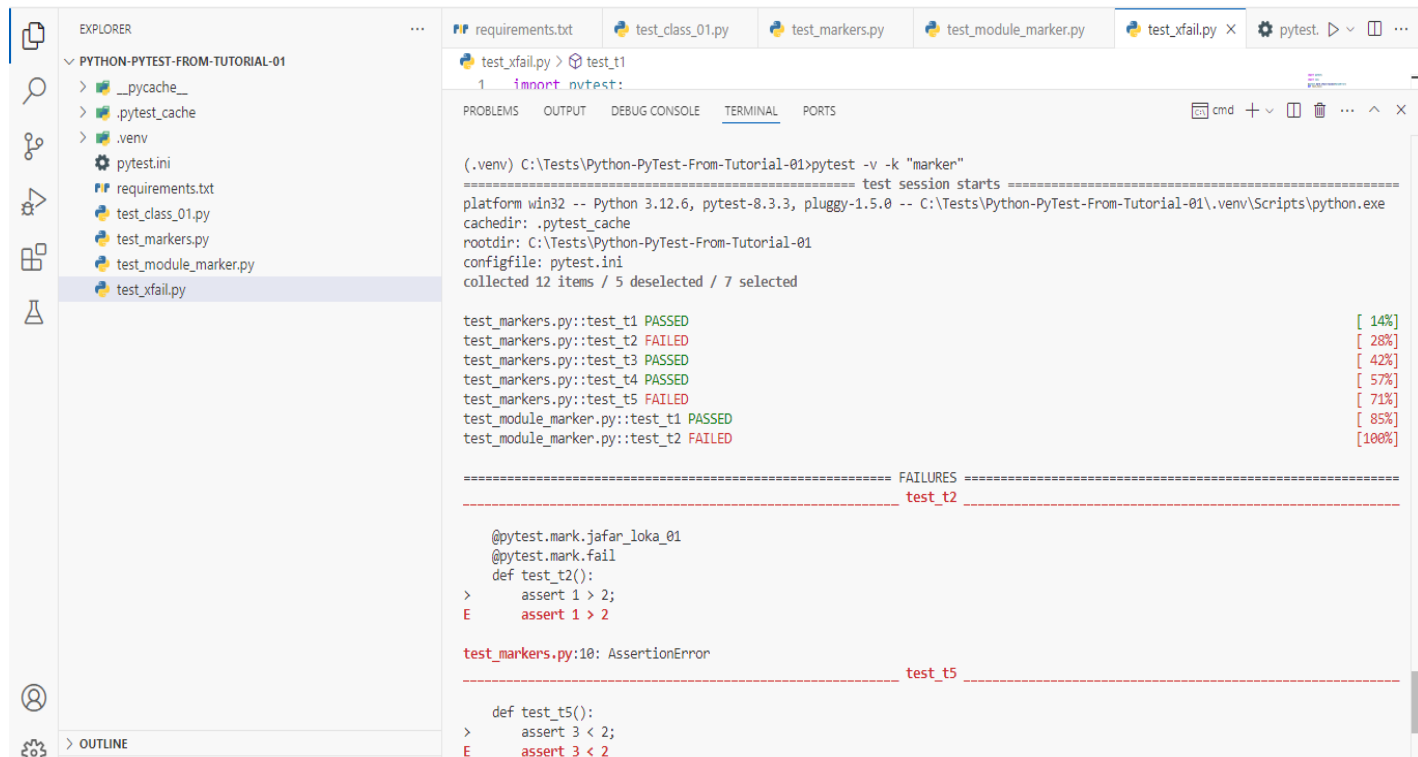


If We Want To Select The Test Cases Based On The File/s Name OR Test Function/s Name, We Can Use -k option:



The screenshot shows the VS Code interface with a file explorer on the left and a terminal window on the right. The file explorer shows a project named 'PYTHON-PYTEST-FROM-TUTORIAL-01' with files like 'requirements.txt', 'test\_class\_01.py', 'test\_markers.py', 'test\_module\_marker.py', and 'test\_xfail.py'. The terminal window shows the command 'pytest -v -k "marker"' being executed. The output displays a list of test cases with their status and progress percentage. The test cases are: test\_markers.py::test\_t1 PASSED [14%], test\_markers.py::test\_t2 FAILED [28%], test\_markers.py::test\_t3 PASSED [42%], test\_markers.py::test\_t4 PASSED [57%], test\_markers.py::test\_t5 FAILED [71%], test\_module\_marker.py::test\_t1 PASSED [85%], and test\_module\_marker.py::test\_t2 FAILED [100%]. The terminal also shows a 'FAILURES' section with details for test\_t2 and test\_t5.

```
test_xfail.py > test_t1
1  import pytest:

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "marker"
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 5 deselected / 7 selected

test_markers.py::test_t1 PASSED [ 14%]
test_markers.py::test_t2 FAILED [ 28%]
test_markers.py::test_t3 PASSED [ 42%]
test_markers.py::test_t4 PASSED [ 57%]
test_markers.py::test_t5 FAILED [ 71%]
test_module_marker.py::test_t1 PASSED [ 85%]
test_module_marker.py::test_t2 FAILED [100%]

===== FAILURES =====
test_t2
@pytest.mark.jafar_loka_01
@pytest.mark.fail
def test_t2():
> assert 1 > 2;
E      assert 1 > 2

test_markers.py:10: AssertionError
test_t5
def test_t5():
> assert 3 < 2;
E      assert 3 < 2
```

\*\*\*\*\*

If We Want To Delete The Error Trace From The Final Result, We Can Use --tb=no



The screenshot shows the VS Code interface with a file explorer on the left and a terminal window on the right. The file explorer shows a project named 'PYTHON-PYTEST-FROM-TUTORIAL-01' with files like 'requirements.txt', 'test\_class\_01.py', 'test\_markers.py', 'test\_module\_marker.py', and 'test\_xfail.py'. The terminal window shows the command 'pytest -v -k "marker" --tb=no' being executed. The output displays a list of test cases with their status and progress percentage. The test cases are: test\_markers.py::test\_t1 PASSED [14%], test\_markers.py::test\_t2 FAILED [28%], test\_markers.py::test\_t3 PASSED [42%], test\_markers.py::test\_t4 PASSED [57%], test\_markers.py::test\_t5 FAILED [71%], test\_module\_marker.py::test\_t1 PASSED [85%], and test\_module\_marker.py::test\_t2 FAILED [100%]. The terminal also shows a 'short test summary info' section with details for test\_t2 and test\_t5.

```
test_xfail.py > test_t1
1  import pytest:

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "marker" --tb=no
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 5 deselected / 7 selected

test_markers.py::test_t1 PASSED [ 14%]
test_markers.py::test_t2 FAILED [ 28%]
test_markers.py::test_t3 PASSED [ 42%]
test_markers.py::test_t4 PASSED [ 57%]
test_markers.py::test_t5 FAILED [ 71%]
test_module_marker.py::test_t1 PASSED [ 85%]
test_module_marker.py::test_t2 FAILED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
===== 3 failed, 4 passed, 5 deselected in 0.09s =====
```

\*\*\*\*\*

```
Python-PyTest-From-Tutorial-01

EXPLORER
PYTHON-PYTEST-FROM-TUTORIAL-01
  > __pycache__
  > .pytest_cache
  > .venv
    pytest.ini
    requirements.txt
    test_class_01.py
    test_markers.py
    test_module_marker.py
    test_xfail.py

test_xfail.py > test_t1
1 import pytest:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "t1" --tb=no
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 8 deselected / 4 selected

test_class_01.py::TestClass::test_t1 SKIPPED (No Reason Here) [ 25%]
test_markers.py::test_t1 PASSED [ 50%]
test_module_marker.py::test_t1 PASSED [ 75%]
test_xfail.py::test_t1 FAILED [100%]

===== short test summary info =====
FAILED test_xfail.py::test_t1 - TypeError: can only concatenate str (not "int") to str
===== 1 failed, 2 passed, 1 skipped, 8 deselected in 0.07s =====
```

\*\*\*\*\*

**Note:** We Can Use Condition Statements With -k:

```
Python-PyTest-From-Tutorial-01

EXPLORER
PYTHON-PYTEST-FROM-TUTORIAL-01
  > __pycache__
  > .pytest_cache
  > .venv
    pytest.ini
    requirements.txt
    test_class_01.py
    test_markers.py
    test_module_marker.py
    test_xfail.py

test_xfail.py > test_t1
1 import pytest:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "marker or t1" --tb=no
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 3 deselected / 9 selected

test_class_01.py::TestClass::test_t1 SKIPPED (No Reason Here) [ 11%]
test_markers.py::test_t1 PASSED [ 22%]
test_markers.py::test_t2 FAILED [ 33%]
test_markers.py::test_t3 PASSED [ 44%]
test_markers.py::test_t4 PASSED [ 55%]
test_markers.py::test_t5 FAILED [ 66%]
test_module_marker.py::test_t1 PASSED [ 77%]
test_module_marker.py::test_t2 FAILED [ 88%]
test_xfail.py::test_t1 FAILED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
FAILED test_xfail.py::test_t1 - TypeError: can only concatenate str (not "int") to str
===== 4 failed, 4 passed, 1 skipped, 3 deselected in 0.26s =====
```

```
Python-PyTest-From-Tutorial-01

EXPLORER
PYTHON-PYTEST-FROM-TUTORIAL-01
  __pycache__
  .pytest_cache
  .venv
  pytest.ini
  requirements.txt
  test_class_01.py
  test_markers.py
  test_module_marker.py
  test_xfail.py

requirements.txt test_class_01.py test_markers.py test_module_marker.py test_xfail.py

test_xfail.py > test_t1
1 import pytest:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "marker and not t1" --tb=no
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 7 deselected / 5 selected

test_markers.py::test_t2 FAILED [ 20%]
test_markers.py::test_t3 PASSED [ 40%]
test_markers.py::test_t4 PASSED [ 60%]
test_markers.py::test_t5 FAILED [ 80%]
test_module_marker.py::test_t2 FAILED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
===== 3 failed, 2 passed, 7 deselected in 0.08s =====
```

\*\*\*\*\*

If We Want To Run The Test Cases, And Stop When First Fail Occur We Can Use -x Option:

```
Python-PyTest-From-Tutorial-01

EXPLORER
PYTHON-PYTEST-FROM-TUTORIAL-01
  __pycache__
  .pytest_cache
  .venv
  pytest.ini
  requirements.txt
  test_class_01.py
  test_markers.py
  test_module_marker.py
  test_xfail.py

requirements.txt test_class_01.py test_markers.py test_module_marker.py test_xfail.py

test_xfail.py > test_t1
1 import pytest:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -k "marker and not t1" --tb=no -x
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items / 7 deselected / 5 selected

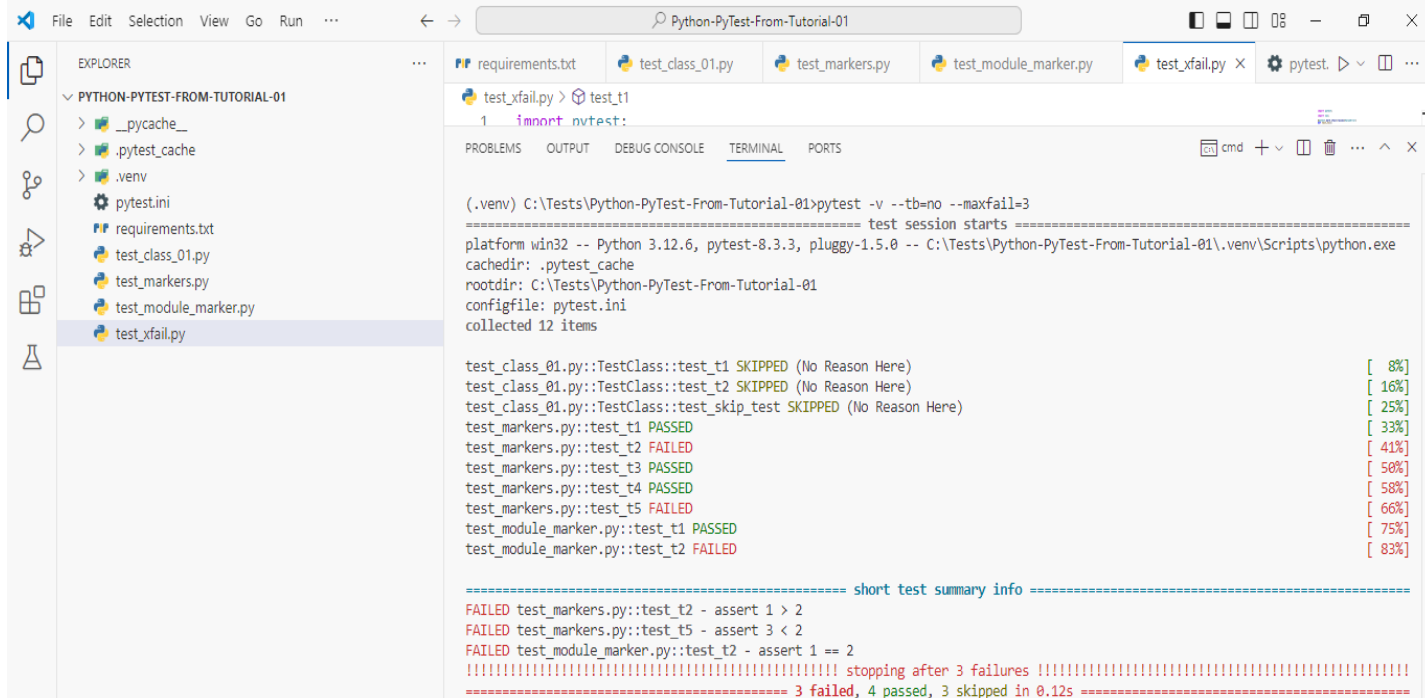
test_markers.py::test_t2 FAILED [ 20%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! stopping after 1 failures !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
===== 1 failed, 7 deselected in 0.06s =====
```

\*\*\*\*\*

Also, We Can Set The Maximum Of Failure Times Of Specific Tests, OR The Wall Tests.

Note: Here We Use --maxfail=3.



The screenshot shows the VS Code interface with a file explorer on the left and a terminal window on the right. The file explorer shows a project named 'PYTHON-PYTEST-FROM-TUTORIAL-01' with files like 'requirements.txt', 'test\_class\_01.py', 'test\_markers.py', 'test\_module\_marker.py', and 'test\_xfail.py'. The terminal window shows the command 'pytest -v --tb=no --maxfail=3' being executed. The output displays the test session starting, collecting 12 items, and running tests. The results show that 3 tests failed, 4 passed, and 3 were skipped. The failed tests are 'test\_markers.py::test\_t2', 'test\_markers.py::test\_t5', and 'test\_module\_marker.py::test\_t2'. The terminal output is as follows:

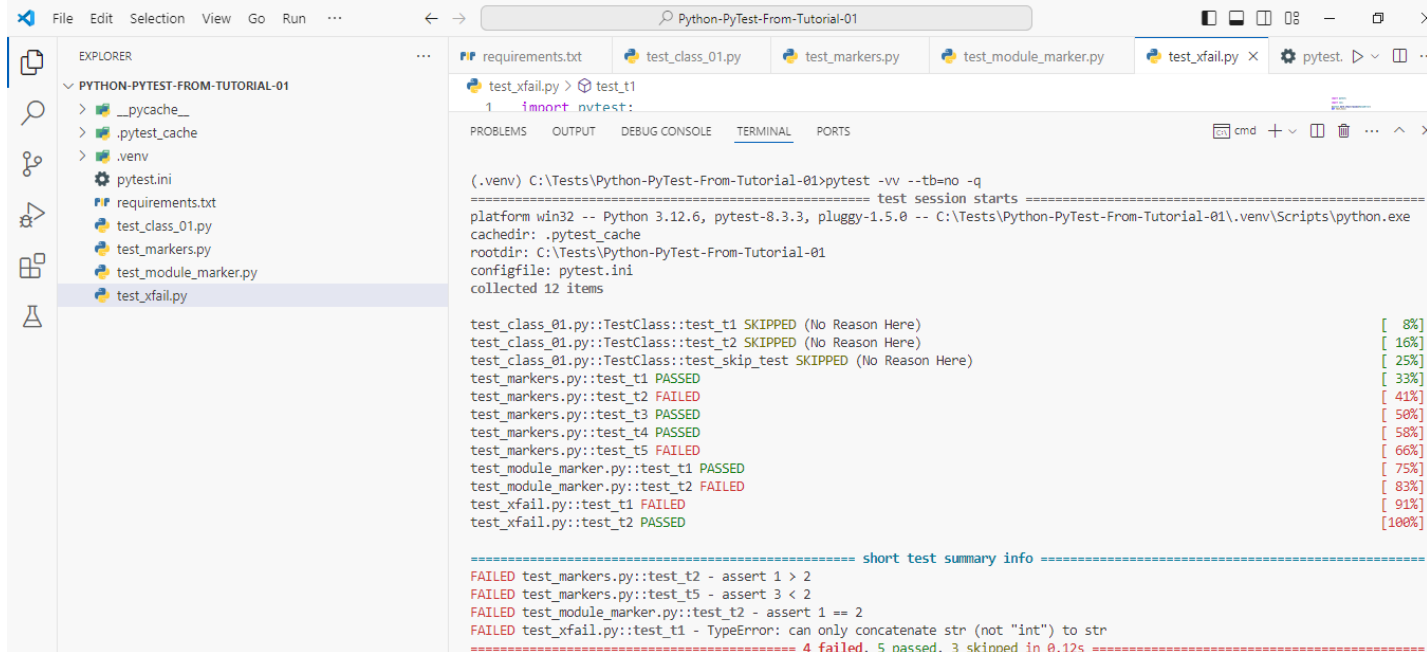
```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v --tb=no --maxfail=3
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items

test_class_01.py::TestClass::test_t1 SKIPPED (No Reason Here) [ 8%]
test_class_01.py::TestClass::test_t2 SKIPPED (No Reason Here) [ 16%]
test_class_01.py::TestClass::test_skip_test SKIPPED (No Reason Here) [ 25%]
test_markers.py::test_t1 PASSED [ 33%]
test_markers.py::test_t2 FAILED [ 41%]
test_markers.py::test_t3 PASSED [ 50%]
test_markers.py::test_t4 PASSED [ 58%]
test_markers.py::test_t5 FAILED [ 66%]
test_module_marker.py::test_t1 PASSED [ 75%]
test_module_marker.py::test_t2 FAILED [ 83%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
!!!!!!!!!!!!!!!!!!!!!! stopping after 3 failures !!!!!!!!!!!!!!!!!!!!!!!
===== 3 failed, 4 passed, 3 skipped in 0.12s =====
```

\*\*\*\*\*

When We Are Write The Actual Test Scenarios, We Can use -q.



```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -vv --tb=no -q
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items

test_class_01.py::TestClass::test_t1 SKIPPED (No Reason Here) [ 8%]
test_class_01.py::TestClass::test_t2 SKIPPED (No Reason Here) [ 16%]
test_class_01.py::TestClass::test_skip_test SKIPPED (No Reason Here) [ 25%]
test_markers.py::test_t1 PASSED [ 33%]
test_markers.py::test_t2 FAILED [ 41%]
test_markers.py::test_t3 PASSED [ 50%]
test_markers.py::test_t4 PASSED [ 58%]
test_markers.py::test_t5 FAILED [ 66%]
test_module_marker.py::test_t1 PASSED [ 75%]
test_module_marker.py::test_t2 FAILED [ 83%]
test_xfail.py::test_t1 FAILED [ 91%]
test_xfail.py::test_t2 PASSED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
FAILED test_xfail.py::test_t1 - TypeError: can only concatenate str (not "int") to str
===== 4 failed, 5 passed, 3 skipped in 0.12s =====
```

\*\*\*\*\*

If We Want To Run The Last Failed Test Cases From PyTest Cache, Then We Can use --lf

```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v --tb=no --lf
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 7 items / 3 deselected / 4 selected
run-last-failure: rerun previous 4 failures

test_markers.py::test_t2 FAILED [ 25%]
test_markers.py::test_t5 FAILED [ 50%]
test_module_marker.py::test_t2 FAILED [ 75%]
test_xfail.py::test_t1 FAILED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
FAILED test_xfail.py::test_t1 - TypeError: can only concatenate str (not "int") to str
===== 4 failed, 3 deselected in 0.07s =====
```

\*\*\*\*\*

If We Want To Run The Failed Tests First From Cache, And Then The Other Test Cases, We Can Use --ff

```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -vvv -q --tb=no --ff
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
collected 12 items
run-last-failure: rerun previous 4 failures first

test_markers.py::test_t2 FAILED [ 8%]
test_markers.py::test_t5 FAILED [ 16%]
test_module_marker.py::test_t2 FAILED [ 25%]
test_xfail.py::test_t1 FAILED [ 33%]
test_class_01.py::TestClass::test_t1 SKIPPED (No Reason Here) [ 41%]
test_class_01.py::TestClass::test_t2 SKIPPED (No Reason Here) [ 50%]
test_class_01.py::TestClass::test_skip_test SKIPPED (No Reason Here) [ 58%]
test_markers.py::test_t1 PASSED [ 66%]
test_markers.py::test_t3 PASSED [ 75%]
test_markers.py::test_t4 PASSED [ 83%]
test_module_marker.py::test_t1 PASSED [ 91%]
test_xfail.py::test_t2 PASSED [100%]

===== short test summary info =====
FAILED test_markers.py::test_t2 - assert 1 > 2
FAILED test_markers.py::test_t5 - assert 3 < 2
FAILED test_module_marker.py::test_t2 - assert 1 == 2
FAILED test_xfail.py::test_t1 - TypeError: can only concatenate str (not "int") to str
===== 4 failed, 5 passed, 3 skipped in 0.12s =====
```

\*\*\*\*\*

If We Want To Use Parameters With Our Test Cases:

**Note:** If We Don't Set [] For The Arguments (Name And Surname) It Will Be Consider As Error.

```
import pytest;

@pytest.mark.parametrize(['name', 'surname'], [['Jafar', 'Loka'], ['Test', '01']])
def test_parametrize(name, surname):
    print("The Name Is: ", name);
    print("The Surname Is: ", surname);

    assert name == 'Jafar';
    assert surname == 'Loka';

(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v -s --tb=no test_parametrize.py
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
metadata: {'Python': '3.12.6', 'Platform': 'Windows-10-10.0.19043-SP0', 'Packages': {'pytest': '8.3.3', 'pluggy': '1.5.0'}, 'Plugins': {'html': '4.1.1', 'metadata': '3.1.1'}}
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
plugins: html-4.1.1, metadata-3.1.1
collected 2 items

test_parametrize.py::test_parametrize[Jafar-Loka] The Name Is: Jafar
The Surname Is: Loka
PASSED
test_parametrize.py::test_parametrize[Test-01] The Name Is: Test
The Surname Is: 01
FAILED

===== short test summary info =====
FAILED test_parametrize.py::test_parametrize[Test-01] - AssertionError: assert 'Test' == 'Jafar'
===== 1 failed, 1 passed in 0.08s =====

*****
```

```
@pytest.mark.parametrize('number', [55, 45, 35, 75])
def test_numbers(number):
```

```
    assert number > 50;
```

```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v test_parametrize.py --tb=no
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
metadata: {'Python': '3.12.6', 'Platform': 'Windows-10-10.0.19043-SP0', 'Packages': {'pytest': '8.3.3', 'pluggy': '1.5.0'}, 'Plugins': {'html': '4.1.1', 'metadata': '3.1.1'}}
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
plugins: html-4.1.1, metadata-3.1.1
collected 6 items

test_parametrize.py::test_parametrize[Jafar-Loka] PASSED [ 16%]
test_parametrize.py::test_parametrize[Test-01] FAILED [ 33%]
test_parametrize.py::test_numbers[55] PASSED [ 50%]
test_parametrize.py::test_numbers[45] FAILED [ 66%]
test_parametrize.py::test_numbers[35] FAILED [ 83%]
test_parametrize.py::test_numbers[75] PASSED [100%]

===== short test summary info =====
FAILED test_parametrize.py::test_parametrize[Test-01] - AssertionError: assert 'Test' == 'Jafar'
FAILED test_parametrize.py::test_numbers[45] - assert 45 > 50
FAILED test_parametrize.py::test_numbers[35] - assert 35 > 50
===== 3 failed, 3 passed in 0.10s =====
```

\*\*\*\*\*

```
@pytest.mark.parametrize(["name1", "name2"], [("Jafar", "Loka"), ("Test", "01")])
def test_names(name1, name2):
    assert name1 + ' ' + name2 == "Jafar Loka"
```

```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v --tb=no test_parametrize.py
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
metadata: {'Python': '3.12.6', 'Platform': 'Windows-10-10.0.19043-SP0', 'Packages': {'pytest': '8.3.3', 'pluggy': '1.5.0'}, 'Plugins': {'html': '4.1.1', 'metadata': '3.1.1'}}
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
plugins: html-4.1.1, metadata-3.1.1
collected 8 items

test_parametrize.py::test_parametrize[Jafar-Loka] PASSED [ 12%]
test_parametrize.py::test_parametrize[Test-01] FAILED [ 25%]
test_parametrize.py::test_numbers[55] PASSED [ 37%]
test_parametrize.py::test_numbers[45] FAILED [ 50%]
test_parametrize.py::test_numbers[35] FAILED [ 62%]
test_parametrize.py::test_numbers[75] PASSED [ 75%]
test_parametrize.py::test_names[Jafar-Loka] PASSED [ 87%]
test_parametrize.py::test_names[Test-01] FAILED [100%]

===== short test summary info =====
FAILED test_parametrize.py::test_parametrize[Test-01] - AssertionError: assert 'Test' == 'Jafar'
FAILED test_parametrize.py::test_numbers[45] - assert 45 > 50
FAILED test_parametrize.py::test_numbers[35] - assert 35 > 50
FAILED test_parametrize.py::test_names[Test-01] - AssertionError: assert 'Test 01' == 'Jafar Loka'
===== 4 failed, 4 passed in 0.13s =====
```

\*\*\*\*\*



```
data=[([2, 3, 4], 'sum', 9), ([4, 5, 6], 'prod', 256)]
```

```
@pytest.mark.parametrize("a,b,c", data)
```

```
def test_data(a, b, c):
```

```
    if b == 'sum':
```

```
        assert sum(a) == c;
```

```
    elif b == 'prod':
```

```
        assert math.prod(a) == c;
```

```
(.venv) C:\Tests\Python-PyTest-From-Tutorial-01>pytest -v --tb=no test_parametrize.py::test_data
```

```
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Tests\Python-PyTest-From-Tutorial-01\.venv\Scripts\python.exe
cachedir: .pytest_cache
metadata: {'Python': '3.12.6', 'Platform': 'Windows-10-10.0.19043-SP0', 'Packages': {'pytest': '8.3.3', 'pluggy': '1.5.0'}, 'Plugins': {'html': '4.1.1', 'metadata': '3.1.1'}}
rootdir: C:\Tests\Python-PyTest-From-Tutorial-01
configfile: pytest.ini
plugins: html-4.1.1, metadata-3.1.1
collected 2 items
```

```
test_parametrize.py::test_data[a0-sum-9] PASSED
```

```
[ 50%]
```

```
test_parametrize.py::test_data[a1-prod-256] FAILED
```

```
[100%]
```

```
===== short test summary info =====
FAILED test_parametrize.py::test_data[a1-prod-256] - assert 120 == 256
===== 1 failed, 1 passed in 0.09s =====
```

```
*****
```