

Red test – widzimy, że test zawiera błąd skoro w pliku app.py nie zostało zdefiniowane „hello”

```
Lab_13_proba_2\myapp\test\test_app.py F [100%]  
===== FAILURES =====  
test_hello  
  
def test_hello():  
> got = hello("Aleksandra")  
E NameError: name 'hello' is not defined  
  
Lab_13_proba_2\myapp\test\test_app.py:2: NameError  
  
===== short test summary info =====  
FAILED Lab_13_proba_2\myapp\test\test_app.py::test_hello - NameError: name 'hello' is not defined  
===== 1 failed in 0.19s =====  
(studyssession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project> |]
```

Również załączam screen kiedy mamy kilka testów zdefiniowanych i nie zdefiniowaliśmy ich funkcjonalności w pliku app.py, więc mamy błędnie przeprowadzone testy

The screenshot shows the Visual Studio Code interface. On the left, the 'TESTING' sidebar shows a tree view with 'pandas_project' expanded, then 'Lab_13_proba_2', then 'myapp', and finally 'test'. Under 'test', 'test_app.py' is selected, showing a list of tests: 'test_hello', 'test_extract_sentiment', 'test_test_contain_word', and 'There is a duck in this test-duck-its'. The 'test_hello' test is marked as failed. The main editor shows the 'test_app.py' file with the following code:

```
1 from textblob import TextBlob  
2  
3  
  
def hello(name):  
    output = f'Hello {name}'  
    return output  
  
def extract_sentiment(text):  
    text = TextBlob(text)  
  
    return text.sentiment.polarity  
  
def text_contain_word(word: str, text: str):  
    return word in text
```

The terminal window at the bottom shows the following error message:

```
ERROR Lab_13_proba_2\myapp\test\test_app.py  
ImportError: cannot import name 'hello' from 'app' (D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project\Lab_13_proba_2\myapp\app.py)  
----- short test summary info -----  
Interrupted: 1 error during collection  
===== 1 error in 1.47s =====  
(studyssession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project> & "d:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project\studyssession\scripts\python.exe" "d:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project\Lab_13_proba_2\myapp\test\test_app.py"  
Traceback (most recent call last):  
  File "d:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project\Lab_13_proba_2\myapp\test\test_app.py", line 2, in <module>  
    from app import extract_sentiment  
ImportError: cannot import name 'extract_sentiment' from 'app' (D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project\studyssession\lib\site-packages\app\__init__.py)  
(studyssession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project> |]
```

Green Test – teraz zgodnie z metodą TDD zdefiniowaliśmy funkcjonalność testów z pliku test_app.py w pliku app.py i mamy dobrze przeprowadzone testy, jak widać korzystaliśmy z testów unittest, ale na pytest, to działa tak samo.

The screenshot shows the Visual Studio Code interface. On the left, the 'TESTING' sidebar shows a tree view with 'pandas_project' expanded, then 'Lab_13_proba_2', then 'myapp', and finally 'test'. Under 'test', 'test_app.py' is selected, showing a list of tests: 'test_hello', 'test_extract_sentiment', 'test_test_contain_word', and 'There is a duck in this test-duck-its'. The 'test_hello' test is marked as passed. The main editor shows the 'test_app.py' file with the following code:

```
1 from textblob import TextBlob  
2  
3  
  
def hello(name):  
    output = f'Hello {name}'  
    return output  
  
def extract_sentiment(text):  
    text = TextBlob(text)  
  
    return text.sentiment.polarity  
  
def text_contain_word(word: str, text: str):  
    return word in text
```

The terminal window at the bottom shows the following success message:

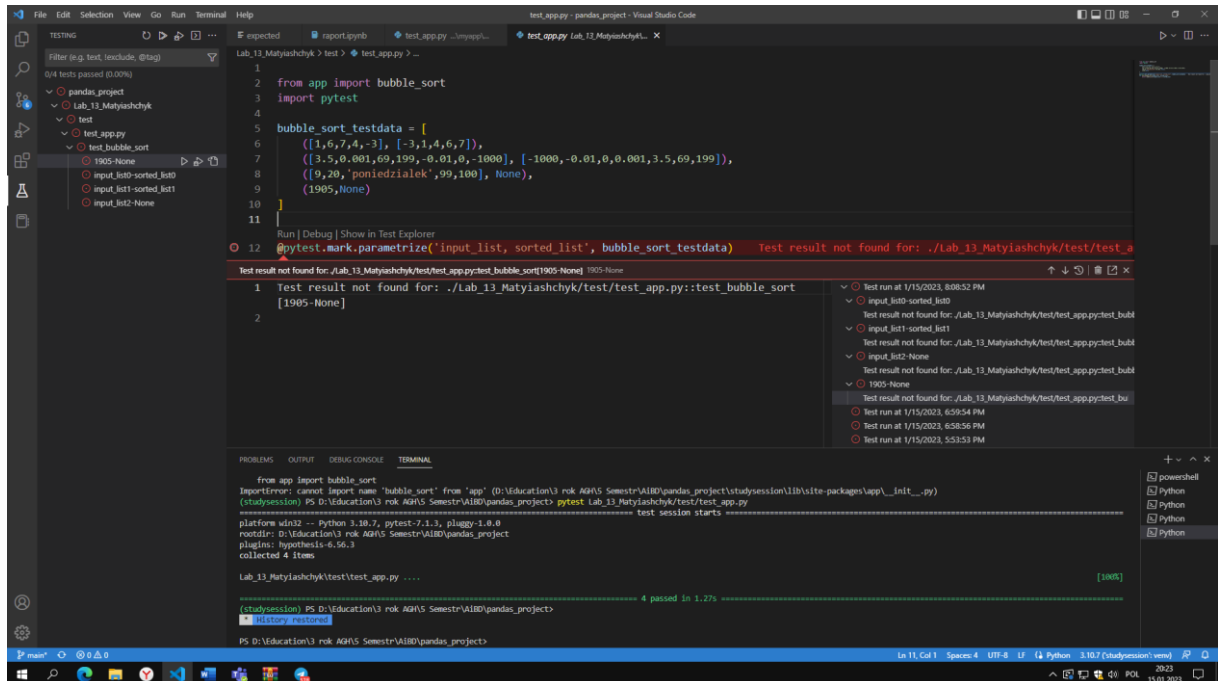
```
root@D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project  
plugins: hypothesis-6.56.3  
collected 4 items  
  
Lab_13_proba_2\myapp\test\test_app.py .....  
===== 4 passed in 1.18s =====  
(studyssession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project> |]
```

Faza **Refaktor** – tu ze względu na nie dużą skomplikowalność kodu nie musimy cokolwiek poprawiać albo polepszać

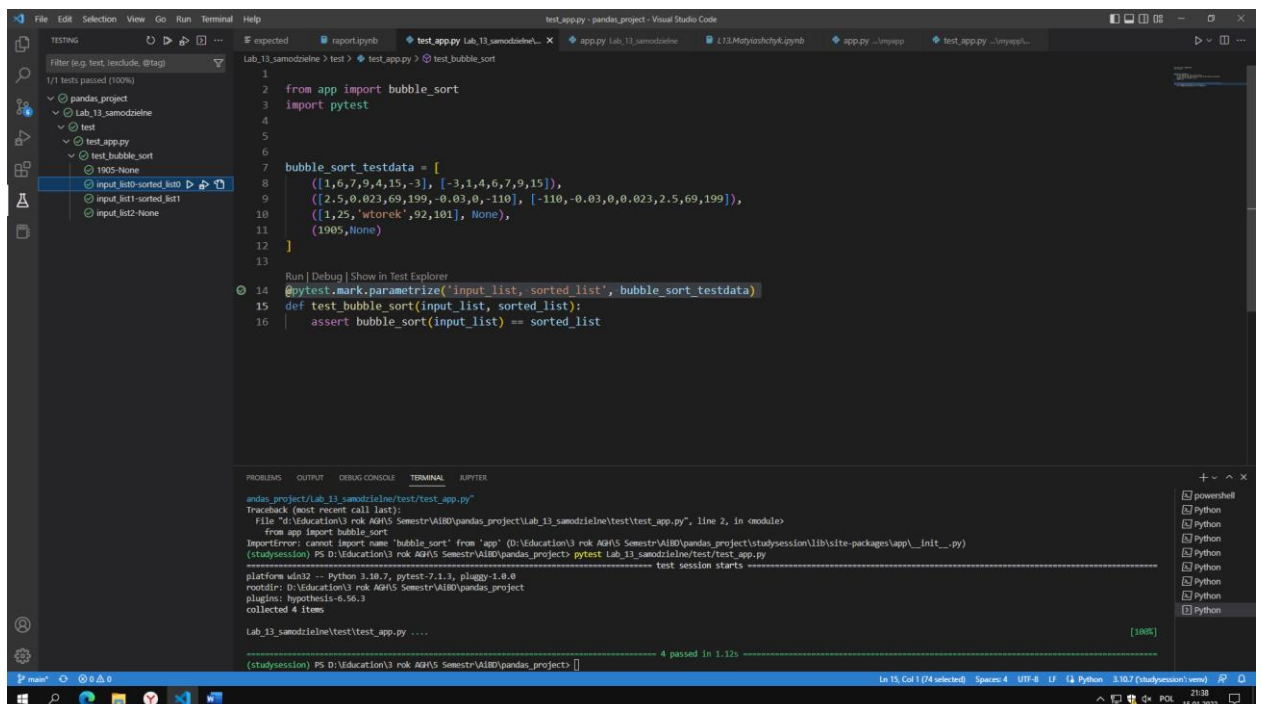
ZADANIE DO SAMOEDZIELNEGO WYKONANIA

Mieliśmy za zadanie napisać kod realizujący sortowanie bąbelkowe polegając się na metodzie TDD

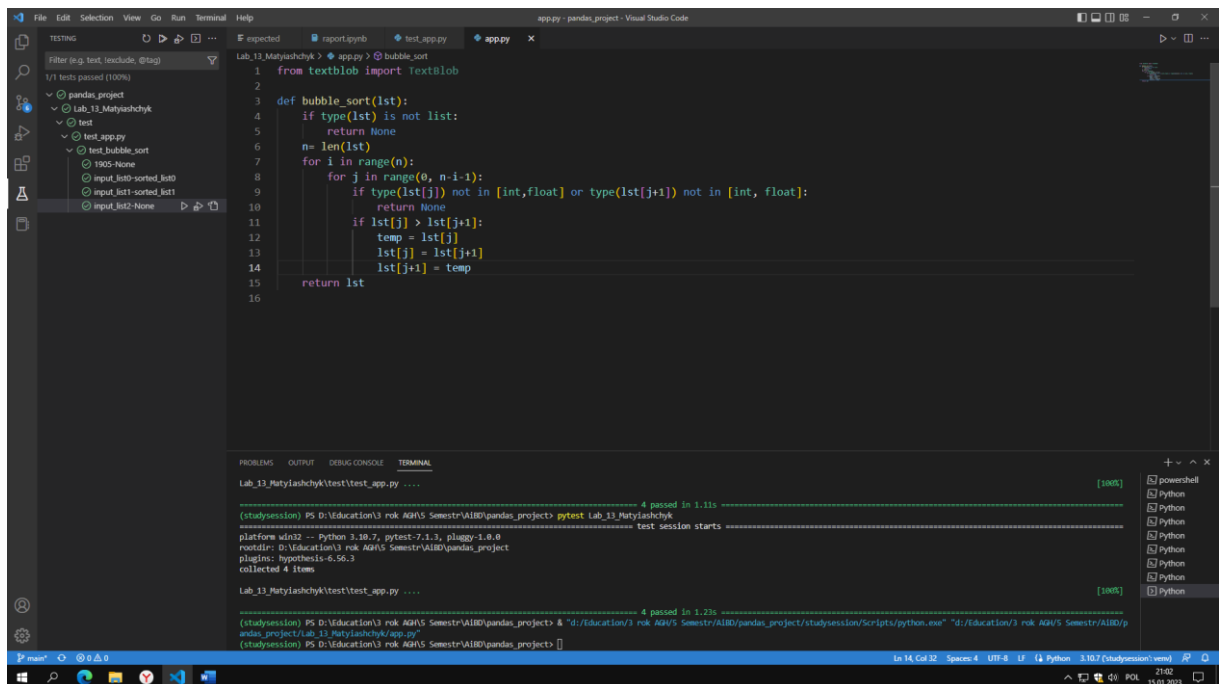
Red test – zaczynamy od tego, że definiujemy listę do sortowania i sam test



Green Test – po zdefiniowaniu funkcji które sprawdzają typy wpisanych obiektów oraz realizujące logiczne sortowanie wartości(wszystko to robimy w pliku app.py), to widzimy, że testy nasze działają i na screen'ie z ekranu widać że jest przetestowane przez pytest w konsoli jak również przez unittest z lewa



Faza **Refaktor** – tu po prostu skróciliśmy nasz kod usuwając zmienną „temp” i przesuając kod do jednego wierszu.



The screenshot shows the Visual Studio Code interface with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The code editor displays a Python file named `bubble_sort.py` with the following content:

```
1 from textblob import TextBlob
2
3 def bubble_sort(lst):
4     if type(lst) is not list:
5         return None
6     n = len(lst)
7     for i in range(n):
8         for j in range(0, n-i-1):
9             if type(lst[j]) not in [int, float] or type(lst[j+1]) not in [int, float]:
10                return None
11             if lst[j] > lst[j+1]:
12                 temp = lst[j]
13                 lst[j] = lst[j+1]
14                 lst[j+1] = temp
15     return lst
16
```

The terminal at the bottom shows the output of a pytest command, indicating that all tests passed:

```
Lab_13_Matyiaschuk\test\test_app.py .... [PASS]
(studysession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project> pytest Lab_13_Matyiaschuk
platform win32 -- Python 3.10.7, pytest-7.1.3, pluggy-1.0.0
rootdir: D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project
plugins: hypothesis-6.56.3
collected 4 items

Lab_13_Matyiaschuk\test\test_app.py .... [PASS]
(studysession) PS D:\Education\3 rok AGH\5 Semestr\AI\BD\pandas_project>
```