

How to run unittest

-To run it and being able to see print output

```
python3 -s test1.py
```

-If everything goes fine you should have this output

```
-----  
Ran 7 tests in 0.070s  
  
OK  
root@92c08c213c16:/codice#
```

-Super brief explanation on how the tests work

SetUp method runs before each test to “make the environment”

```
def setUp(self):  
    # populate data before each test by doing two POST  
    #TODO make it read data from a csv or similar to avoid students hardcoding answers  
    self.url_to_shorten_1="https://en.wikipedia.org/wiki/Docker_(software)"  
    self.url_to_shorten_2="https://fastapi.tiangolo.com"  
    self.id_shortened_url_1=""  
    self.id_shortened_url_2=""  
  
    def do_post(url_to_shorten):  
        endpoint = "/"  
        print(url_to_shorten)  
        url = f"{self.base_url}{endpoint}"  
        response = requests.post(url, json={'value': str(url_to_shorten)})  
        self.assertEqual(response.status_code, 201, f"Expected status code 201, but got {response.status_code}")  
        self.assertIsNotNone(response.text, "Response text should not be None.")  
        response_extracted=response.json()  
        id_returned=response_extracted["id"]#TODO add try catch / handle in case no id key  
        return id_returned  
  
    self.id_shortened_url_1=do_post(self.url_to_shorten_1)  
    print("id 1 obtained "+str(self.id_shortened_url_1))  
    self.id_shortened_url_2=do_post(self.url_to_shorten_2)  
    print("id 2 obtained "+str(self.id_shortened_url_2))
```

tearDown method runs after each test to “clean the environment”

```
def tearDown(self): #OK  
    #erase everything  
    endpoint = "/"  
    url = f"{self.base_url}{endpoint}"  
    response = requests.delete(url)  
    #self.assertEqual(response.status_code, 204)
```

Then there are the tests eg.

```
"""
/:id GET
Given an ID to the service, redirects the user to t
If the ID does not exist, 404 should be returned.
"""
def test_get_request_with_id_expect_301(self):

    id = self.id_shortened_url_1
    expected_value = self.url_to_shorten_1

    endpoint = "/"
    url = f"{self.base_url}{endpoint}{id}"
    response = requests.get(url)

    self.assertEqual(response.status_code, 301, f"E

    self.assertEqual(response.json().get("value"),
```

Feel free to modify the tests while developing, but when you are going to get evaluated you are going to be graded against the untouched version of the tests. So your final version should pass the tests against the vanilla version.

PS don't try to hardcode you answer cause the values passed by the tests are going to be different when marking.

The only section of the tests that you can tweak is the base url, in order to adapt it to the ip and port you are running it on

```
ass TestApi(unittest.TestCase):
    #with fastapi default port is 8000 with flask is 5000
    base_url = "http://127.0.0.1:8000"
```

Tests are available at “link/source where we keep the tests”

Tests use request to query your API, you are going to need request, installable with:
pip install requests