression to leak a few bytes of memory (only with white noise). Long enough? Crash.

3:32 PM - 23 Nov 2018

Bugs are Hard to Fix

Bugs can be Very Expensive

Bugs are Everywhere!

Types of Bugs

Broadly speaking:

- Compile Errors
- Runtime Errors
- Logic Errors

Let's look at some code.

(runtime_errors_example.c0)

```
#use <coni>

void print_characters(char character, int num)
{
    int i=num;
    while (i>0)
    {
        printf("%c",character);
        i--;
    }
}

int main()
{
    int j = 0;
    int k = 0;
    int sizeOfDiamond = 20;
    for (int i=0; i<sizeOfDiamond;i++)
    {
```

```
mayankgoel@MacBook-Pro-255 C and C0 % cc0 diamond_pattern.c0
Could not find coni.h0
```

Locating Bugs

User or Third-Party Input
Program State
Our Code
Compiler
Operating System
Hardware
Physics

Locating Bugs: Ockham's Razor

Whenever you have competing theories for how some effect comes to be, *pick the simplest.*

What NOT to say when You Encounter an Error?

"That's a weird error!"

"That should have worked!"

"I bet it is because of….."

That's a bit too pedantic!

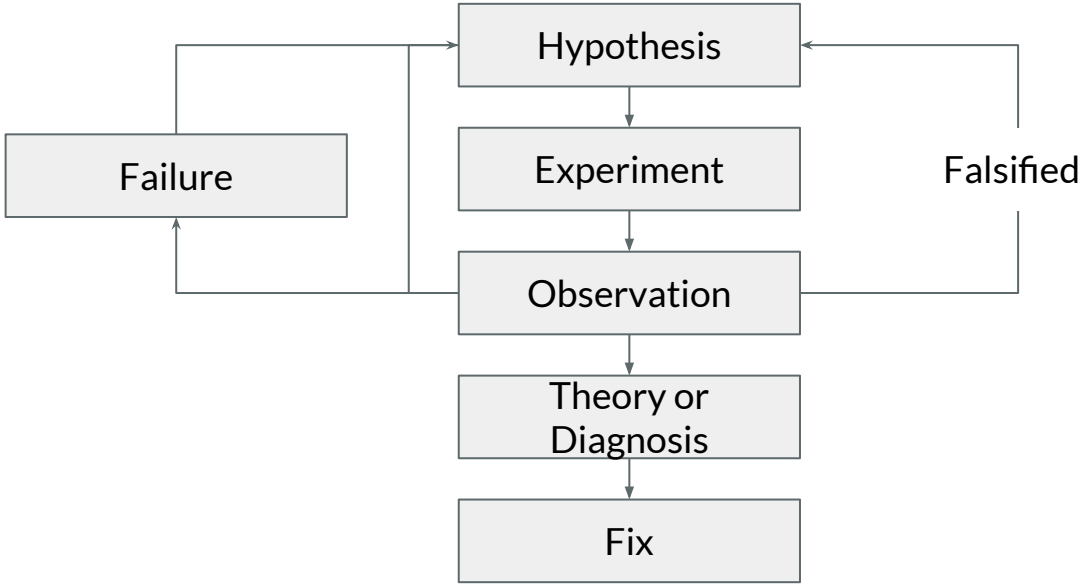
The idea is do not rely too much on intuition.

It takes time and cannot be guaranteed.

So, What Should We Do?

Start by asking: “I wonder what could cause this.”

Scientific Debugging



Overkill? After some experience, maybe spend 10-15 minutes hacking and trying things out.

How To Come Up With A Hypothesis

Think of 20 questions.

Do not start with something super specific.

Resources to generate a hypothesis:

- Error messages
- Deducing from the code *How?*
- Program output *How?*
- Multiple trials/runs/outputs
- Earlier hypotheses

Deducing from the Code

- Code review
- Rubber duck method
- Flowcharts / Pseudo-codes

Program Output

Print Debugging

Debugger tools to step through code - *later*.

Write It All Down

Hypothesis	
Prediction	
Experiment	
Observation	
Conclusion	

Let Us Look At An Example

Code to generate factorial of a number.

(debug_factorial.c0)

Initial Hypothesis

Hypothesis	n = 5, the factorial should be 120
Prediction	Output is 120
Experiment	Run the code
Observation	Compile error
Conclusion	Hypothesis is rejected

Experiment

Hypothesis	The header file is misnamed
Prediction	Renaming header file will fix the problem
Experiment	Rename the header file
Observation	
Conclusion	

Experiment

Hypothesis	The header file is misnamed
Prediction	Renaming header file will fix the problem
Experiment	Rename the header file
Observation	A different error for an uninitialized variable.
Conclusion	Header file was misnamed. Variable must be uninitialized.

Experiment

Hypothesis	Variable result is uninitialized.
Prediction	Initializing result will compile code.
Experiment	Initialize result
Observation	
Conclusion	

Experiment

Hypothesis	Variable result is uninitialized.
Prediction	Initializing result will compile code.
Experiment	Initialize result
Observation	Code compiles
Conclusion	Variable result was not initialized.

Back to our Original Hypothesis

Hypothesis	n = 5, the factorial should be 120
Prediction	Output is 120
Experiment	Run the code
Observation	
Conclusion	

Experiment

Hypothesis	n = 5, the factorial should be 120
Prediction	Output is 120
Experiment	Run the code
Observation	Output is 0
Conclusion	Hypothesis rejected

Experiment

Hypothesis	The value of n is not preserved when I call factorial function.
Prediction	n at the invocation of factorial is not 5
Experiment	Observe value of n
Observation	
Conclusion	

Experiment

Hypothesis	The value of n is not preserved when I call factorial function.
Prediction	n at the invocation of factorial is not 5
Experiment	Observe value of n
Observation	Value of n at the beginning of factorial is 5
Conclusion	Hypothesis rejected

Experiment

Hypothesis	result is initialized to 0 and is causing all future multiplications to result in 0
Prediction	Changing result's initial value to 1 should make the output correct
Experiment	Set result's initial value to 1
Observation	
Conclusion	

Experiment

Hypothesis	result is initialized to 0 and is causing all future multiplications to result in 0
Prediction	Changing result's initial value to 1 should make the output correct
Experiment	Set result's initial value to 1
Observation	As predicted
Conclusion	Hypothesis is confirmed.

Why Document?

- Documentation for the future you, other programmers
- Helps you revisit the problem and code later (perhaps years later in many cases)
- Allows you to “sleep on it”
- As you progress, complexity increases and recollection decreases

One more example for the class exercise

(diamond_pattern.c0)

```
      **
     *  *
    *    *
   *      *
  *        *
 *          *
*            *
*            *
 *          *
  *        *
   *      *
    *    *
     *  *
      **
```