



Урок 3. Python: Docstring и проверяемые примеры

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 3 из 33 · 45 минут · Модуль «Документирование программы. Docstring, PEP 257»

Уровень: начинающий программист; Python 3.

Программа: <https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

В документации написано, что 125 секунд — это две полные минуты. Превратим пример в проверку, которая заметит случайную замену // на /.

Результат урока: Написать docstring с исполняемыми примерами, запустить doctest и отличить проверку примера от доказательства всех случаев.

01. Проблемная ситуация

0-5 минут. Документация обещает `full_minutes(125) → 2`. Автор заменил // на / и получил 2.0833...

Предложите, как обнаруживать такую ошибку без ручного просмотра после каждой правки. Запишите ещё два входа, важные для минуты как единицы.

02. Понятия и разобранный пример

5-14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Документация, которую можно исполнить

Строка документации первой инструкцией функции описывает её назначение. По PEP 257 краткое резюме отделяют пустой строкой от подробностей; тройные кавычки удобны для многострочного текста. В примере ниже `>>>` обозначает команду интерактивного сеанса, следующая строка — ожидаемое текстовое представление результата.

Модуль `doctest` находит такие примеры, выполняет и сравнивает фактический вывод с ожидаемым. Для числа 2 и 2.0 текст различен. Если функция возвращает значение, интерактивный пример показывает его представление; это не означает, что внутри функции обязателен `print`.

Сохраните файл `lesson.py` и выполните `python lesson.py`. Флаг `verbose=True` показывает ход проверки. Строка `if __name__ == "__main__":` запускает тесты при непосредственном выполнении файла; при импорте они не стартуют сами. Нет сторонних библиотек и регистрации. Успех трёх примеров говорит о трёх примерах, а не обо всех возможных входах.

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

`125//60=2,0//60=0,60//60=1`. Запуск этого полного файла сообщает об успешных трёх тестах. Если заменить // на /, результаты станут вещественными: проверка увидит несовпадение, в том числе 0.0 вместо 0. Верните // перед дальнейшей работой.

```
def full_minutes(seconds):
    """Вернуть полные минуты; seconds – целое 0...3600.

    >>> full_minutes(125)
    2
    >>> full_minutes(0)
    0
    >>> full_minutes(60)
    1
    """
    return seconds // 60

if __name__ == "__main__":
    import doctest
    doctest.testmod(verbose=True)
```



Команда	>>> full_minutes(125)
Ожидание	2
Правильный код	125 // 60 → 2
Изменение /	2.0833... → несовпадение

Docstring хранит команду и ожидаемый результат; doctest выполняет команду и сравнивает вывод.

03. Практика 1. Проверяем материал

Пример до и после границы

14–25 минут. Добавьте в docstring пять примеров из таблицы к трём уже имеющимся. Повтор 60 допустим для упражнения, но в рабочем наборе его можно удалить.

seconds	Полные минуты
59	0
60	1
61	1
3599	59
3600	60

2 минуты — прогноз; 4 — примеры; 3 — успешный запуск, временная замена // на / и повторный запуск; 2 — возврат правильного кода. Продукт: файл и запись «какая правка вызвала какой сбой». Критерии: 8 тестов при сохранённом повторе, точные ожидаемые строки, исправление после опыта.

04. Практика 2. Перенос способа

Новая функция и собственные doctest

25–33 минуты. Создайте seconds_left(seconds), возвращающую остаток секунд внутри минуты для целого 0...3600. Включите doctest для 0,59,60,125,3600. За 2 минуты вычислите ожидания, за 4 напишите полный файл, за 2 запустите. Критерии: собственный контракт, %, пять верных примеров.

05. Обсуждение и границы вывода

33–40 минут. Можно ли после пяти успешных примеров удалить описание ограничений? Что произойдёт, если ожидаемый ответ ошибочно переписать с неправильной программы? Обсудите независимый расчёт $125=2\cdot60+5$ и необходимость граничных примеров.

06. Индивидуальная проверка

40–45 минут. Напишите две строки doctest для seconds_left(121); назовите результат full_minutes(121) и объясните, доказывают ли они работу на всех входах. По баллу за команду, ожидание, число минут и объяснение.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Все пять новых результатов приведены в таблице. С исходными тремя получается 8 примеров. При удалении повторного (60) — 7. Не меняйте ожидаемое 2 на 2.0833... ради зелёного отчёта: сначала сверяйте с контрактом полных минут.

Практика 2: разбор и критерии

Ответы 0,59,0,5,0. Остаток всегда от 0 до 59. Для 125 проверка $2 \cdot 60 + 5 = 125$ связывает обе функции. Полный файл ниже.

```
def seconds_left(seconds):
    """Вернуть остаток секунд; целое seconds в 0...3600.

    >>> seconds_left(0)
    0
    >>> seconds_left(59)
    59
    >>> seconds_left(60)
    0
    >>> seconds_left(125)
    5
    >>> seconds_left(3600)
    0
    """
    return seconds % 60

if __name__ == "__main__":
    import doctest
    doctest.testmod(verbose=True)
```

Обсуждение: возможный вывод

Примеры дополняют, но не заменяют контракт. Ошибочный код и скопированное с него ожидание могут согласованно ошибаться. Ожидания получают из смысла задачи независимо: целое число минут и остаток восстанавливают исходные секунды.

Индивидуальная проверка: ответ и критерии

Команда `>>> seconds_left(121)`, следующая строка `1; full_minutes(121)=2`; два конкретных примера не доказывают все случаи. 4 балла.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Python 3.11+ и IDLE или редактор; запуск `python lesson.py` (либо `python3 lesson.py`). Внешние библиотеки не нужны. Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 8 класс. Предпосылки: `def`, `return`, целое деление, контракт функции. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. Матрица программы предназначена учителю.



Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ

07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Документирование программы. Docstring, PEP 257	Разбирает пример и создаёт две подпрограммы	Исходники, прогнозы и протокол проверки
Составление и отладка программ	Сравнивает фактический результат с контрактом	Объяснение границ и самостоятельный ответ

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

Python: модуль doctest

<https://docs.python.org/3/library/doctest.html>

PEP 257: соглашения о docstring

<https://peps.python.org/pep-0257/>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 18.10.2026.