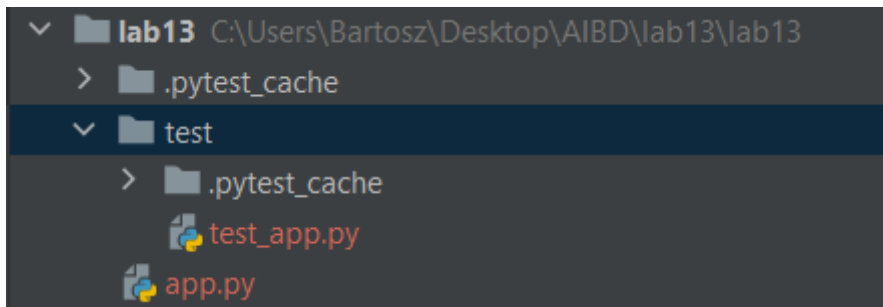


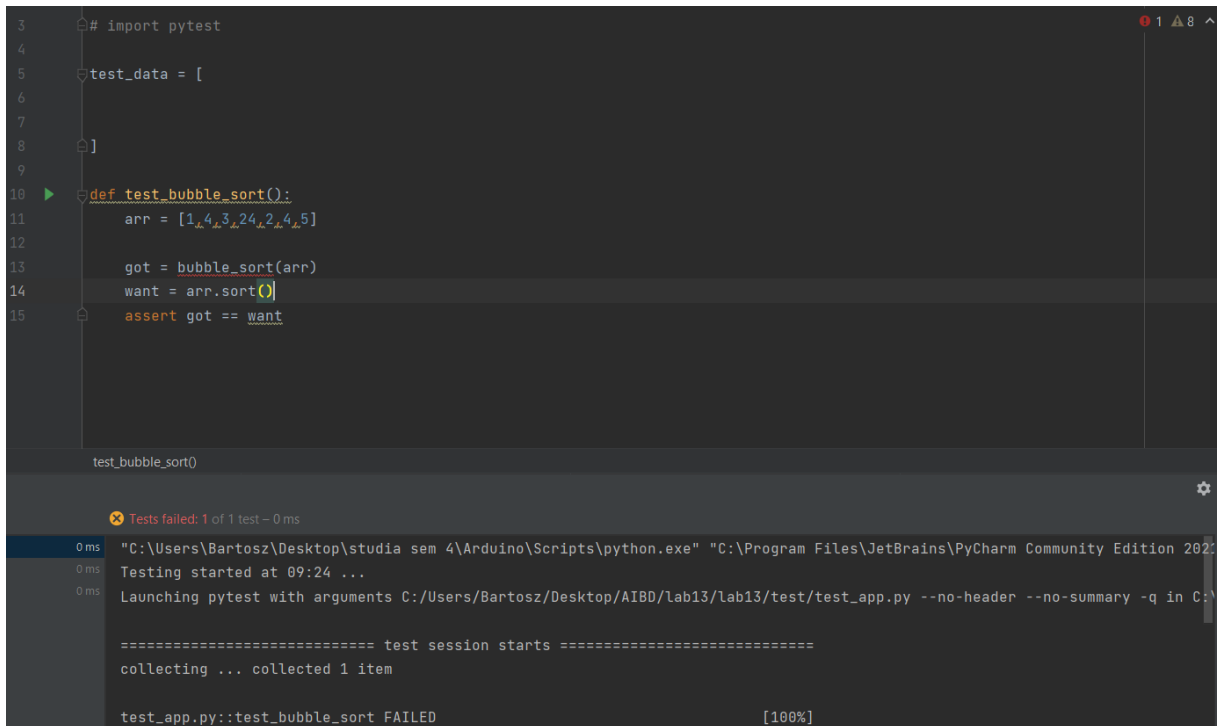
Bartosz Stokłosa 405766

PROJECT TREE:



RED:

(Test który z założenia ma nie rpechodzić)



GREEN:

(Uzupełnienie funkcjonalności)

```
1  from app import bubble_sort
2
3
4  # import pytest
5
6  test_data = [
7
8
9  ]
10
11 def test_bubble_sort():
12     arr = [1,4,3,2,4,5]
13
14     got = bubble_sort(arr)
15     want = arr.sort()
16     assert got == want
```

✓ Tests passed: 1 of 1 test – 0 ms

0 ms ===== test session starts =====
collecting ... collected 1 item

test_app.py::test_bubble_sort PASSED [100%]

===== 1 passed in 0.01s =====

Process finished with exit code 0

REAFCTOR:

Urozmaicenie testów

```
1 from app import bubble_sort
2
3
4 import pytest
5
6 test_data = [
7     ([1, 4, 3, 5, 136, 621, 32, 13, 2], ([1, 4, 3, 5, 136, 621, 32, 13, 2]).sort()),
8     ([1, 4, 3, 24, 2, 4, 5], ([1, 4, 3, 24, 2, 4, 5]).sort()),
9     ([-1, -4, 3, -5, -5, 2, -4, -5, 2, -4], ([-1, -4, 3, -5, -5, 2, -4, -5, 2, -4]).sort()),
10    ([1, 1, 1, 1, 1, 1, 1, 1], ([1, 1, 1, 1, 1, 1, 1, 1]).sort())
11]
12 @pytest.mark.parametrize('sample, expected_output', test_data)
13 def test_bubble_sort(sample, expected_output):
14     assert bubble_sort(sample) == expected_output
```

✓ Tests passed: 4 of 4 tests – 0 ms

0 ms	test_app.py::test_bubble_sort[sample0-None]	PASSED	[25%]
	test_app.py::test_bubble_sort[sample1-None]	PASSED	[50%]
	test_app.py::test_bubble_sort[sample2-None]	PASSED	[75%]
	test_app.py::test_bubble_sort[sample3-None]	PASSED	[100%]

===== 4 passed in 0.03s =====

Process finished with exit code 0