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

2. **Write minimal Production Code (GREEN)**
   There should be no need to modify the production code.

3. **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.