Congratulations! You passed!

Grade received 100% **Latest Submission Grade** 100% **To pass** 80% or higher

Go to next item

1. What is one drawback of using **`unittest`** , Python's standard library testing framework for testing? **1 / 1 point**

- ○ That it requires a separate test runner to execute tests
- ○ That running tests are slower than other test frameworks
- [✓ That it forces class inheritance which requires to memorize many assert methods]

Correct
Correct! The **`unittest`** library has more than 30 assert methods and requires using inheritance.

2. What are three benefits of using the Pytest framework? **1 / 1 point**

○
1. You can use the **`unittest.main`** runner for powerful reporting
2. You can enhance the testing with plugins
3. It can run exclusively with functions

[✓
1. It doesn't require using class inheritance, making it easier to write tests
2. It is a test runner as well as a test framework, although you aren't required to use the framework
3. That it has rich error reporting, making it easier to debug failures
]

- ○
  1. It doesn't require to remember many assert methods and can use only the assert statement
  2. Increased test coverage due to powerful test assertions
  3. It can run exclusively with test methods

✓ **Correct**
Correct! The Pytest framework and test runner makes it easier to write tests.

3. What are two conditions for tests to automatically be discovered by Pytest? **1 / 1 point**

- [✓]
  1. Files need to be prefixed with **`test_`**
  2. Test Functions and test methods need to be prefixed with **`test_`**
- ○
  1. Tests have to inherit from **`unittest.TestCase`**
  2. Files need to exist in a **`test`** directory
- ○
  1. Files need to have the **`_test.py`** suffix
  2. Test functions and test methods need to use the **`_test()`** suffix

✓ **Correct**
Correct! These prefixes are necessary for Pytest to find the tests and run them automatically.

4. What is a valid statement about writing Pytest tests? **1 / 1 point**

- [✓ Pytest supports writing tests in functions as well as classes and test methods.]
- ○ Pytest allows you to write tests in functions as long as they use the **`@pytest`** decorator
- ○ It is a requirement to **`import pytest`** in tests that are written for the Pytest framework

✓ **Correct**
Correct! One benefit is that you can write tests in functions which other frameworks like **`unittest`** do not support.

5. What is the difference between **`setup_class`** and **`setup`** methods? **1 / 1 point**

- ○ That **`setup_class`** is executed after a test in a class and happens just once, and **`setup`** is executed after every test in the class.
- ● That **`setup_class`** is executed before a test in a class and happens just once, and **`setup`** is executed before every test in the class.
- ○ That **`setup_class`** is executed before a test in a class, and **`setup`** is executed before every test in a module.

✓ **Correct**
Correct! You can use these special methods to run code before all tests in a class or before each one.

6. When it is useful to use **`parametrize`** in tests? **1 / 1 point**

- ○ When you are asserting different results with the same inputs
- ● When you are asserting the same result from different inputs
- ○ When you need to parallelize tests with different inputs

✓ **Correct**
Correct! You can write the same test but give it different inputs for less repetitive code.

7. What is one benefit of using plain asserts with Pytest? **1 / 1 point**

- ● That you only need to remember one statement to do assertions while leveraging all of Python's comparison operators
- ○ That you can have shorter error reporting that makes it easier to debug failures
- ○ That you can get faster test execution and reporting when there are failures

✓ **Correct**
Correct! By using the pre-existing operators in Python, you can perform all kinds of comparison operations instead of learning other new

methods to perform the comparisons.

8. What are Pytest fixtures? **1 / 1 point**

- ○ Fixtures allow richer error reporting with Pytest and are requested using arguments in tests
- ● Pytest fixtures are used as arguments in tests, they are helpers that allow code reuse.
- ○ Fixtures are used for parallelizing tests and are used as arguments

✓ **Correct**
Correct. Pytest fixtures allow code re-usability and are requested as arguments.

9. What is one of the benefits of testing software? **1 / 1 point**

- ● It increases the confidence that the code is correct and working as expected
- ○ Testing helps add more logic into functions without lowering the quality of code
- ○ It allows faster development cycles when releasing to production

✓ **Correct**
Correct. Increasing the confidence in code is one of the benefits of testing.

10. What is a common excuse to avoid testing or not do testing as much? **1 / 1 point**

- ● That it takes too long to write tests when one should be concentrating in writing production code
- ○ That it makes production code run slower because it needs to import tests
- ○ That it makes code harder to read because it adds more lines to the overall codebase

✓ **Correct**

Correct. A common excuse is that it takes too much time to write tests, but in reality even if it makes development take longer, it produces better code, and easier ways to debug failures.