Testing

Luis Pedro Coelho

Programming for Scientists

October 22, 2012
Defensive programming means writing code that will catch bugs early.
def trim(qs, thresh):
   
   assert thresh >= 0, 'threshold should be positive'
assert <condition>, <error message>
Do you test your code?

Of course you do, interactively, informally. Here we are only going to make this automatic.
Testing trim

```python
import numpy as np
from trimfq import trim

qs = np.array([])
trim(qs, 20)

qs = np.array([20, 20])
trim(qs, 20)
```

These simple sort of tests are called smoke tests.

A “real” smoke test is when you put smoke through pipes and check for leaks.
Testing trim

```python
import numpy as np
from trimfq import trim

qs = np.array([])
trim(qs, 20)

qs = np.array([20, 20])
trim(qs, 20)

These simple sort of tests are called smoke tests.
```
Testing trim II

qs = np.array([10, 10, 10, 20, 20, 20, 20, 10])
s, e = trim(qs, 15)
assert np.all(qs[s:e] >= 15)
assert s < e
Where are errors likely to lurk?
Testing trim III

Where are errors likely to lurk?

- At the edges?
- What if the whole string is above threshold?
- What if the whole string is below threshold?
s, e = trim(np.array([10, 10, 10, 10]), 5)
assert s == 0
assert e == 4

s, e = trim(np.array([10, 10, 10, 10]), 15)
assert s == e  # Note that we
    # DO NOT care about
    # actual values
Testing trim V

```python
s, e = trim(np.array([10, 10, 20, 20]), 15)
assert s == 2
assert e == 4

s, e = trim(np.array([20, 20, 10, 10]), 15)
assert s == 0
assert e == 2
```
Fencepost Errors

If you build a straight fence 100 meters long with posts 10 meters apart, how many posts do you need?
Fencepost Errors

If you build a straight fence 100 meters long with posts 10 meters apart, how many posts do you need?

Eleven, but we often think 10.
What is the use of testing?

- Ok, I tested it
- It seems to work
- Now, I am happy
What is the use of testing?

- Ok, I tested it
- It seems to work
- Now, I am happy
- But save those tests!
When your code changes

- When your code changes...
When your code changes...

...you rerun your tests.

Over time, you will accumulate a collection of tests.
Software Testing Philosophies

1 Test everything. Test it twice.
2 Write tests first.
3 Regression testing.
Regression Testing

Make sure bugs only appear once!
Many utilities already exist to help manage test suites (A test suite is a fancy name for “a bunch of tests). In Python, nose is the most popular one.