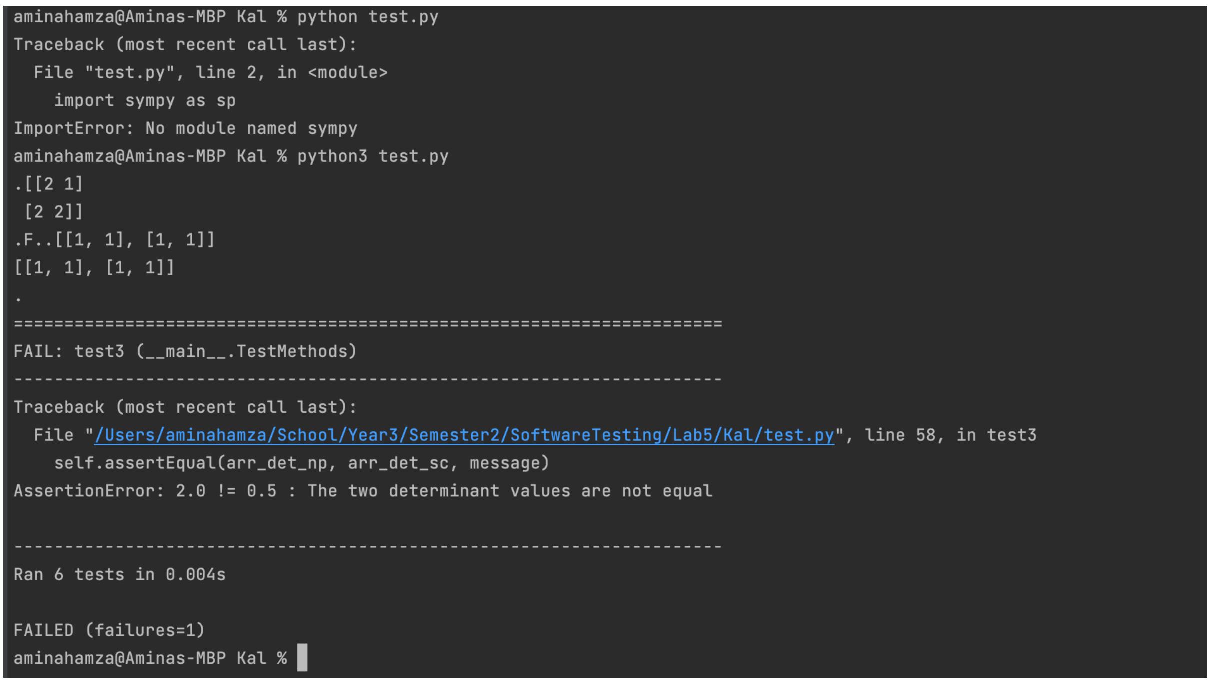
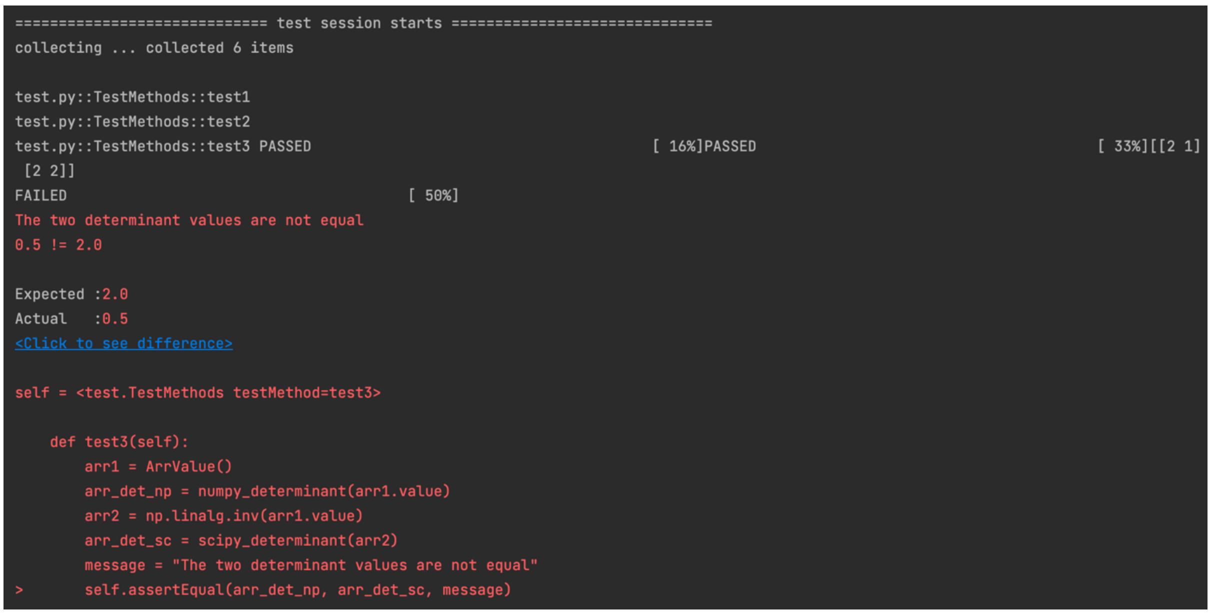
Lab5

Amina Hamza

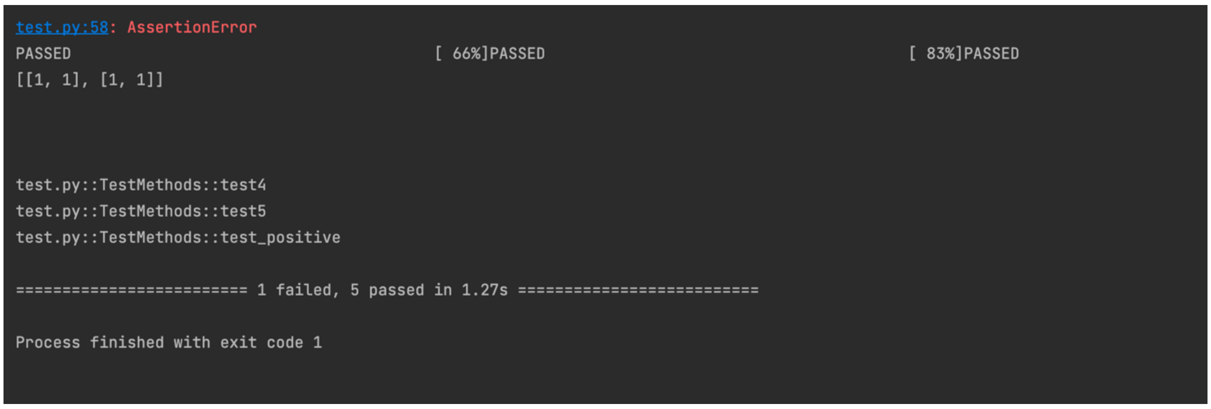


```
aminahamza@Aminas-MBP Kal % python test.py
Traceback (most recent call last):
  File "test.py", line 2, in <module>
    import sympy as sp
ImportError: No module named sympy
aminahamza@Aminas-MBP Kal % python3 test.py
.[[2 1]
 [2 2]]
.F..[[1, 1], [1, 1]]
[[1, 1], [1, 1]]
.
======================================================================
FAIL: test3 (__main__.TestMethods)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/aminahamza/School/Year3/Semester2/SoftwareTesting/Lab5/Kal/test.py", line 58, in test3
    self.assertEqual(arr_det_np, arr_det_sc, message)
AssertionError: 2.0 != 0.5 : The two determinant values are not equal

----------------------------------------------------------------------
Ran 6 tests in 0.004s

FAILED (failures=1)
aminahamza@Aminas-MBP Kal % 
```

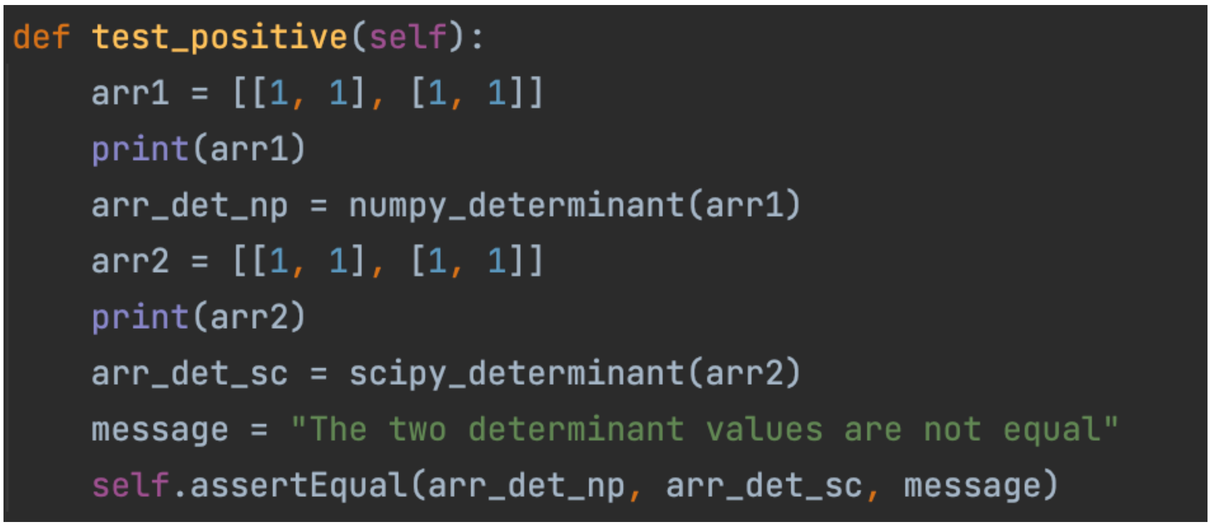
Display after running the test



```
============================= test session starts ==============================
collecting ... collected 6 items

test.py::TestMethods::test1
test.py::TestMethods::test2
test.py::TestMethods::test3 PASSED                                       [ 16%]PASSED                                   [ 33%][[2 1]
 [2 2]]
FAILED                                            [ 50%]
The two determinant values are not equal
0.5 != 2.0

Expected :2.0
Actual   :0.5
<Click to see difference>

self = <test.TestMethods testMethod=test3>

    def test3(self):
        arr1 = ArrValue()
        arr_det_np = numpy_determinant(arr1.value)
        arr2 = np.linalg.inv(arr1.value)
        arr_det_sc = scipy_determinant(arr2)
        message = "The two determinant values are not equal"
>       self.assertEqual(arr_det_np, arr_det_sc, message)
```

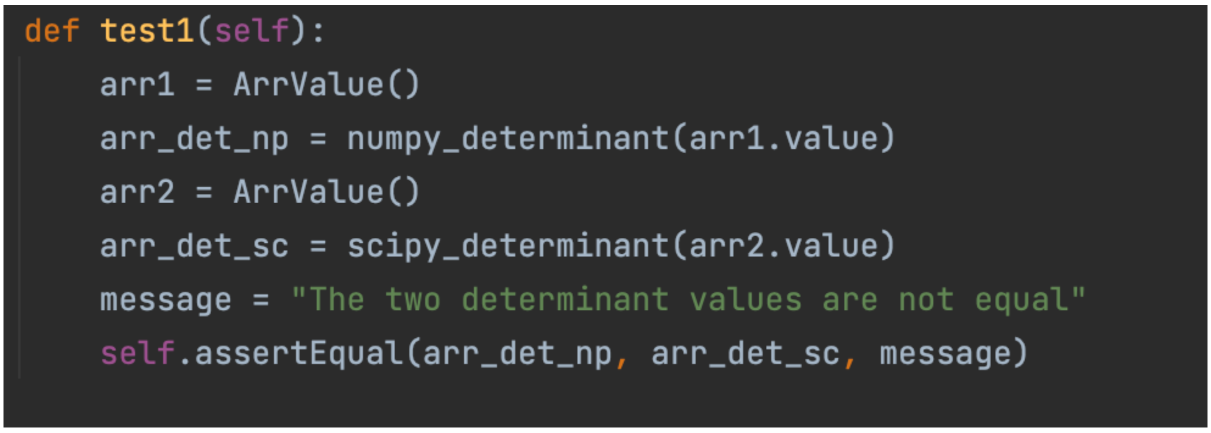


```
test.py:58: AssertionError
PASSED                                        [ 66%]PASSED                                                [ 83%]PASSED
[[1, 1], [1, 1]]



test.py::TestMethods::test4
test.py::TestMethods::test5
test.py::TestMethods::test_positive

========================= 1 failed, 5 passed in 1.27s ==========================

Process finished with exit code 1
```



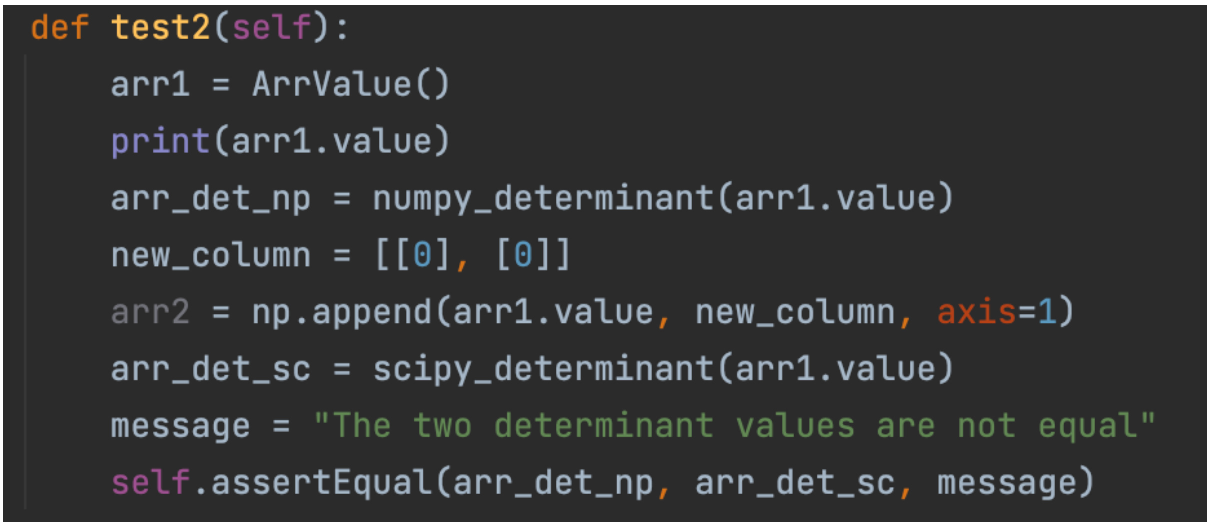
```python
def test_positive(self):
    arr1 = [[1, 1], [1, 1]]
    print(arr1)
    arr_det_np = numpy_determinant(arr1)
    arr2 = [[1, 1], [1, 1]]
    print(arr2)
    arr_det_sc = scipy_determinant(arr2)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test_positive(self):   # this test is passing because both arr1 and arr2 are same (arr1=arr2= [[1,1],[1,1]]) and both numpy determinant and scipy determinant give same answer.



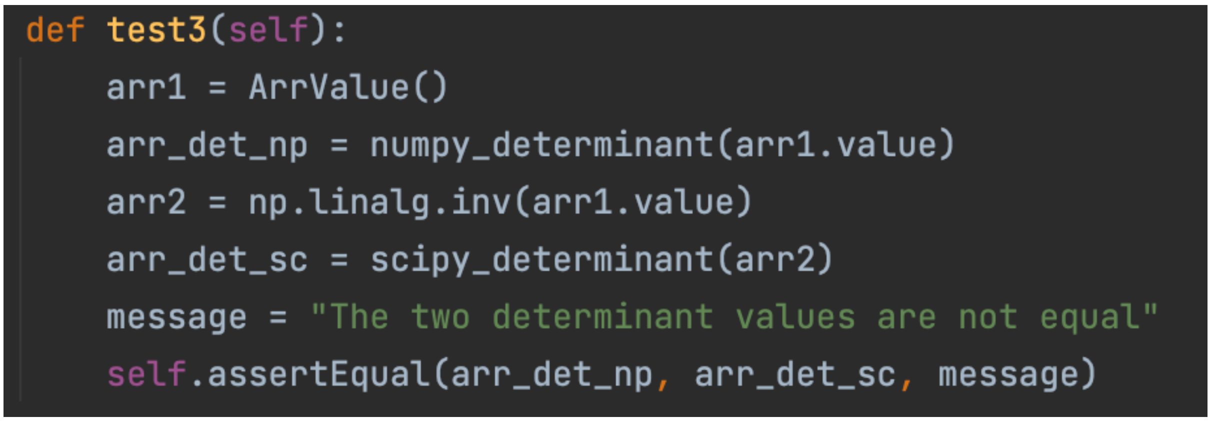
```python
def test1(self):
    arr1 = ArrValue()
    arr_det_np = numpy_determinant(arr1.value)
    arr2 = ArrValue()
    arr_det_sc = scipy_determinant(arr2.value)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test1(self):   # this test is passing because both arr1 and arr2 are same and both numpy determinant and scipy determinant give same answer



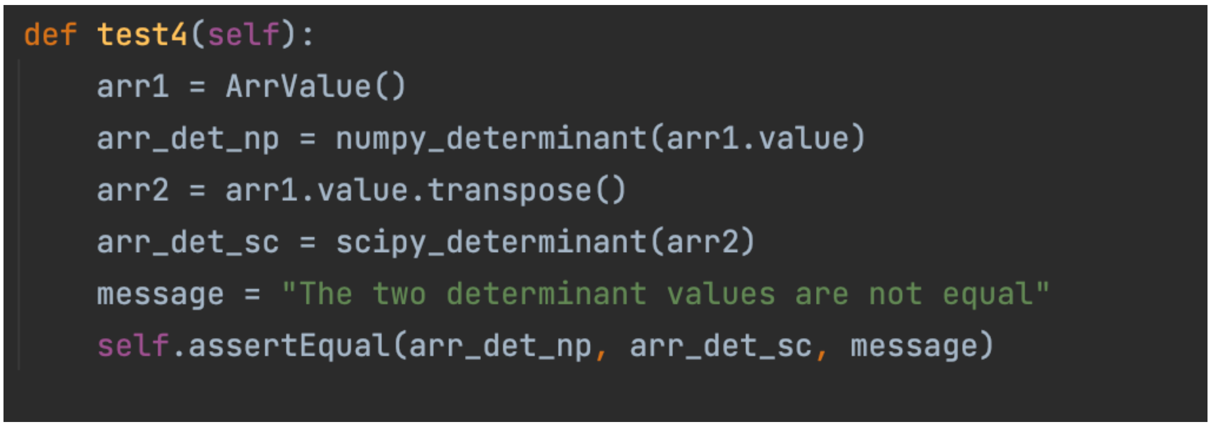
```python
def test2(self):
    arr1 = ArrValue()
    print(arr1.value)
    arr_det_np = numpy_determinant(arr1.value)
    new_column = [[0], [0]]
    arr2 = np.append(arr1.value, new_column, axis=1)
    arr_det_sc = scipy_determinant(arr1.value)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test2(self):   # arr1 is ArrValue and arr2 is getting by appending new column [[0], [0]], and then we get same output from both numpy and scipy determinant



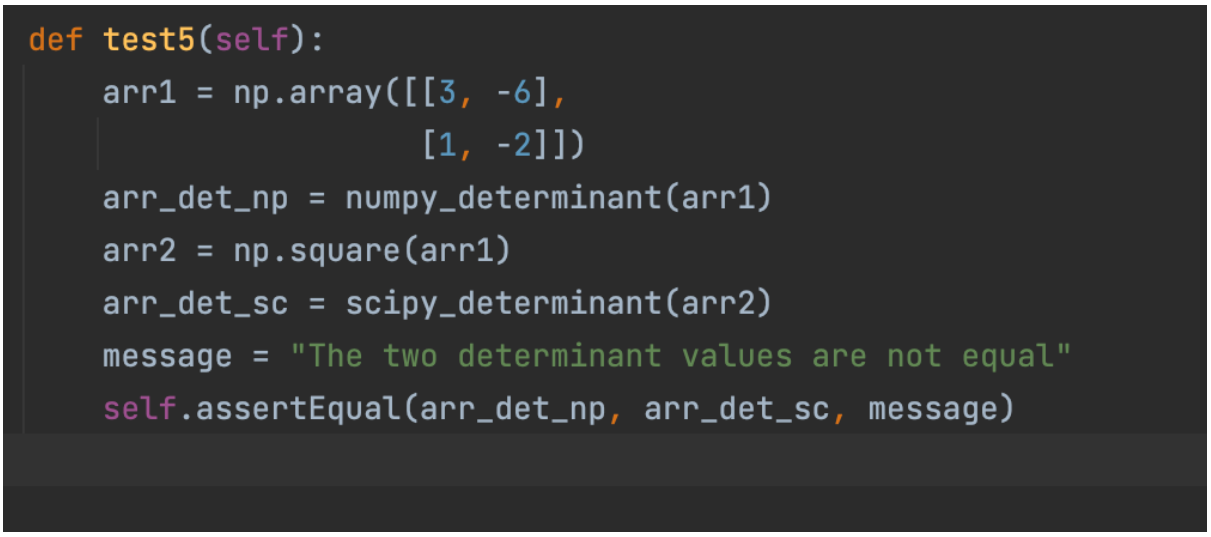
```python
def test3(self):
    arr1 = ArrValue()
    arr_det_np = numpy_determinant(arr1.value)
    arr2 = np.linalg.inv(arr1.value)
    arr_det_sc = scipy_determinant(arr2)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test3(self):   # arr2 is inverse of arr1 so we get different determinant



```python
def test4(self):
    arr1 = ArrValue()
    arr_det_np = numpy_determinant(arr1.value)
    arr2 = arr1.value.transpose()
    arr_det_sc = scipy_determinant(arr2)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test4(self): # arr2 is transpose of arr1, so when we transpose the array we still get same determinant.



```python
def test5(self):
    arr1 = np.array([[3, -6],
                     [1, -2]])
    arr_det_np = numpy_determinant(arr1)
    arr2 = np.square(arr1)
    arr_det_sc = scipy_determinant(arr2)
    message = "The two determinant values are not equal"
    self.assertEqual(arr_det_np, arr_det_sc, message)
```

def test5(self): # an idempotent matrix is a matrix which, when multiplied by itself, yields itself. arr2 is idempotent and then square of idempotent is also idempotent and hence we get same idempotent.