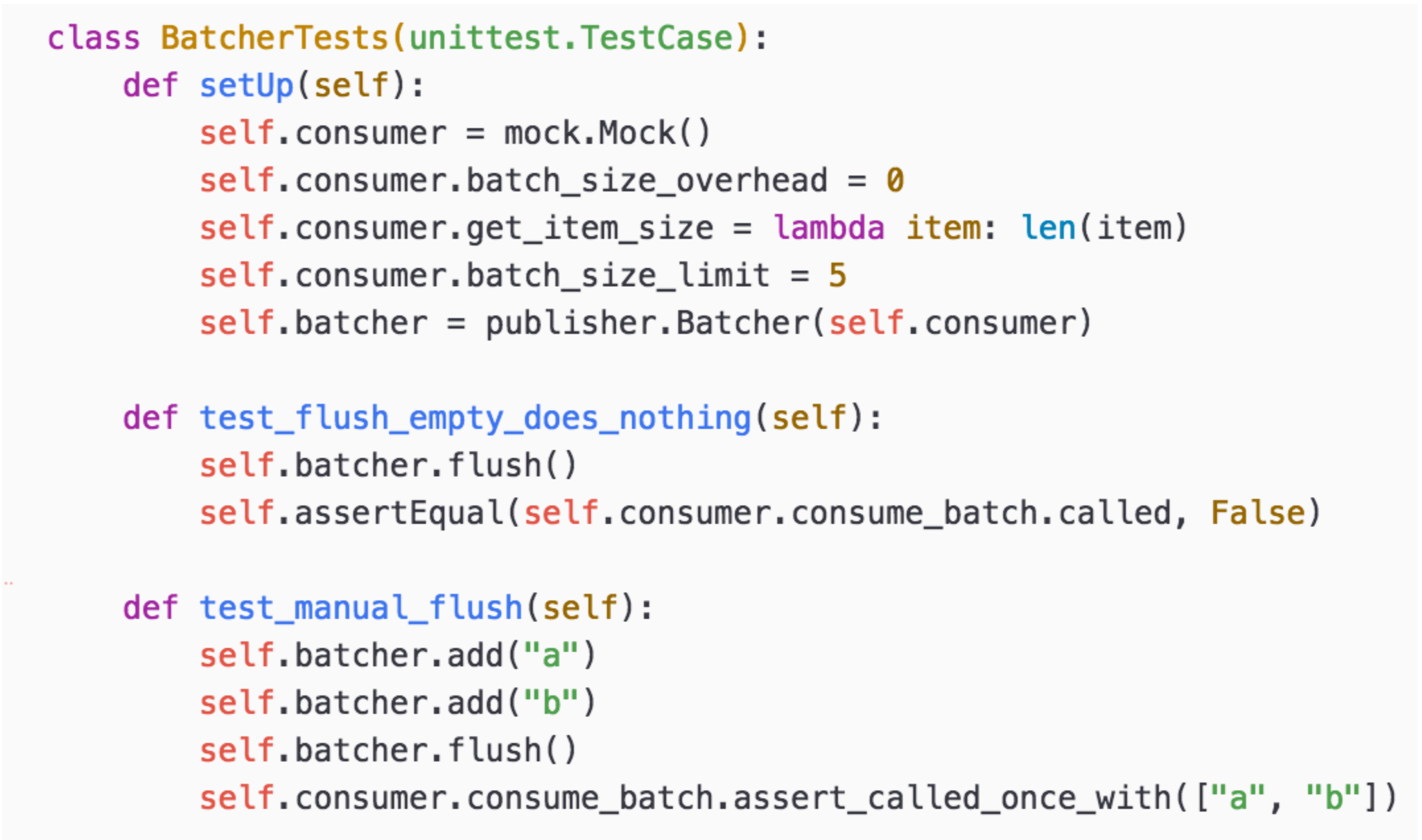
Example 1 - Stub the Stubbable



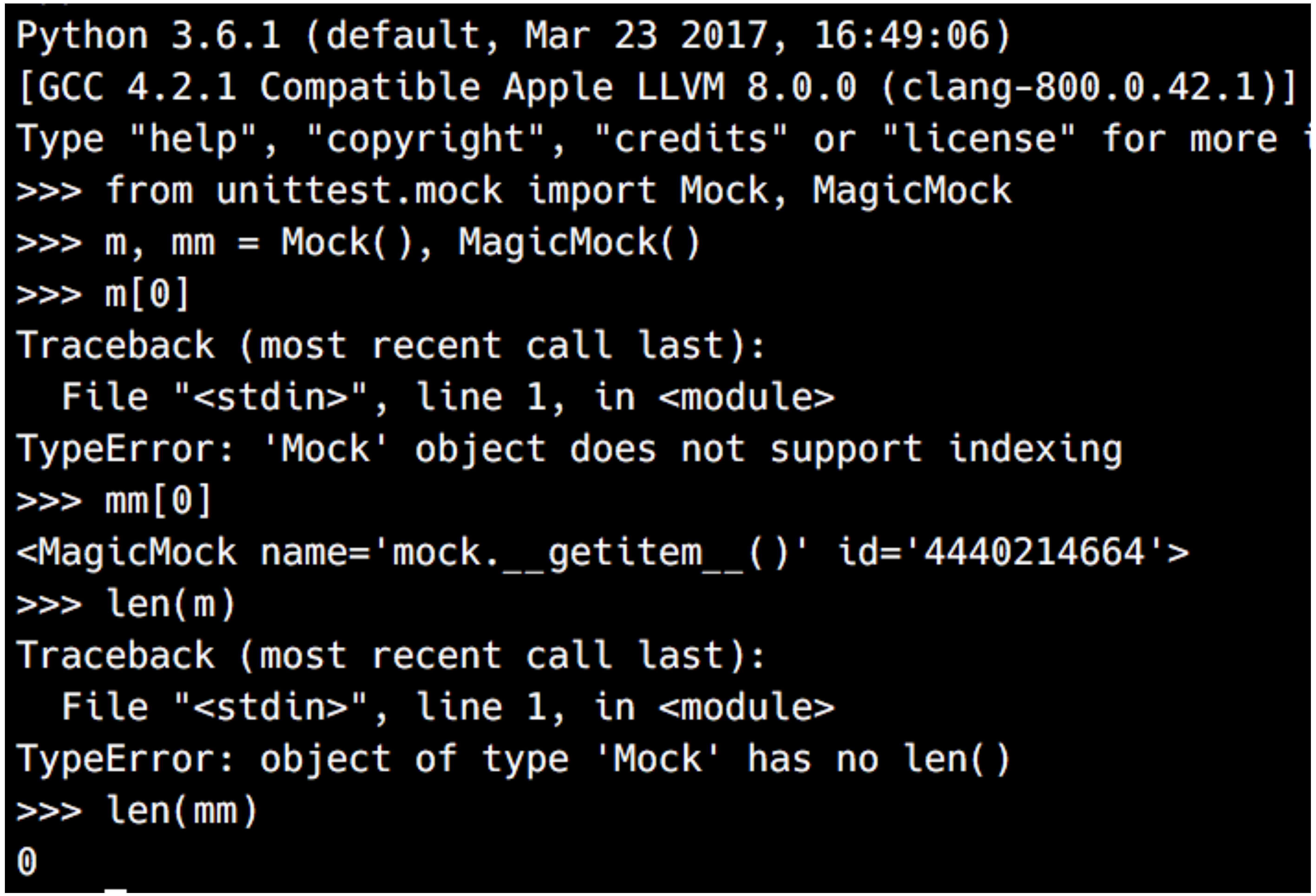
```
class BatcherTests(unittest.TestCase):
    def setUp(self):
        self.consumer = mock.Mock()
        self.consumer.batch_size_overhead = 0
        self.consumer.get_item_size = lambda item: len(item)
        self.consumer.batch_size_limit = 5
        self.batcher = publisher.Batcher(self.consumer)

    def test_flush_empty_does_nothing(self):
        self.batcher.flush()
        self.assertEqual(self.consumer.consume_batch.called, False)

    def test_manual_flush(self):
        self.batcher.add("a")
        self.batcher.add("b")
        self.batcher.flush()
        self.consumer.consume_batch.assert_called_once_with(["a", "b"])
```

- **Notice a few things:**
  - You can set attributes on Mocks
  - You can see if they Mock was called
  - You can see how many times it was called
  - And what it was called with!!!

Example 2 - ✨MagicMock✨



```
Python 3.6.1 (default, Mar 23 2017, 16:49:06)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)]
Type "help", "copyright", "credits" or "license" for more
>>> from unittest.mock import Mock, MagicMock
>>> m, mm = Mock(), MagicMock()
>>> m[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'Mock' object does not support indexing
>>> mm[0]
<MagicMock name='mock.__getitem__()' id='4440214664'>
>>> len(m)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: object of type 'Mock' has no len()
>>> len(mm)
0
```

- Like Mock, but **acts more like an instance of an Object out of the box**
- For example, implements methods to act like it supports indexing and length calculations
- Helpful, but **easily abused by the lazy… (we'll see)**