**QLC-2.4) – Stubs to Mocks**

**Solution**

This solution uses a mock to overcome the limitations of using a stub.

* [[“Physics”, [56, 67, 45, 89]], [“Art”, [87, 66, 78]], [“Comp Sci”, [45, 88, 97, 56]]], the result should be [[“Physics”, 89],[“Art”, 87],[“Comp Sci”, 97]]

1. **Write your Test Highest score of many items (RED)**

Write a new test that finds the highest score for all topics using a mock.

* Name the test:

test\_find\_highest\_score\_with\_list\_of\_many\_returns\_list\_of\_many\_using\_mocks

**from unittest.mock import Mock**

def test\_find\_highest\_score\_with\_list\_of\_many\_returns\_list\_of\_many\_using\_mocks(self):

# Arrange

physics\_scores = [56, 67, 45, 89]

art\_scores = [87, 66, 78]

compsci\_scores = [45, 88, 97, 56]

topic\_scores = [

TopicScores("Physics", physics\_scores),

TopicScores("Art", art\_scores),

TopicScores("Comp Sci", compsci\_scores)

]

**# Create mock for HighestNumberFinder**

**hnf\_mock = Mock()**

**hnf\_mock.find\_highest\_number.side\_effect = [89, 87, 97]**

# System under test

cut = TopicManager(hnf\_mock)

# Expected results

expected\_result = [

TopicTopScore("Physics", 89),

TopicTopScore("Art", 87),

TopicTopScore("Comp Sci", 97)

]

# Act

result = cut.find\_topic\_high\_scores(topic\_scores)

# Assert

for res, exp in zip(result, expected\_result):

self.assertEqual(res.get\_topic\_name(), exp.get\_topic\_name())

self.assertEqual(res.get\_top\_score(), exp.get\_top\_score())

* Ensure test passes
* Commit code to Git

1. **Write minimal Production Code (GREEN)**

There should be no need to modify the production code.

1. **Refactor Code - Optional.**

No refactoring required at this stage.

* Re-Run ALL tests to confirm no Regression.
* Commit code to Git, if not done in previous step.