

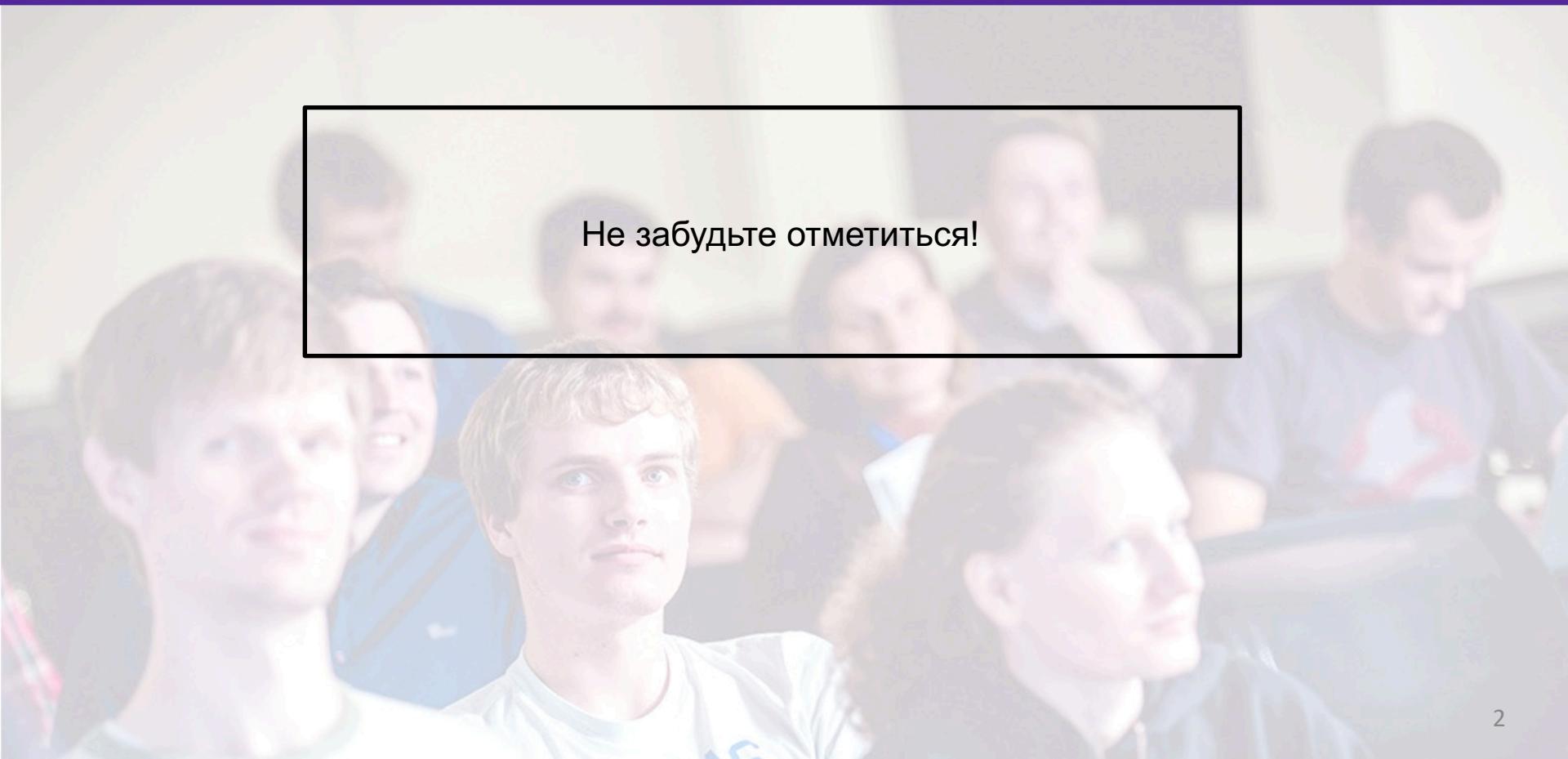
ТЕХНОАТОМ



Автоматизация тестирования на Python

Вводная лекция

Илья Кириллов
Ярослав Чередниченко



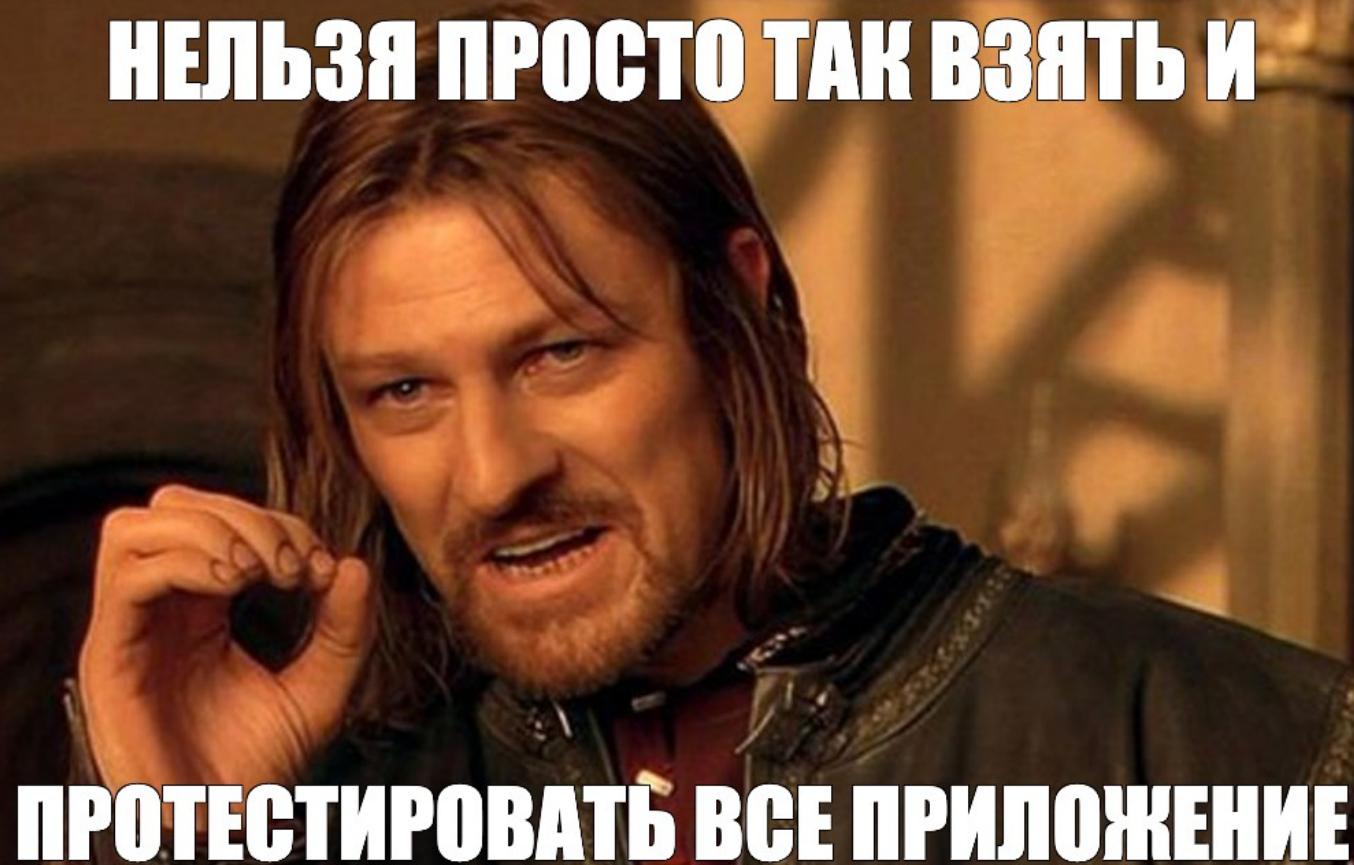
Не забудьте отметить!

Основные тезисы

- Цель курса – освоить профессию Software Developer Engineer in Test
- Научимся программировать на языке Python
- 12 лекций
- Домашнее задание раз в 2-3 лекции
- Разработка – групповая
- Итоговый проект

Что такое тестирование?

Тестирование – процесс демонстрации
отсутствия ошибок в приложении.



Что такое тестирование?

Тестирование – процесс демонстрации того, что приложение работает правильно.

Что такое тестирование?

Тестирование – процесс выполнения программы с целью поиска ошибок.



Атрибуты качества ПО

- Функциональность
- Надежность
- Практичность/удобство
- Эффективность
- Сопровождаемость
- Мобильность

Методы обеспечения качества ПО

- Статический:
 - Статический анализ кода
 - Проверка спецификаций и требований
- Динамический:
 - Тестирование
 - Профилирование
 - Динамический анализ
- Формальный:
 - Верификация моделей

Немного терминов

- Тестировщик (тестер, тостер)
- Тест (test-case)
- Сценарий
- Баг (дефект, ошибка)
- Билд (сборка)
- Покрытие

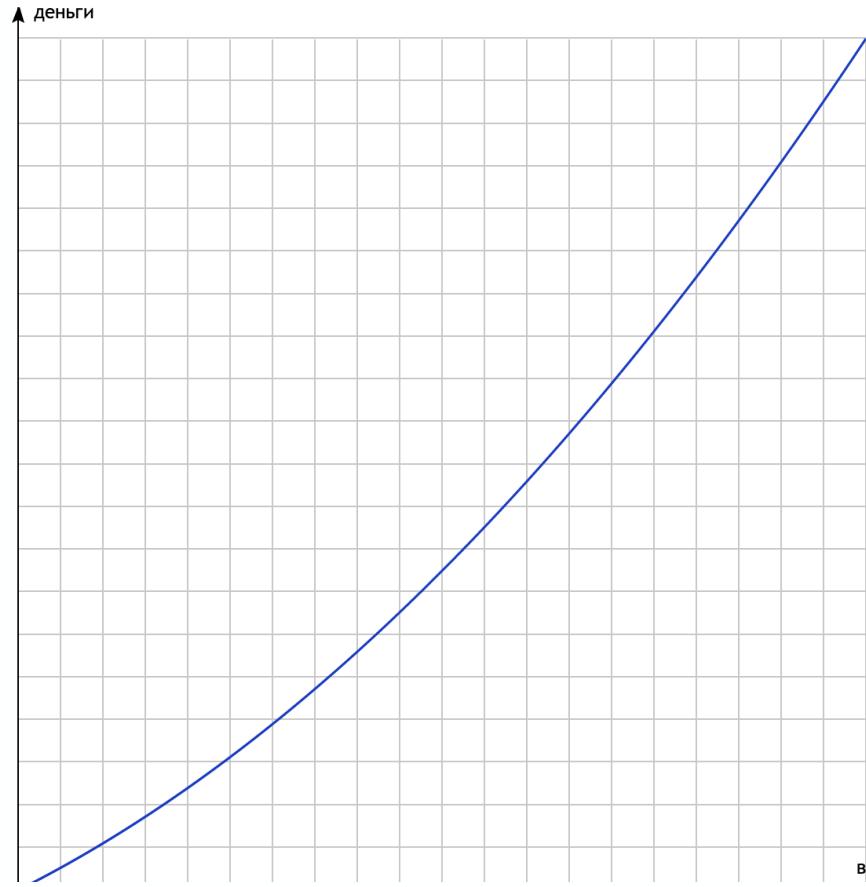
Покрытие кода (code coverage)

- Line coverage
- Branch coverage

```
a = input()  
b = input()  
  
if int(a) > 0 or int(b) < 10:  
    print('Success')  
else:  
    print('Failure')
```

Кто еще может заниматься тестированием?

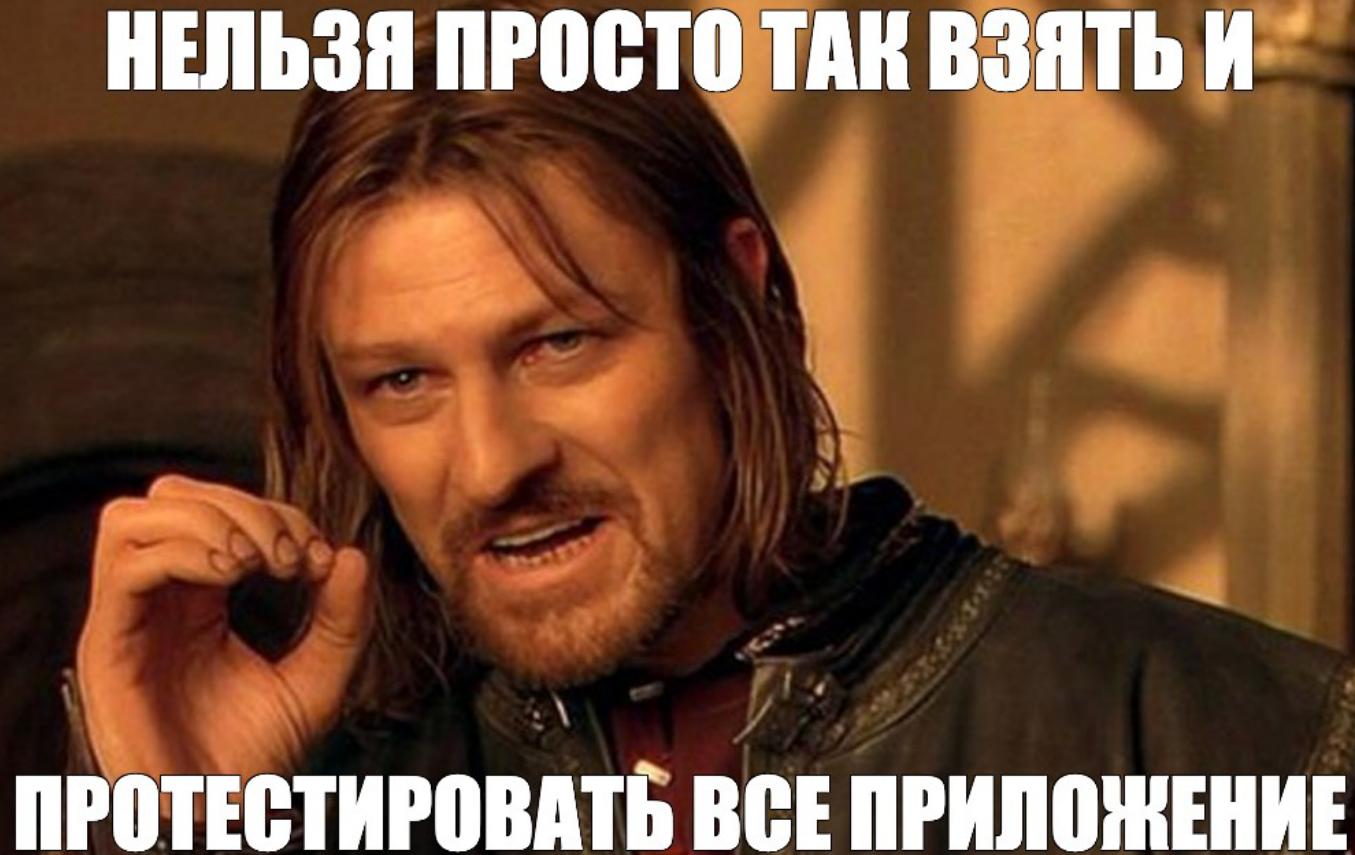
- Разработчик
- Менеджер
- Заказчик



Сложности тестирования

- Тестов - много, времени – мало

Решение: необходимо создать ограниченный набор тестов, который позволит протестировать приложение, не утратив при этом качество тестирования



Техники тест-дизайна

- Классы эквивалентности
- Границные условия
- Таблица принятия решений
- Граф потока управления
- Граф потока данных
- Pair-wise

Классы эквивалентности

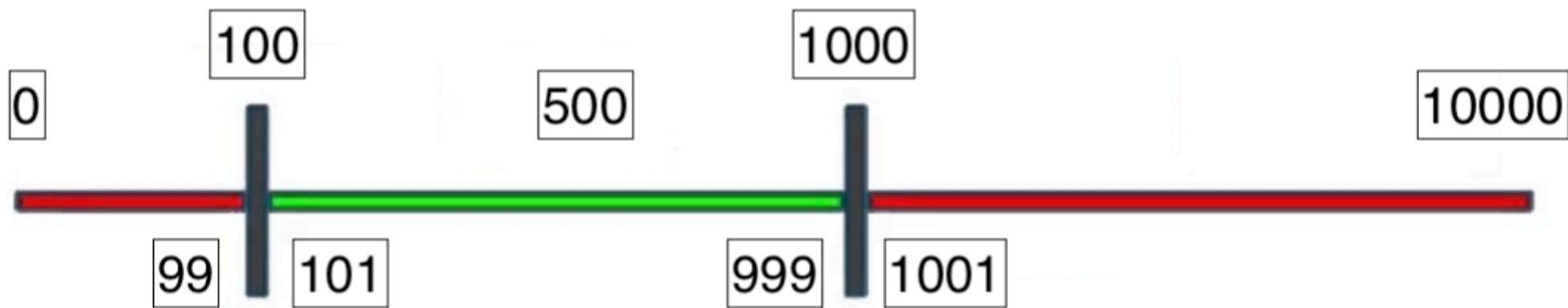
- Классы эквивалентности – наборы входных параметров, данные которых система считает эквивалентными.
- Тесты эквивалентны тогда, когда:
 - они тестируют одно и то же
 - если один из них выявляет ошибку, то и остальные ее выявят
 - если один из них не выявляет ошибку, то и остальные ее не выявят

Введение в тестирование ПО



Границевые условия

- Границевые условия (значения) – это ситуации перед границей, на границе и за границей входных условий.



Pair-wise

- Pair-wise - формирование таких наборов данных, в которых каждое тестируемое значение каждого из проверяемых параметров хотя бы единожды сочетается с каждым тестируемым значением всех остальных проверяемых параметров

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	1	1	1	1	1	1
3	1	1	0	1	0	1	0	1	0	1
4	1	0	1	0	1	0	1	0	1	0

Тестовые наборы

- Smoke
- Regression
- Acceptance

Введение в тестирование ПО

Header - wrong image shown

[Edit](#) [Comment](#) [Assign](#) [To Do](#) [In Progress](#) [Done](#) [...!\[\]\(3ea0b4bbc25be831ea423abcf9c7ab1a_img.jpg\)](#)

Type:	<input checked="" type="checkbox"/> Bug	Status:	TO DO (View workflow)	Assignee:	 Unassigned Assign to me
Priority:	 Medium	Resolution:	Unresolved	Reporter:	 Usersnap Developer
Labels:	None			Votes:	 0
Environment:	Chrome Version 70.0.3538.77 (Official Build) (64-bit) - Screen Resolution: 1920 x 947			Watchers:	 1 Stop watching this issue
Sprint:				Created:	14 minutes ago
Description				Updated:	Just now
Components:	on homepage				
Description:	Outdated header image on www.usersnap.com			Development	
<i>Steps to reproduce:</i>				Create branch	
1) Go to usersnap.com					
2) check the header image					
<i>Javascript console errors:</i>				Agile	
no error messages				View on Board	
Result:					
<i>expected Result:</i>					
header_image_2.jpeg					
<i>actual Result:</i>					
header_image_1.jpeg					

Attachments 

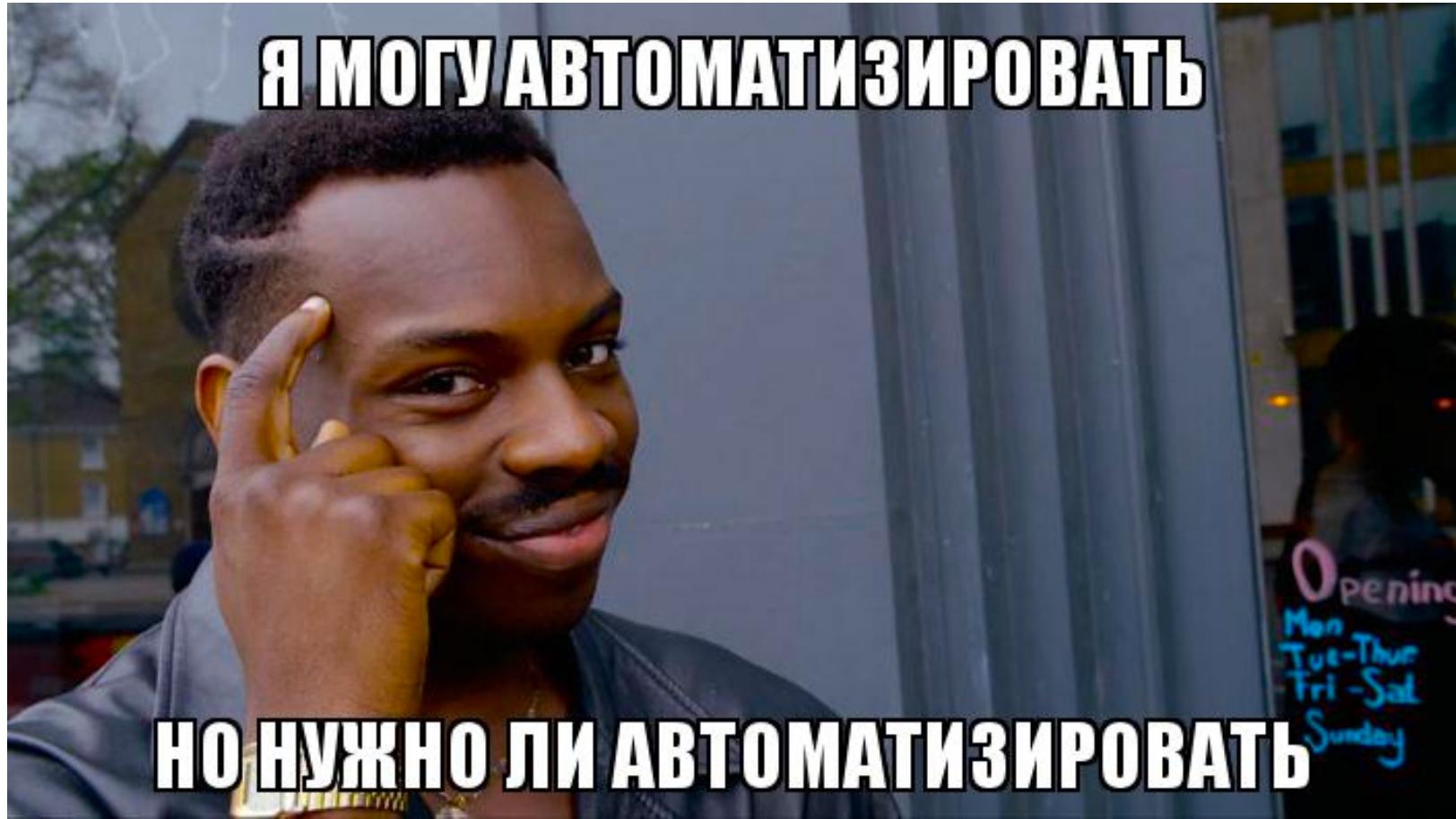
Drop files to attach, or [browse](#).

Автоматизированное тестирование

- Введение в автоматизацию тестирования
- Цели
- Виды
- Инструменты
- Инфраструктура

Что такое автоматизированное тестирование?

Это процесс верификации программного обеспечения, при котором основные функции и шаги теста, такие как запуск, инициализация, выполнение, анализ и выдача результата, выполняются автоматически



Преимущества автоматизированного тестирования

- Быстрое выполнение
- Повторяемость
- Меньше затрат на поддержку
- Отчеты

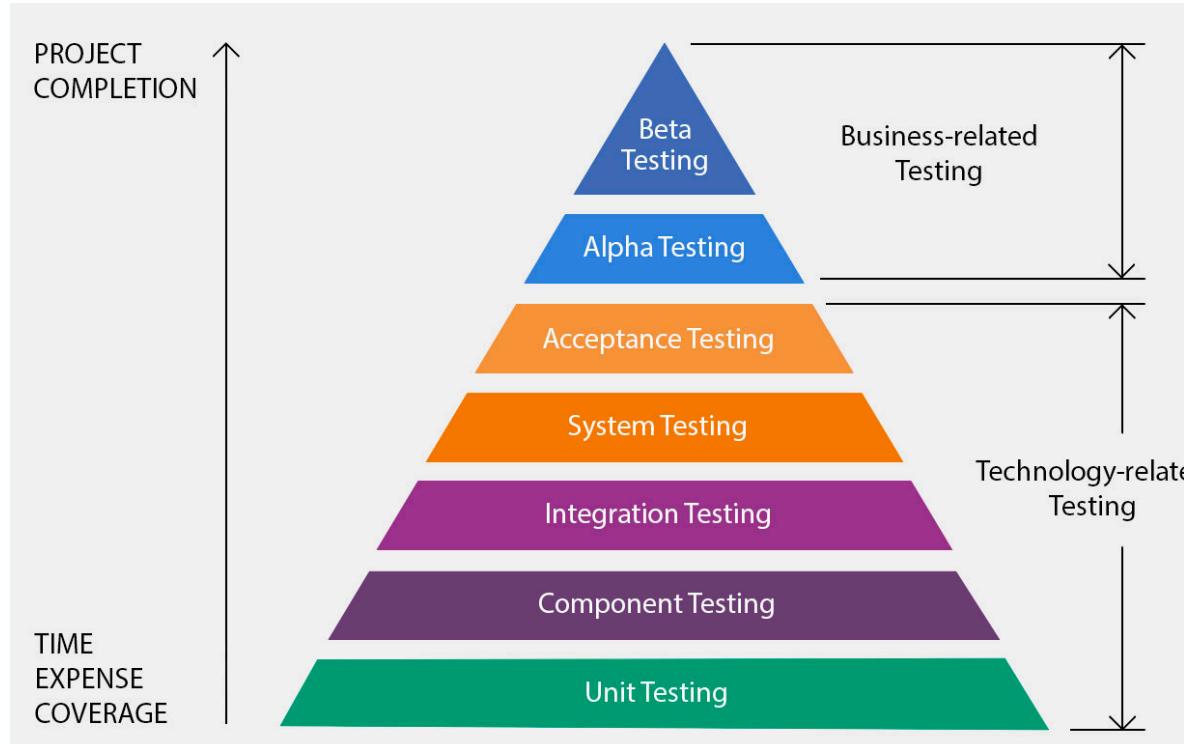
Недостатки автоматизации тестирования

- Повторяемость
- Затраты на поддержку
- Большие затраты на разработку
- Пропуск мелких ошибок

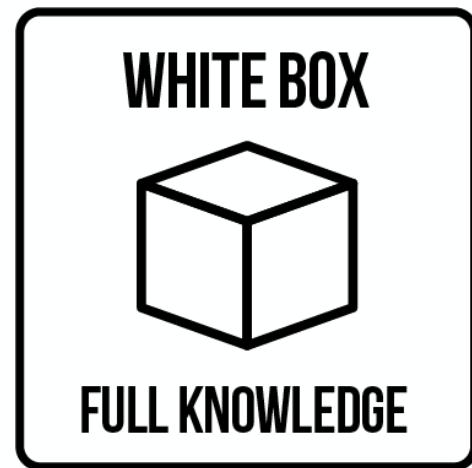
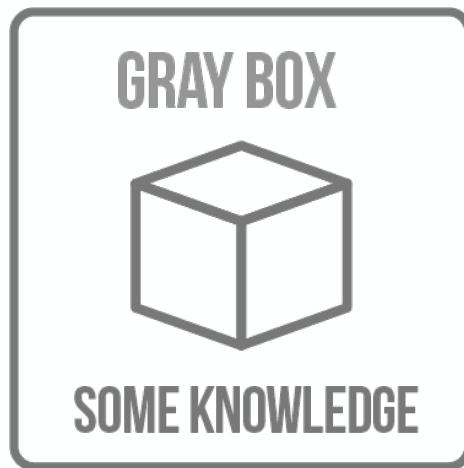
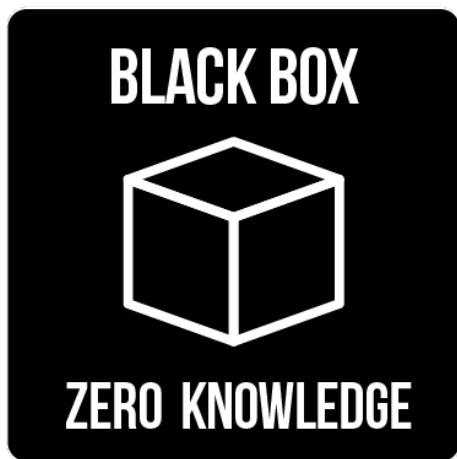
Цели автоматизации тестирования

- Повышение производительности труда
- Повышение надежности
- Повышение скорости разработки

Уровни тестирования

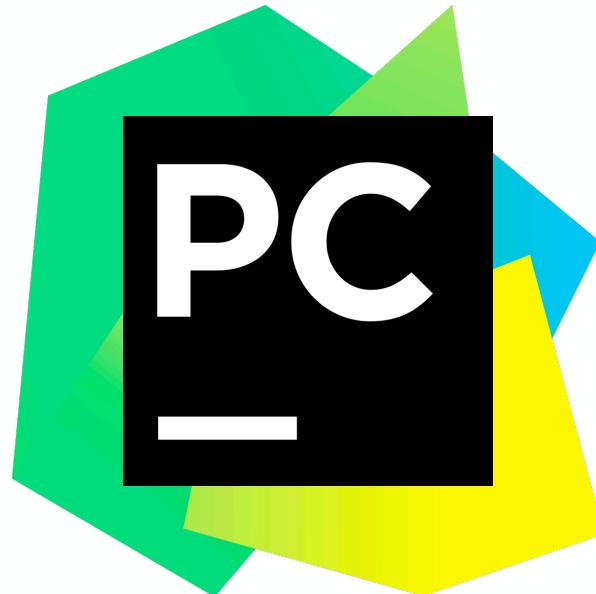


Методы тестирования





PyCharm



Виртуальное окружение

Виртуальное окружение - это изолированное окружение среды (в нашем случае это окружение Python), которое позволяет нам использовать определенные версии приложений.

Pipenv

Для установки нам необходимо выполнить команду:

```
pip3 install pipenv
```

```
$ pip3 install pipenv
Collecting pipenv
  Downloading https://files.pythonhosted.org/packages/13/b4/3ffa55f77161cff9a5220f162670f7c5eb00df52e00939e203f601b0f579/pipenv-2018.11.26-py3-none-any.whl (5.2MB)
    |████████| 5.2MB 762kB/s
Requirement already satisfied: pip>=9.0.1 in /usr/local/lib/python3.7/site-packages (from pipenv) (19.3.1)
Collecting virtualenv-clone>=0.2.5
  Using cached https://files.pythonhosted.org/packages/ba/f8/50c2b7dbc99e05fce5e5b9d9a31f37c988c99acd4e8dedd720b7b8d4011d/virtualenv_clone-0.5.3-py2.py3-none-any.whl
Requirement already satisfied: certifi in /usr/local/lib/python3.7/site-packages (from pipenv) (2019.3.9)
Requirement already satisfied: setuptools>=36.2.1 in /usr/local/lib/python3.7/site-packages (from pipenv) (42.0.2)
Collecting virtualenv
  Downloading https://files.pythonhosted.org/packages/33/4e/9f3e70cd2bf3de603bf4de46cb60af1309c8d97fd81cbde317507a57573e/virtualenv-20.0.5-py2.py3-none-any.whl (4.6MB)
    |████████| 4.6MB 10.6MB/s
Collecting appdirs<2,>=1.4.3
  Downloading https://files.pythonhosted.org/packages/56/eb/810e700ed1349edde4cbdc1b2a21e28cdf115f9faf263f6bbf8447c1abf3/appdirs-1.4.3-py2.py3-none-any.whl
Collecting distlib<1,>=0.3.0
  Downloading https://files.pythonhosted.org/packages/7d/29/694a3a4d7c0e1aef76092e9167fbe372e0f7da055f5dcf4e1313ec21d96a/distlib-0.3.0.zip (571kB)
    |████████| 573KB 17.2MB/s
Collecting importlib-metadata<2,>=0.12; python_version < "3.8"
  Downloading https://files.pythonhosted.org/packages/8b/03/a00d504808808912751e64ccf414be53c29cad620e3de2421135fcac3025/importlib_metadata-1.5.0-py2.py3-none-any.whl
Requirement already satisfied: six<2,>=1.9.0 in /usr/local/lib/python3.7/site-packages (from virtualenv->pipenv) (1.12.0)
Collecting filelock<4,>=3.0.0
  Downloading https://files.pythonhosted.org/packages/93/83/71a2ee6158bb9f39a90c0dea1637f81d5eef866e188e1971a1b1ab01a35a/filelock-3.0.12-py3-none-any.whl
Collecting zipp>=0.5
  Downloading https://files.pythonhosted.org/packages/6f/6d/a55f6e81ac213942b9a19cbc05b560c726c3e16f8fb17555f059c17d65f2/zipp-3.0.0-py3-none-any.whl
Building wheels for collected packages: distlib
  Building wheel for distlib (setup.py) ... done
  Created wheel for distlib: filename=distlib-0.3.0-cp37-none-any.whl size=340428 sha256=ec58c3ba3f71d85cb93e223ff4846e82fc1dd357725a361fed791331a05a9a0f
  Stored in directory: /Users/y.cherednichenko/Library/Caches/pip/wheels/6e/e8/db/c73dae4867666e89ba3cfbc4b5c092446f0e584eda6f409cbb
Successfully built distlib
Installing collected packages: virtualenv-clone, appdirs, distlib, zipp, importlib-metadata, filelock, virtualenv, pipenv
Successfully installed appdirs-1.4.3 distlib-0.3.0 filelock-3.0.12 importlib-metadata-1.5.0 pipenv-2018.11.26 virtualenv-20.0.5 virtualenv-clone-0.5.3 zipp-3.0.0
```

Создаем директорию с тестовым проектом:

```
mkdir pipenv_test
```

Перемещаемся в созданную директорию:

```
cd pipenv_test
```

Создаем виртуальное окружение:

```
pipenv --python3
```

Устанавливаем пакет requests:

```
pipenv install requests
```

Инфраструктура

Инфраструктура

```
~/PycharmProjects/pipenv_test
$ cat Pipfile
[source]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true

[dev-packages]

[packages]
requests = "*"

[requires]
python_version = "3.7"

~/PycharmProjects/pipenv_test
$ cat Pipfile.lock
{
    "_meta": {
        "hash": {
            "sha256": "bb57e0d7853b45999e47c163c46b95bc2fde31c527d8d7b5b5539dc979444a6d"
        },
        "pipfile-spec": 6,
        "requires": {
            "python_version": "3.7"
        },
        "sources": [
            {
                "name": "pypi",
                "url": "https://pypi.org/simple",
                "verify_ssl": true
            }
        ]
    },
    "default": {
        "certifi": {
            "hashes": [
                "sha256:017c25db2a153ce562900032d5bc68e9f191e44e9a0f762f373977de9df1fbb3",
                "sha256:25b64c7da4cd7479594d035c08c2d809eb4aab3a26e5a990ea98cc450c320f1f"
            ],
            "version": "2019.6.29"
        }
    }
}
```

Для активации виртуального окружения:

pipenv shell

Для запуска кода:

pipenv run python 1.py

```
~/PycharmProjects/pipenv_test
$ pipenv shell
Launching subshell in virtual environment...
. /Users/y.cherednichenko/.local/share/virtualenvs/pipenv_test-54YYcHA_/_bin/activate
```

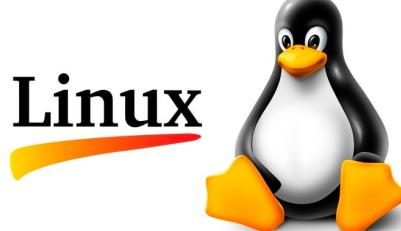
```
Identity added: /Users/y.cherednichenko/.ssh/id_rsa (/Users/y.cherednichenko/.ssh/id_rsa)
```

```
~/PycharmProjects/pipenv_test
$ . /Users/y.cherednichenko/.local/share/virtualenvs/pipenv_test-54YYcHA_/_bin/activate
```

```
~/PycharmProjects/pipenv_test > pipenv test-54YYcHA_
$ vim 1.py
```

```
~/PycharmProjects/pipenv_test > pipenv test-54YYcHA_
$ pipenv run python 1.py
WOW
```

Использование различных ОС при автоматизации тестирования



MacTMOS



Почему py.test?

```
def test_1():
    assert 1 == 2

test_1()
```

```
(venv) i-kirillov:~ python lection1/test_basic.py
Traceback (most recent call last):
  File "lection1/test_basic.py", line 8, in <module>
    test_1()
  File "lection1/test_basic.py", line 5, in test_1
    assert 1 == 2
AssertionError
```

Почему py.test?

- 300+ плагинов
- Минималистичность
- Параметризация из коробки
- Для проверок теста используется ключевое слово assert, а не assertEquals() или assertLessThan(). Просто assert!
- Фикстуры
- Можно использовать для запуска тестов, написанных для unittest или nosetests
- Очень подробная документация

```
import pytest

def test_1():
    assert 1 == 2

class Test2:

    def test_2(self):
        assert 1 != 2

    def test_3(self):
        with pytest.raises(ZeroDivisionError):
            assert 1 / 0
```

```
(venv) i-kirillov:% pytest lection1/test_basic.py
=====
 test session starts =====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items

lection1/test_basic.py F..
[100%]

=====
 FAILURES =====
 test_1
-----
def test_1():
>     assert 1 == 2
E     assert 1 == 2

lection1/test_basic.py:5: AssertionError
=====
 1 failed, 2 passed in 0.09s =====
```

```
(venv) i-kirillov:% pytest -v lection1/test_basic.py
=====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items

lection1/test_basic.py::test_1 FAILED
[ 33%]
lection1/test_basic.py::Test2::test_2 PASSED
[ 66%]
lection1/test_basic.py::Test2::test_3 PASSED
[100%]

=====
FAILURES =====
----- test_1 -----
----- test_1 -----
def test_1():
>     assert 1 == 2
E     assert 1 == 2
E     -1
E     +2

lection1/test_basic.py:5: AssertionError
===== 1 failed, 2 passed in 0.09s =====
```

Фикстуры

```
import random
import pytest

@pytest.fixture()
def random_value():
    print('entering')
    yield random.randint(0, 100)
    print('exiting')
```

```
def test(random_value):
    assert random_value > 50
```

```
(venv) i-kirillov:% pytest lection1/test_with_fixture.py
=====
test session starts =====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 1 item

lection1/test_with_fixture.py F [100%]

=====
FAILURES =====
-----
test -----
-----
random_value = 37

def test(random_value):
>     assert random_value > 50
E     assert 37 > 50

lection1/test_with_fixture.py:13: AssertionError
----- Captured stdout setup -----
entering
----- Captured stdout teardown -----
exiting
===== 1 failed in 0.07s =====
```

Фикстуры: scopes

- Function
- Class
- Module
- Session

```
import random
import pytest

@pytest.fixture(scope='function')
def function_fixture():
    return random.randint(0, 100)

@pytest.fixture(scope='class')
def class_fixture():
    return random.randint(0, 100)

@pytest.fixture(scope='session')
def session_fixture():
    return random.randint(0, 100)
```

Фикстуры: scopes

```
def test1(function_fixture, session_fixture):
    print(f"func fixture:{function_fixture}")
    print(f"session fixture:{session_fixture}")

def test2(function_fixture, session_fixture):
    print(f"func fixture:{function_fixture}")
    print(f"session fixture:{session_fixture}")

def test3(function_fixture, session_fixture):
    print(f"func fixture:{function_fixture}")
    print(f"session fixture:{session_fixture}")
```

```
(venv) i-kirillov:% pytest -s -v lection1/test_scopes.py
=====
test session starts =====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, infile: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items

lection1/test_scopes.py::test1 func fixture:100
session fixture:58
PASSED
lection1/test_scopes.py::test2 func fixture:7
session fixture:58
PASSED
lection1/test_scopes.py::test3 func fixture:72
session fixture:58
PASSED

===== 3 passed in 0.04s =====
```

```
class Test3:

    def test3_1(self, function_fixture, class_fixture, session_fixture):
        print(f"func fixture:{function_fixture}")
        print(f"class fixture:{class_fixture}")
        print(f"session fixture:{session_fixture}")

    def test3_2(self, function_fixture, class_fixture, session_fixture):
        print(f"func fixture:{function_fixture}")
        print(f"class fixture:{class_fixture}")
        print(f"session fixture:{session_fixture}")

    def test3_3(self, function_fixture, class_fixture, session_fixture):
        print(f"func fixture:{function_fixture}")
        print(f"class fixture:{class_fixture}")
        print(f"session fixture:{session_fixture}")
```

```
(venv) i-kirillov:% pytest -s -v lection1/test_scopes.py
=====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, iniﬁle: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items

lection1/test_scopes.py::Test4::test4_1 func fixture:42
class fixture:39
session fixture:100
PASSED
lection1/test_scopes.py::Test4::test4_2 func fixture:4
class fixture:39
session fixture:100
PASSED
lection1/test_scopes.py::Test4::test4_3 func fixture:63
class fixture:39
session fixture:100
PASSED

=====
 3 passed in 0.05s =====
```

Фикстуры: advanced

```
import pytest

@pytest.fixture(autouse=True)
def new_file(random_file):
    f = open(random_file, 'w')
    yield
    f.close()
    os.remove(random_file)
```

```
import os
from random import randint

@pytest.fixture()
def random_file():
    return str(randint(0, 1000))
```

Параметризация

```
def test():
    for i in range(10):
        assert i % 2 == 0
```

```
def test():
    errors = []
    for i in range(10):
        try:
            assert i % 2 == 0
        except AssertionError:
            errors.append(f'{i} not even')

    assert not errors
```

```
(venv) i-kirillov:~ pytest -s lection1/test_parametrization.py
=====
test session starts =====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1
rootdir: /Users/i.kirillov/PycharmProjects, iniﬁle: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 1 item

lection1/test_parametrization.py F
=====
FAILURES =====
test
-----
def test():
    errors = []

    for i in range(10):
        try:
            assert i % 2 == 0
        except AssertionError:
            errors.append(f'{i} not yven')

>     assert not errors
E     AssertionError: assert not ['1 not yven', '3 not yven', '5 not yven', '7 not yven', '9 not yven']

lection1/test_parametrization.py:19: AssertionError
=====
1 failed in 0.08s =====
```

Параметризация

```
import pytest

@pytest.mark.parametrize('i', list(range(10)))
def test_even(i):
    """
    :param i: range of integers
    Parametrized test which checks that number if even.
    """

    assert i % 2 == 0
```

py.test: введение

```
(venv) i-kirillov:~ pytest -v lection1/test_parametrization.py
=====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 10 items

lection1/test_parametrization.py::test_even[0] PASSED [ 10%]
lection1/test_parametrization.py::test_even[1] FAILED [ 20%]
lection1/test_parametrization.py::test_even[2] PASSED [ 30%]
lection1/test_parametrization.py::test_even[3] FAILED [ 40%]
lection1/test_parametrization.py::test_even[4] PASSED [ 50%]
lection1/test_parametrization.py::test_even[5] FAILED [ 60%]
lection1/test_parametrization.py::test_even[6] PASSED [ 70%]
lection1/test_parametrization.py::test_even[7] FAILED [ 80%]
lection1/test_parametrization.py::test_even[8] PASSED [ 90%]
lection1/test_parametrization.py::test_even[9] FAILED [100%]

=====
===== FAILURES =====
test_even[1] -----
i = 1

    @pytest.mark.parametrize('i', list(range(10)))
def test_even(i):
    """
    :param i: range of integers
    Parametrized test which checks that number is even.
    """
    assert i % 2 == 0
E    assert 1 == 0
E      -1
E      +0

lection1/test_parametrization.py:11: AssertionError
```

Маркировка и фильтрация

```
import pytest

@pytest.mark.smoke
def test1():
    pass

@pytest.mark.smoke
def test2():
    pass

@pytest.mark.regress
def test3():
    pass
```

```
(venv) i-kirillov:~ pytest -v -m smoke lection1/test_marks.py
=====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items / 1 deselected / 2 selected

lection1/test_marks.py::test1 PASSED [ 50%]
lection1/test_marks.py::test2 PASSED [100%]

=====
warnings summary =====
/Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323
  /Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323: PytestUnknownMarkWarning: Unknown py
test.mark.smoke - is this a typo? You can register custom marks to avoid this warning - for details, see https://docs.pytest.org/en/latest/mark.html
    warnings.warn(
/Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323
  /Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323: PytestUnknownMarkWarning: Unknown py
test.mark.regress - is this a typo? You can register custom marks to avoid this warning - for details, see https://docs.pytest.org/en/latest/mark.html
    warnings.warn(
-- Docs: https://docs.pytest.org/en/latest/warnings.html
=====
2 passed, 1 deselected, 2 warnings in 0.07s =====
```

```
(venv) i-kirillov:% pytest -v lection1/test_marks.py -k 'test1 or test3'
=====
platform darwin -- Python 3.8.1, pytest-5.3.2, py-1.8.1, pluggy-0.13.1 -- /Users/i.kirillov/PycharmProjects/target/venv/bin/python
cachedir: .pytest_cache
rootdir: /Users/i.kirillov/PycharmProjects, ini file: pytest.ini
plugins: timeout-1.3.4, xdist-1.31.0, ordering-0.6, forked-1.1.3, allure-pytest-2.8.6
collected 3 items / 1 deselected / 2 selected

lection1/test_marks.py::test1 PASSED [ 50%]
lection1/test_marks.py::test3 PASSED [100%]

=====
warnings summary =====
/Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323
  /Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323: PytestUnknownMarkWarning: Unknown py
test.mark.smoke - is this a typo? You can register custom marks to avoid this warning - for details, see https://docs.pytest.org/en/latest/mark.html
    warnings.warn(
      /Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323
      /Users/i.kirillov/PycharmProjects/target/venv/lib/python3.8/site-packages/_pytest/mark/structures.py:323: PytestUnknownMarkWarning: Unknown py
test.mark.regress - is this a typo? You can register custom marks to avoid this warning - for details, see https://docs.pytest.org/en/latest/mark.html
    warnings.warn(
-- Docs: https://docs.pytest.org/en/latest/warnings.html
=====
  2 passed, 1 deselected, 2 warnings in 0.05s =====
```

Структура проекта

```
ROOT
└── fixtures
    └── fixtures.py
└── tests
    ├── test_1.py
    └── test_2.py
└── tools
    └── helpers.py
└── conftest.py
└── pytest.ini
```

```
# content of pytest.ini
[pytest]
markers =
    smoke: mark a test as a smoke.
    regress: mark a test as a regress
```

Домашнее задание

1. Подготовка рабочего окружения для разработки автоматических тестов.
2. Написать по 5 автотестов с учетом техник тест-дизайна на каждую из структур данных в Python:
 1. List
 2. Set
 3. Dictionary
 4. String
 5. Int
3. Как минимум один тест на каждую структуру должен быть параметризован.
4. Как минимум один тест на каждую структуру должен быть в классе.
5. Проверить стиль кода линтером (pylint, flake8, pycodestyle).

ТЕХНОАТОМ



Спасибо за внимание!

Илья Кириллов

i.kirillov@corp.mail.ru

Ярослав Чередниченко

y.cherednichenko@corp.mail.ru