Piotr Artym, 122212

Link do githuba: https://github.com/kingslayer335/python-intro/tree/main/zadanie_2

Użycie narzędzia Coverage do przetestowania testów z zadania 2.

Wynik polecenia : coverage run -m unittest test_app.py

```
D:\coding\github\python-intro\zadanie_2>coverage run -m unittest test_app.py
..........................
----------------------------------------------------------------------
Ran 26 tests in 0.065s

OK
```

Wynik polecenia: coverage report

```
D:\coding\github\python-intro\zadanie_2>coverage report
Name          Stmts   Miss  Cover
---------------------------------
app.py           18      0   100%
test_app.py      70      0   100%
---------------------------------
TOTAL            88      0   100%
```

Wynik polecenia: coverage html

(tworzy folder ‘htmlcov’ z plikami między innymi test_app.py)

Coverage for **test_app.py**: 100%

70 statements | 70 run | 0 missing | 0 excluded

« prev ^ index » next coverage.py v7.7.0, created at 2025-03-21 14:06 +0100

```python
import unittest
from datetime import datetime
from app import is_email_on_the_list, square_of_number, sort_numbers, convert_date, is_palindrome

class TestIsEmailOnTheList(unittest.TestCase):
    # testy sprawdzajace czy email znajduje lub nie znajduje sie na roznych listach:
    def setUp(self):
        self.email_list = ['123@gmail.com', '456@gmail.com', '789@gmail.com']
        self.large_list = [f'{i}@gmail.com' for i in range(100000)]
        self.large_list.append('u456@gmail.com')
    # zwykla lista
    def test_email_in_list(self):
        self.assertTrue(is_email_on_the_list(self.email_list, '123@gmail.com'))

    def test_email_not_in_list(self):
        self.assertFalse(is_email_on_the_list(self.email_list, 'not_in_list@gmail.com'))
    # lista z duza iloscia elementow
    def test_email_in_large_list(self):
        self.assertTrue(is_email_on_the_list(self.large_list, 'u456@gmail.com'))

    def test_email_not_in_large_list(self):
        self.assertFalse(is_email_on_the_list(self.large_list, 'u56@gmail.com'))
    # pusta lista
    def test_email_in_empty_list(self):
        self.assertFalse(is_email_on_the_list([], '123@gmail.com'))


class TestSquareOfNumber(unittest.TestCase):
    # testy sprawdzajace czy funkcja dziala poprawnie dla roznych liczb:
    # dodatnie
    def test_square_of_positive(self):
        self.assertEqual(square_of_number(5), 25)
    # ujemne
    def test_square_of_negative(self):
        self.assertEqual(square_of_number(-3), 9)
    # zero
    def test_square_of_zero(self):
        self.assertEqual(square_of_number(0), 0)
    # floaty
    def test_square_of_float(self):
```