debugging and testing

Ben Bolker

22:27 22 January 2015

Errors

Types of errors

- syntax errors vs. logic errors
- failure modes:
  - obvious failure
    - program stops with an error partway through: bad mandelbrot #0
    - Python crashes
    - machine crashes
    - program never stops (infinite loop)
  - wrong answer
    - always
    - occasionally

Next section follows this presentation

- infinite loops: bad mandelbrot #1
- incorrect operator precedence, e.g. $\Delta\text{fahrenheit} = \Delta\text{Celsius} \times 1.8$

```
fahrdiff = celsius_high - celsius_low * 1.8
```

- off-by-one error ("fencepost problem")
- . . . more generally, **edge** or **corner cases**
- incorrect code inside/outside loops:
- bad mandelbrot #2
- bad mandelbrot #3
- bad mandelbrot #4
- array index error (outside bounds)

Error messages

- error messages are *trying* to tell you something
- Google error messages (with quotation marks)

Debugging

- *brute-force logic*: stare at your code, try to figure out what's wrong (test cases really help: why is it failing in one specific situation?)
- flow charts, *pseudocode*
- `print()`
- interactive tracing
- debugging tools (breakpoints/watchpoints/watches)
- checkpointing

Searching for/asking for help

Searching for help

- Google (or your search engine of choice)
- be as specific as possible

Asking for help

- reproducible/minimal workable examples
  - right amount of context
  - "how to ask" (StackOverflow)
- browse/lurk in forums first!
- tone
- where:
  - forums
  - StackOverflow

Testing

- Simplify, simplify, simplify
- Reduce the size of your problem
- Cases with easy/known answers
- Corner & edge cases
- Random tests (fuzz testing)
- Automatic testing framework: `nose`
  - built-in Python package
  - define test file
    - basic: `assert <condition>`

    - extra: `from nose.tools import assert_equal, assert_raises` (or something)
    - (generating an error: `raise ErrorType("message")`, e.g. `raise ValueError("non-conformable matrices")`
    - each test or set of tests as a separate function
    - see test_mm.py
  - `nosetests`/run in PyCharm
- Test-driven development: write tests *first*!

Additional resources

- http://stackoverflow.com/questions/1623039/python-debugging-tips
- https://www.udacity.com/course/cs259
- http://www.cs.yale.edu/homes/aspnes/pinewiki/C%282f%29Debugging.html
- http://www.cs.cf.ac.uk/Dave/PERL/node149.html