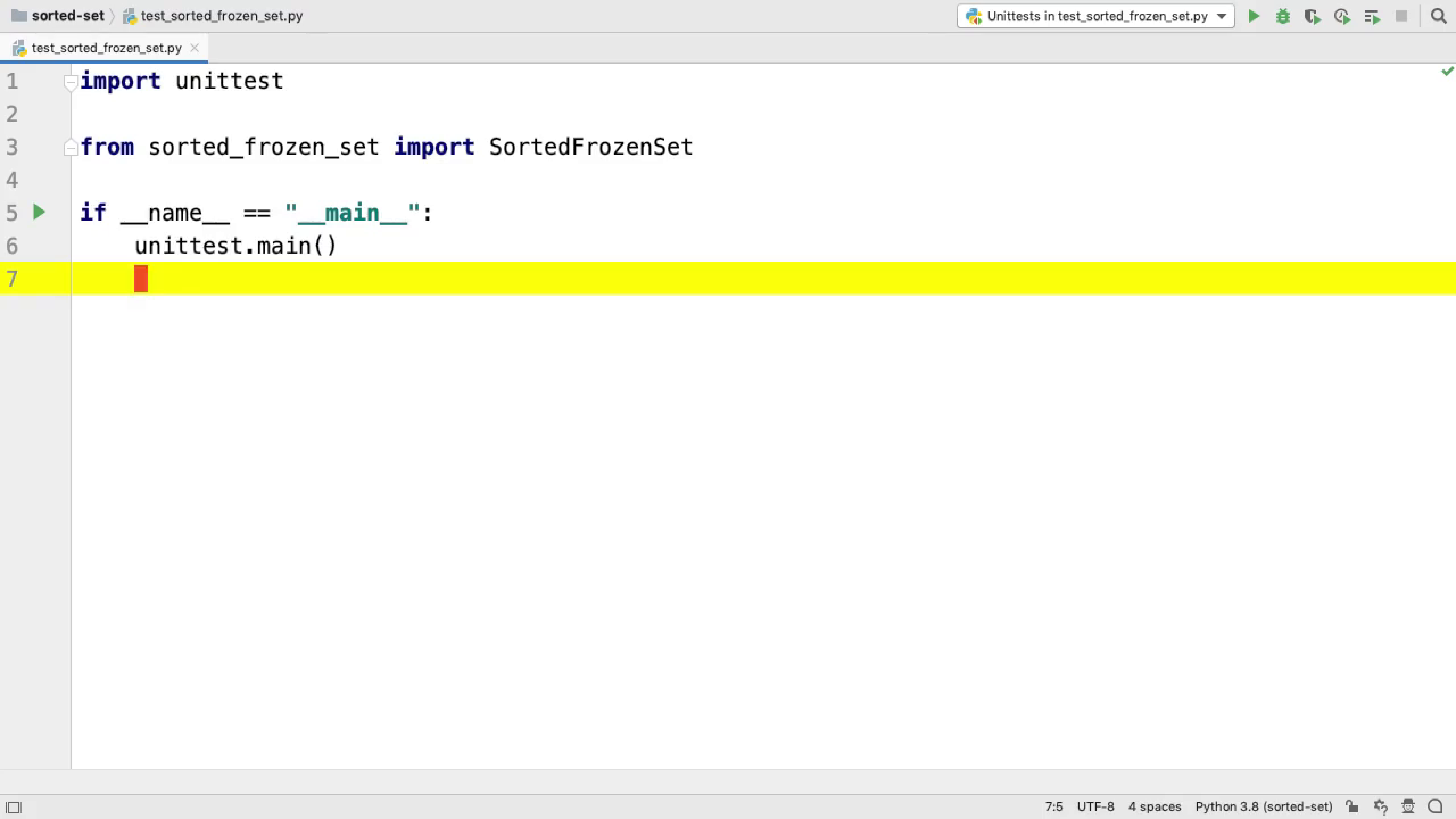
The Construction Convention

Robert Smallshire
COFOUNDER - SIXTY NORTH

@robsmallshire

Austin Bingham
COFOUNDER - SIXTY NORTH

@austin_bingham


sorted-set
test_sorted_frozen_set.py
Unittests in test_sorted_frozen_set.py
test_sorted_frozen_set.py
7:5 UTF-8 4 spaces Python 3.8 (sorted-set)

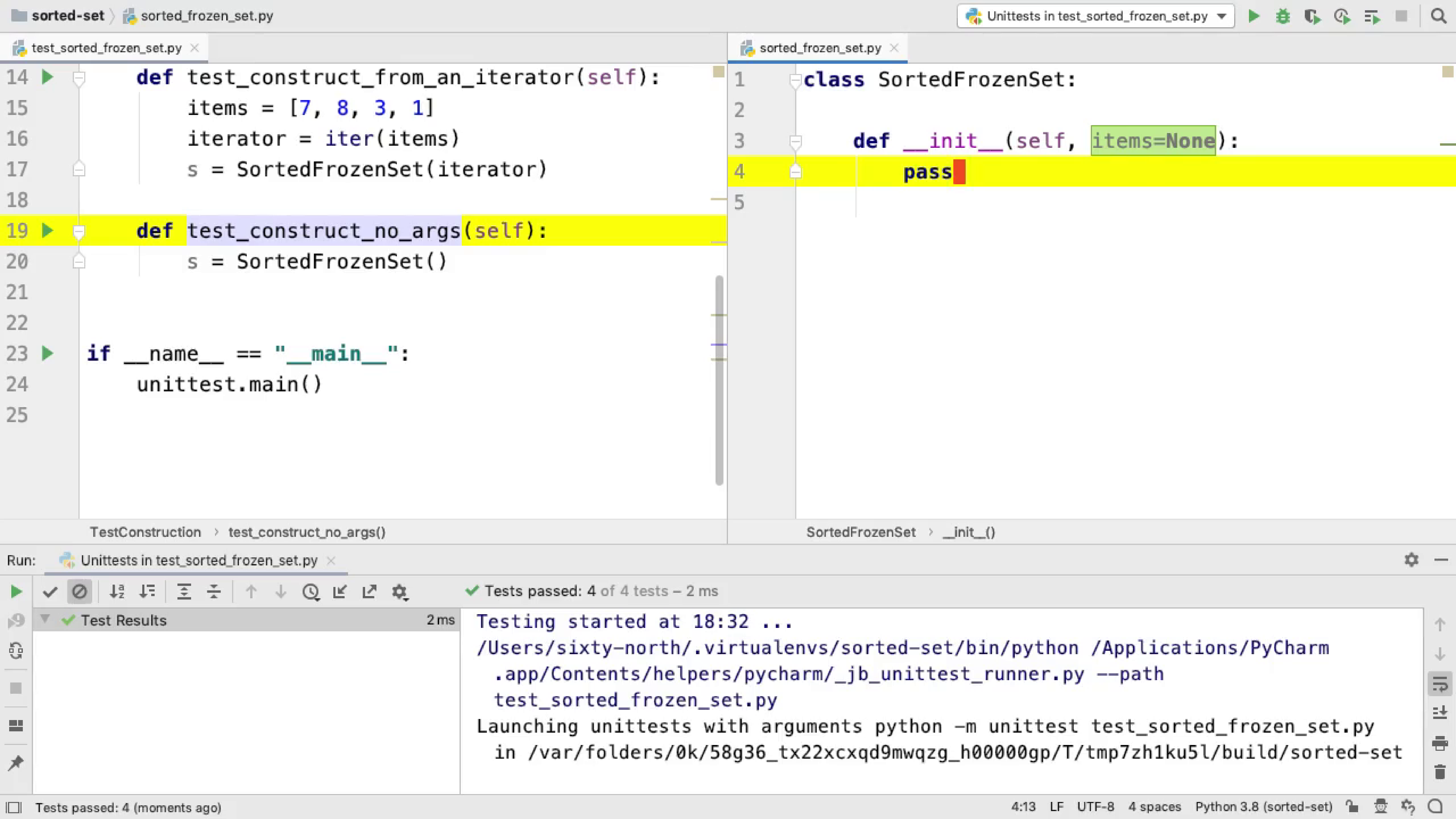
```
import unittest

from sorted_frozen_set import SortedFrozenSet

if __name__ == "__main__":
    unittest.main()
    
```

Executing Unit Tests

```
$ python test_sorted_frozen_set.py -v
Traceback (most recent call last):
  File "test_sorted_frozen_set.py", line 3, in <module>
    from sorted_frozen_set import SortedFrozenSet
ModuleNotFoundError: No module named 'sorted_frozen_set'
$
```



sorted-set › sorted_frozen_set.py

Unittests in test_sorted_frozen_set.py

test_sorted_frozen_set.py

```python
    def test_construct_from_an_iterator(self):
        items = [7, 8, 3, 1]
        iterator = iter(items)
        s = SortedFrozenSet(iterator)

    def test_construct_no_args(self):
        s = SortedFrozenSet()


if __name__ == "__main__":
    unittest.main()

```

TestConstruction › test_construct_no_args()

sorted_frozen_set.py

```python
class SortedFrozenSet:

    def __init__(self, items=None):
        pass

```

SortedFrozenSet › __init__()

Run: Unittests in test_sorted_frozen_set.py

Tests passed: 4 of 4 tests – 2 ms

Test Results 2 ms

```
Testing started at 18:32 ...
/Users/sixty-north/.virtualenvs/sorted-set/bin/python /Applications/PyCharm
 .app/Contents/helpers/pycharm/_jb_unittest_runner.py --path
 test_sorted_frozen_set.py
Launching unittests with arguments python -m unittest test_sorted_frozen_set.py
 in /var/folders/0k/58g36_tx22xcxqd9mwqzg_h00000gp/T/tmp7zh1ku5l/build/sorted-set
```

Tests passed: 4 (moments ago)   4:13 LF UTF-8 4 spaces Python 3.8 (sorted-set)

Table of Contents

`unittest` — Unit testing framework

Source code: Lib/unittest/__init__.py

(If you are already familiar with the basic concepts of testing, you might want to skip to the list of assert methods.)

The `unittest` unit testing framework was originally inspired by JUnit and has a similar flavor as major unit testing frameworks in other languages. It supports test automation, sharing of setup and shut down code for tests, aggregation of tests into collections, and independence of the tests from the reporting framework.

To achieve this, `unittest` supports some important concepts in an object-oriented way:

test fixture
: A *test fixture* represents the preparation needed to perform one or more tests, and any associated cleanup actions. This may involve, for example, creating temporary or proxy databases, directories, or starting a server process.

test case
: A *test case* is the individual unit of testing. It checks for a specific response to a particular set of inputs. `unittest` provides a base class, `TestCase`, which may be used to create new test cases.

test suite
: A *test suite* is a collection of test cases, test suites, or both. It is used to aggregate tests that should be executed together.

test runner
: A *test runner* is a component which orchestrates the execution of tests and provides the outcome