Unit Testing 2 Solutions

1. We don't need to do anything special here, other than write out a test function, which calls the function we're testing.

```
def add_two_numbers(a, b):
    return a + b

def test_adds_two_whole_numbers():
    # Arrange
    expected = 2

    # Act
    actual = add_two_numbers(1, 1)

    # Assert
    assert expected == actual

test_adds_two_whole_numbers()
```

2. As our function is dependent on another function, we need to use dependency injection. To do this, we can pass the function call for `get_random_number` as an argument. This means that when it comes to testing, we can pass in a mocked version of it.

```
from random import randint
def get_random_number():
    return randint(1, 10)

def add_number_with_random_number(a, get_random_number):
    return a + get_random_number()

def test_add_number_with_random_number():
    # Arrange
    a = 5
    expected = 10
    def mock_get_random_number():
        return 5

    # Act
    actual = add_number_with_random_number(a, mock_get_random_number)

    # Assert
    assert expected == actual

test_add_number_with_random_number()
```

3. The `add_two_random_numbers()` function calls the same function twice, so we still only have one dependency. Just as in the previous exercise, we need to pass `get_random_number()` as an argument so we can mock it in our test.

```
from random import randint
def get_random_number():
    return randint(1, 10)

def add_two_random_numbers(get_random_number):
    return get_random_number() + get_random_number()

def test_add_two_random_numbers():
    # Arrange
    expected = 10
    def mock_get_random_number():
        return 5

    # Act
    actual = add_two_random_numbers(mock_get_random_number)

    # Assert
    assert expected == actual

test_add_two_random_numbers()
```

4. This time we need to mock a function that we haven't written. Luckily it's still the same steps to take and just as easy. We simply need to mock the function and pass it as an argument.

```
from random import randint

def get_random_number(randint):
    return randint(1, 10)

def test_get_random_number():
    # Arrange
    expected = 5
    def mock_randint(a, b):
        return 5

    # Act
    actual = get_random_number(mock_randint)

    # Assert
    assert expected == actual

test_get_random_number()
```