```
python -m venv testy_venv
testy_venv\Scripts\Activate.ps1
ały do zajec> python -m pip install pytest
```

- Tworzymy 3 pliki (funkcje.py, test_with_unittest.py, test_with_pytest.py)

funkcje.py

```python
def add(a,b):
    return a+b
```

test_with_unittest.py

```python
import funkcje
import unittest

# - Klasa powinna dziedziczyć po unittest.TestCase.
# - Metody testowe muszą mieć nazwę zaczynającą się od 'test' (np. test_add),
# inaczej nie zostaną uruchomione przy automatycznym wykrywaniu.

class Test_add(unittest.TestCase):
    def test_add(self):
        self.assertEqual(funkcje.add(2,4), 6)
        self.assertNotEqual(funkcje.add(2,4), 5)

# uruchom tylko jeśli bezposrednio startuje ten plik
if __name__ == '__main__':
    # uruchomienie runnera testów (wyszukuje testy i uruchamia)
    unittest.main()

# URUCHAMIANY TESTY
# - następnie zmieniamy returna w funckji na odejmowanie
#   i patrzymy jaki będzie wynik testów (FAIL)
#   *zwarac dokładnie który assert nie przeszedł
```

test_with_pytest.py

```python
import funkcje

# - Metody testowe muszą mieć nazwę zaczynającą się od 'test' (np. test_add),
# inaczej nie zostaną uruchomione przy automatycznym wykrywaniu.

def test_add():
    assert funkcje.add(2,4) == 6
    assert not funkcje.add(2,3) == 6

# URUCHOMINIE: python -m <nazwa_modułu> <nazwa_programu>
# python -m pytest test_with_pytest.py
# python -m pytest test_with_pytest.py -v (więcej informacji)

# - następnie zmieniamy returna w funckji na odejmowanie
#   i patrzymy jaki będzie wynik testów (FAIL)
#   *zwarac dokładnie który assert nie przeszedł
```

Zadanie 1. Palindrom

Napisz funkcję sprawdzającą czy dane słowo jest palindromem (palindrom to słowo które pisane od przodu i od tyłu jest identyczne).
Napisz testy, które sprawdzą, czy słowa “kamilslimak” i “ala” są palindromami, a słowa “wiadro” i “kamyk” nimi nie są.

Zadanie 2. Boki trójkąta

Napisz program, który sprawdzi, czy można utworzyć trójkąt z boków o podanej długości. Aby utworzyć trójkąt, suma dwóch krótszych boków musi być dłuższa niż najdłuższy bok.
Należy przetestować sytuację, w której zachodzi prawidłowy stosunek między bokami (2,3,4) sytuację, w której najdłuższy bok jest dłuższy od sumy dwóch pozostałych boków (1,1,2) i sytuację w której jeden z boków ma długość niedodatnią (0,-1,1)c

Zadanie 3. Dzielenie

Napisz program dzielący dwie liczby. Program powinien wyświetlić wynik dzielenia, a w przypadku dzielenia przez 0 komunikat “Nie dziel przez 0”.
(Te testy zrobimy wspólnie)
W tym teście zadanie jest nieco trudniejsze - naszym zadaniem jest sprawdzenie, czy funkcja wyświetliła prawidłowy wynik. W tym celu potrzebujemy przechwycić tekst z konsoli.

Zadanie 4. Sortowanie

Posortuj listę stringów według długości występujących w niej stringów.
Funkcja sort pozwala na ustalenie klucza według którego odbywa się sortowanie, w tym wypadku będzie to długość czyli len.
(Te testy zrobimy wspólnie)

Zadanie 5. Liczby pierwsze

Napisz program sprawdzający, czy liczba jest liczbą pierwszą. Liczba pierwsza to taka liczba naturalna, która dzieli się tylko przez 1 i samą siebie. Liczby 0 i 1 nie są liczbami pierwszymi.
Jeżeli liczba jest mniejsza od 2 wiemy że nie jest pierwsza - wynika to z definicji.
Następnie w pętli sprawdzamy czy znajdziemy dzielnik podanej liczby.
Jeżeli tak - wiemy, że liczba nie jest pierwsza,
jeżeli nie - liczba jest pierwsza, ponieważ nie ma żadnego innego dzielnika niż 1 i samą siebie.

Zadanie 6. Średnia z listy

Napisz program (bez korzystania z funkcji mean) który policzy średnią wartość z listy. Program powinien zwrócić -1 jeżeli lista jest pusta oraz -2 jeżeli w liście znajdują się wartości nie będące liczbami.