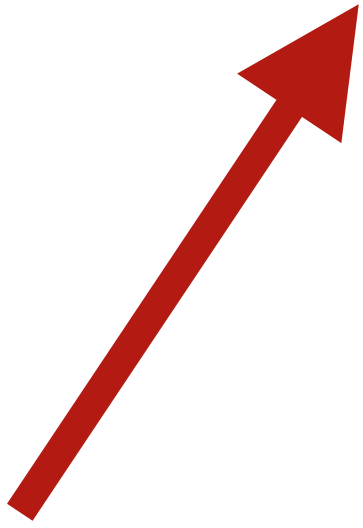
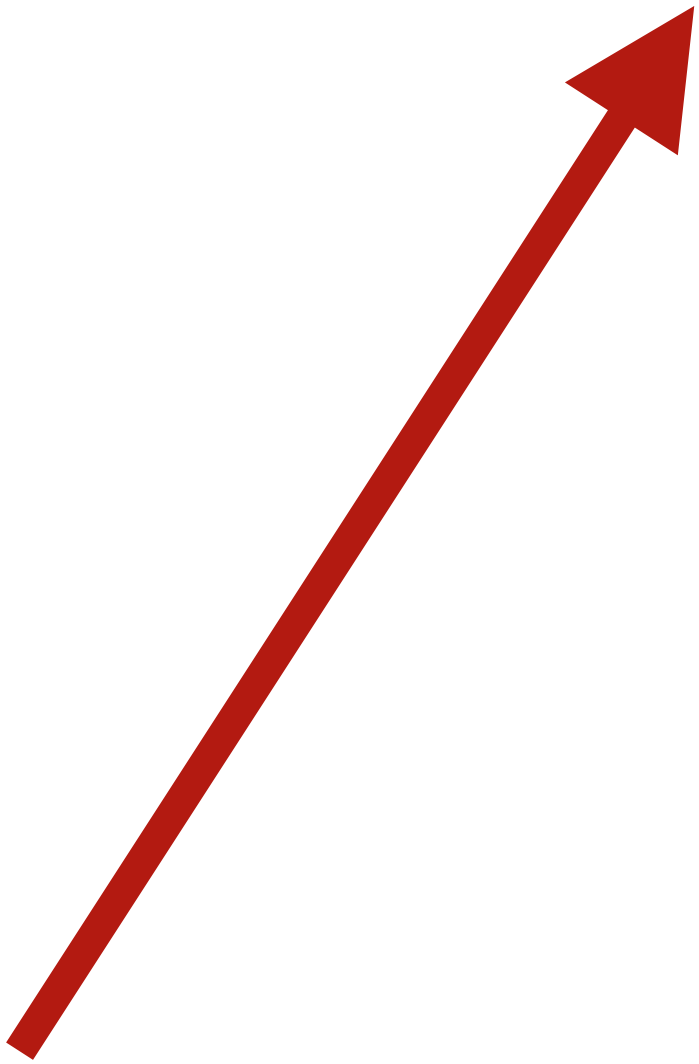
```python
from unittest.mock import patch, call
from my_module import total_value

def test_total_value():
    with patch('my_module.db_read') as mock_read:
        mock_read.return_value = [100, 200]

        assert 300. == total_value(“karl@python.de”, 92)
        assert [call(“karl@python.de”, inv_no=92)] == mock_read.calls
```



mock validates interaction



stub returns canned values

Option: Patch a Mock

```
with patch('my_module.db_read') as mock_read:
```

```
from my_module import total_value
```

```
assert 300. == total_value(“karl@python.de”, 92)
```

```
assert [call(“karl@python.de”, inv_no=92)] == mock_read.calls
```

```
from unittest.mock import patch, call
```

```
def test_total_value():
```

```
mock_read.return_value = [100, 200]
```

Option: Inject a Mock

core.py

```
items = db_read(email, inv_no=invoice_id)
```

```
def total_value(email, invoice_id, discount=0.0):
```

```
from db import db_read
```

```
return sum(items)*(1.0-discount)
```