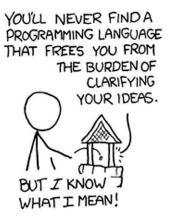
Beyond the Mouse – A Short Course on Programming 11. Debugging Solving Major (and minor) Crises

Ronni Grapenthin

Geophysical Institute, University of Alaska Fairbanks

November 19, 2009



"The Uncomfortable Truths Well", http://xkcd.com/568 (April 13, 2009)

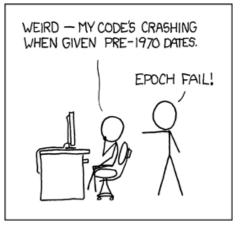
- Design
- 2 Coding
- Test
- Oebugging
- go back to 1,2, or 3, ...

What is a bug?

A mistake in a computer program ... or:

1 de 9/9 andon started 0800 1.2700 9.037 847 025 1000 stornet antan " 9.037 846 95 const +56415 (-2) 4.615925059(-2) 13"0 (032) MP - MC (033) PRO 2 2. 130476415 const 2.130676415 Reas fould special speed test m 033 In telo Started 1100 (Sine check) Adder Te Relay #70 Panel F (moth) in relay. 1545 143100 Andargul started. of bug being found. 1700 cloud dom.

The First "Computer Bug", U.S. Naval Historical Center Photograph, NH 96566-KN 3



"Bug", http://xkcd.com/376

- bad / unexpected values: check user / function input!
- messed up logic: want x programmed y.
- unwarranted assumptions: units.
- rarely moths, usually people.

Debugging is the **art** of finding and fixing mistakes in computer programs. To be successful you need insight, creativity, logic, and determination.

Debugging is the **art** of finding and fixing mistakes in computer programs. To be successful you need insight, creativity, logic, and determination.

Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.

Brian Kernighan

- Bugs are static they won't run away!
- Often, the problem is **simple**.
- You created the bug! It's nobody else's fault suck it up!
- Be critical did you mean '<', '<=', '>', '>='?
- Don't panic be systematic!
- Sleep, go for a walk, come back later.

- echoing: place print statements at useful points in a program (function entry, exit)
- **unit testing**: write calls to particular function, throw artificial values at it
- exception handling: in high level languages: sources of mistakes easier to spot
- **online debuggers**: for our purposes not necessary, useful if you want to step through your code, or for memory problems

Debugging Styles: echoing

... we find stepping through a program less productive than thinking harder and adding output statements and self-checking code at critical places. Clicking over statements takes longer than scanning the output of judiciously-placed displays. It takes less time to decide where to put print statements than to single-step to the critical section of code, even assuming we know where that is. More important, debugging statements stay with the program; debugging sessions are transient.

From: Brian Kernighan, Rob Pike "The Practice of Programming"

Debugging Styles: echoing

... we find stepping through a program less productive than thinking harder and adding output statements and self-checking code at critical places. Clicking over statements takes longer than scanning the output of judiciously-placed displays. It takes less time to decide where to put print statements than to single-step to the critical section of code, even assuming we know where that is. More important, debugging statements stay with the program; debugging sessions are transient.

From: Brian Kernighan, Rob Pike "The Practice of Programming"

- write method that displays text only if a global DEBUG flag is set
- find ways to implement such external switches for SHELL: environment vars, Matlab: create your own preferences
- call this method whenever necessary: entry, exit of functions, to display certain values, to follow the program flow, ...

...see btm_schedule demo ...

- at the simplest: write calls to your functions with artificial values
- execute these calls at the beginning of your code, check function results
- this helps to detect errors due to changes in functions immediately
- also: assertion that function works for tested TYPES
- can be done for any language (some languages come with fancy frameworks)

...see btm_schedule demo ...

Debugging Styles: exception handling

Full exception handling support in Matlab:

Matlab - try-catch

```
% trv. STATEMENT, catch ME, STATEMENT, end.
%
% EXAMPLE: file opening
try
          = fopen('whatever.txt', 'r'); % open a non-existing file
    fid
    data = fread (fid);
                                              % now try to get its data
                                              % any name for error message object
catch myException
    %let the user know, implement graceful program termination ...
    disp(myException);% display full error objectdisp(myException.message);% actual message is more accessibledisp(myException.stack);% where did things occur?
end
disp('We do get here!')
%now without try-catch
fid = fopen('whatever.txt', 'r');
data = fread(fid);
disp('We cannot get here!')
```

... see demo ...