Pytest – Lista zadań

Dokumentacja Pytest: https://docs.pytest.org/en/latest/

Zadanie 1.

Napisz w Pythonie funkcję, która sprawdza, czy podana liczba jest liczbą pierwszą (funkcja powinna zwracać *True*, gdy liczba jest pierwsza oraz *False* w przeciwnym przypadku).
Funkcja powinna rzucać wyjątek z odpowiednim komunikatem, jeśli podany argument nie jest liczbą całkowitą.
Napisz następujące testy:

1. test sprawdzający czy dla podanych liczb funkcja zwraca odpowiednią wartość (użyj dekoratora *@pytest.mark.parametrize*: https://docs.pytest.org/en/latest/parametrize.html#pytest-mark-parametrize-parametrizing-test-functions),
2. test sprawdzający czy dla niepoprawnych argumentów funkcja rzuca oczekiwany wyjątek z odpowiednim komunikatem (wykorzystaj funkcję *pytest.raises:* https://docs.pytest.org/en/latest/assert.html#assertions-about-expected-exceptions i ponownie użyj dekoratora *@pytest.mark.parametrize*).

Zadanie 2.

Napisz testy do kodu załączonego do zadania (klasy *Store* i *Cart*).
W testach sprawdź działanie każdej z metod (możesz ponownie wykorzystać dekorator *@pytest.mark.parametrize* oraz funkcję *pytest.raises*).
Sprawdź do czego służy dektorator *@pytest.fixture* (https://docs.pytest.org/en/latest/fixture.html) i utwórz odpowiednie fixtures (możesz je nazwać odpowiednio *store* i *cart*).
Wewnątrz fixture *store* wczytaj ofertę z załączonego pliku *test_offer.json*.
Zastanów się jaki zasięg (*scope*) powinny mieć utworzone *fixtures*.

Uwaga: Testując metodę *show* z klasy *Cart* użyj jednej z wbudowanych fixtur: *capsys* lub *capfd* (https://docs.pytest.org/en/latest/capture.html).

Zadanie 3.

Sprawdź do czego służą znaczniki w Pytest (https://docs.pytest.org/en/latest/mark.html).
Oznacz kilka napisanych wcześniej testów znacznikiem *skip* i sprawdź czy oznaczone testy faktycznie zostaną pominięte przy uruchomieniu.
Oznacz kilka napisanych wcześniej testów znacznikiem *skipif* (np. test powinien być pomijany przy uruchomieniu starszą wersją Pythona). Sprawdź, czy oznaczone testy faktycznie zostaną pominięte przy uruchomieniu jeśli warunek jest spełniony oraz czy zostaną uruchomione jeśli warunek nie jest spełniony.
Utwórz własny znacznik i oznacz nim kilka napisanych wcześniej testów. Sprawdź jak uruchomić wszystkie testy oznaczone tym znacznikiem i jak uruchomić wszystkie testy oprócz testów oznaczonych tym znacznikiem.

Zadanie 4.

Zainstaluj wtyczkę *pytest-cov* i sprawdź pokrycie testowe w rozwiązanych zadaniach.
Do rozwiązań dodaj plik *pytest.ini* tak, aby Pytest zawsze sprawdzał pokrycie testowe.

Zadanie 5.

Zainstaluj wtyczkę *pytest-pep8*.
Do wcześniejszych rozwiązań dodaj plik *pytest.ini*, w którym ustawisz maksymalną długość linii. Następnie w jednym z plików *pytest.ini* zmień maksymalną długość linii na bardzo krótką (np. 10 znaków). Uruchom testy i sprawdź czy podane ograniczenie zostało zastosowane (powinny zostać wypisane wszystkie linie przekraczające podany limit).
Zobacz kod błędu dotyczącego przekroczenia maksymalnej długości linii i dodaj go do ignorowanych błędów (w pliku *pytest.ini*).
Uruchom testy ponownie i sprawdź czy faktycznie błąd został zignorowany.

Zadanie 6.

Dowiedz się na czym polega parametryzowanie fixtur w Pytest (https://docs.pytest.org/en/latest/fixture.html#parametrizing-fixtures).
Dowiedz się do czego służy wyrażenie *yield* wewnątrz fixtur (https://docs.pytest.org/en/latest/fixture.html#fixture-finalization-executing-teardown-code).
Korzystając z Selenium WebDriver stwórz fixture, która będzie inicjalizowała instancję przeglądarki:

- sparametryzuj fixture tak, aby tworzyła instancje Chrome i Firefox (testy korzystające z tej fixture będą uruchamiane wtedy dla obu przeglądarek),
- użyj wyrażenia *yield*, aby na koniec zamknąć przeglądarkę.

Napisz następujące testy korzystające z tej fixture:

- Test 1:
  1. Otwiera w przeglądarce stronę *page/index.html* (z załącznika do listy zadań).
  2. Sprawdza, czy po kliknięciu przycisku *„Open home page in new window”*, faktycznie otworzyła się strona główna w nowym oknie.
  3. Zamyka nowo otwarte okno.
- Test 2:
  1. Otwiera w przeglądarce stronę *page/index.html* (z załącznika do listy zadań).
  2. Przechodzi do podstrony „*Contact*”.
  3. Czeka, aż na stronie pojawi się formularz.
  4. Wypełnia wszystkie pola formularza.
  5. Sprawdza, czy pola zostały wypełnione zgodnie z oczekiwaniami.
  6. Zatwierdza wypełniony formularz.
  7. Sprawdza, czy tekst w alercie jest poprawny i zatwierdza alert.
- Test 3:
  1. Otwiera w przeglądarce stronę *page/index.html* (z załącznika do listy zadań).
  2. Przechodzi do podstrony „*Gallery*”.
  3. Sprawdza, czy w galerii jest dokładnie 11 zdjęć.