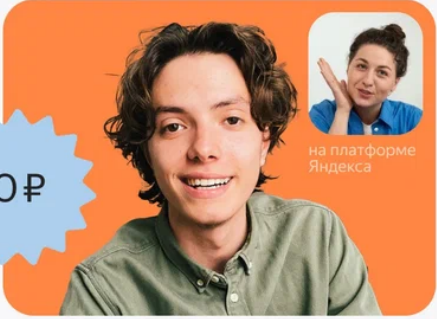


ХОЧУ ПОМОЧЬ

Измерение производительности небольших фрагментов кода

0₽




и масштабируйте бизнес легко

2 000 ₽

Виртуальная АТС Расширенная

MANGO OFFICE



Виртуальная АТС от лидеров рынка

900 ₽

Виртуальная АТС Базовая

practicum.yandex.ru

РЕКЛАМА · 18+ Я

Бесплатное занятие английским в Яндекс Практикуме

Полноценное занятие с преподавателем, а не презентация курсов

Узнать больше

Измерение производительности небольших фрагментов кода

стой способ измерения времени выполнения (производительности) маленьких кусочков

[й строки](#), так и программный интерфейс. Это позволяет избежать ряда распространенных выполнения.

модуля timeit:

ной строки для сравнения производительности трех различных выражений.

```
str(n) for n in range(100))'
sec per loop
[str(n) for n in range(100)])'
sec per loop
map(str, range(100)))'
sec per loop
```

То же самое, только достигнуто с помощью интерфейса Python:

```
>>> import timeit
>>> timeit.timeit('"-".join(str(n) for n in range(100))', number=10000)
0.3018611848820001
>>> timeit.timeit('"-".join([str(n) for n in range(100)])', number=10000)
0.2727368790656328
>>> timeit.timeit('"-".join(map(str, range(100)))', number=10000)
0.23702679807320237
```

Вызываемые объекты ([функции](#), [экземпляры классов](#) и т.д.) также могут быть переданы из интерфейса Python.

```
>>> timeit.timeit(lambda: "-".join(map(str, range(100))), number=10000)
0.19665591977536678
```

Обратите внимание, что [функция timeit.timeit\(\)](#) автоматически определяет количество повторений только при использовании интерфейса командной строки.

Можно предоставить оператор настройки `-s`, который выполняется только один раз в начале.

Командная строка:

```
$ python -m timeit -s 'text = "sample string"; char = "g"' 'char in text'
5000000 loops, best of 5: 0.0877 usec per loop
$ python -m timeit -s 'text = "sample string"; char = "g"' 'text.find(char)'
1000000 loops, best of 5: 0.342 usec per loop
```

То же самое, только программно:

```
>>> import timeit
>>> timeit.timeit('char in text', setup='text = "sample string"; char = "g"')
0.41440500499993504
>>> t.timeit('text.find(char)', setup='text = "sample string"; char = "g"')
1.520006203
```

Вверх

https://docs-python.ru/standart-library/modul-timeit-python/

1/3

То же самое можно сделать с помощью [класса `timeit.Timer\(\)`](#) и его методов:

```
>>> import timeit
>>> t = timeit.Timer('char in text', setup='text = "sample string"; char = "g"')
```



2 000 ₺

Виртуальная АТС Расширенная



900 ₺

Виртуальная АТС Базовая

как синхронизировать выражения, содержащие несколько строк. Здесь мы сравниваем `hasattr()` и `try/except` кроме проверки на отсутствие и представление атрибутов

```
tr.__bool__ 'except AttributeError:' ' pass'
sec per loop
(str, "__bool__"): pass'
sec per loop
```

```
nt.__bool__' 'except AttributeError:' ' pass'
usec per loop
(int, "__bool__"): pass'
usec per loop
```

```
>>> import timeit
>>> # attribute is missing
>>> s = """\
... try:
...     str.__bool__
... except AttributeError:
...     pass
... """
>>> timeit.timeit(stmt=s, number=100000)
0.9138244460009446
>>> s = "if hasattr(str, '__bool__'): pass"
>>> timeit.timeit(stmt=s, number=100000)
0.5829014980008651
>>>
>>> # attribute is present
>>> s = """\
... try:
...     int.__bool__
... except AttributeError:
...     pass
... """
>>> timeit.timeit(stmt=s, number=100000)
0.04215312199994514
>>> s = "if hasattr(int, '__bool__'): pass"
>>> timeit.timeit(stmt=s, number=100000)
0.08588060699912603
```

Чтобы предоставить [модулю timeit](#) доступ к определенным функциям в коде, можно передать параметр настройки, который содержит [оператор импорта](#):

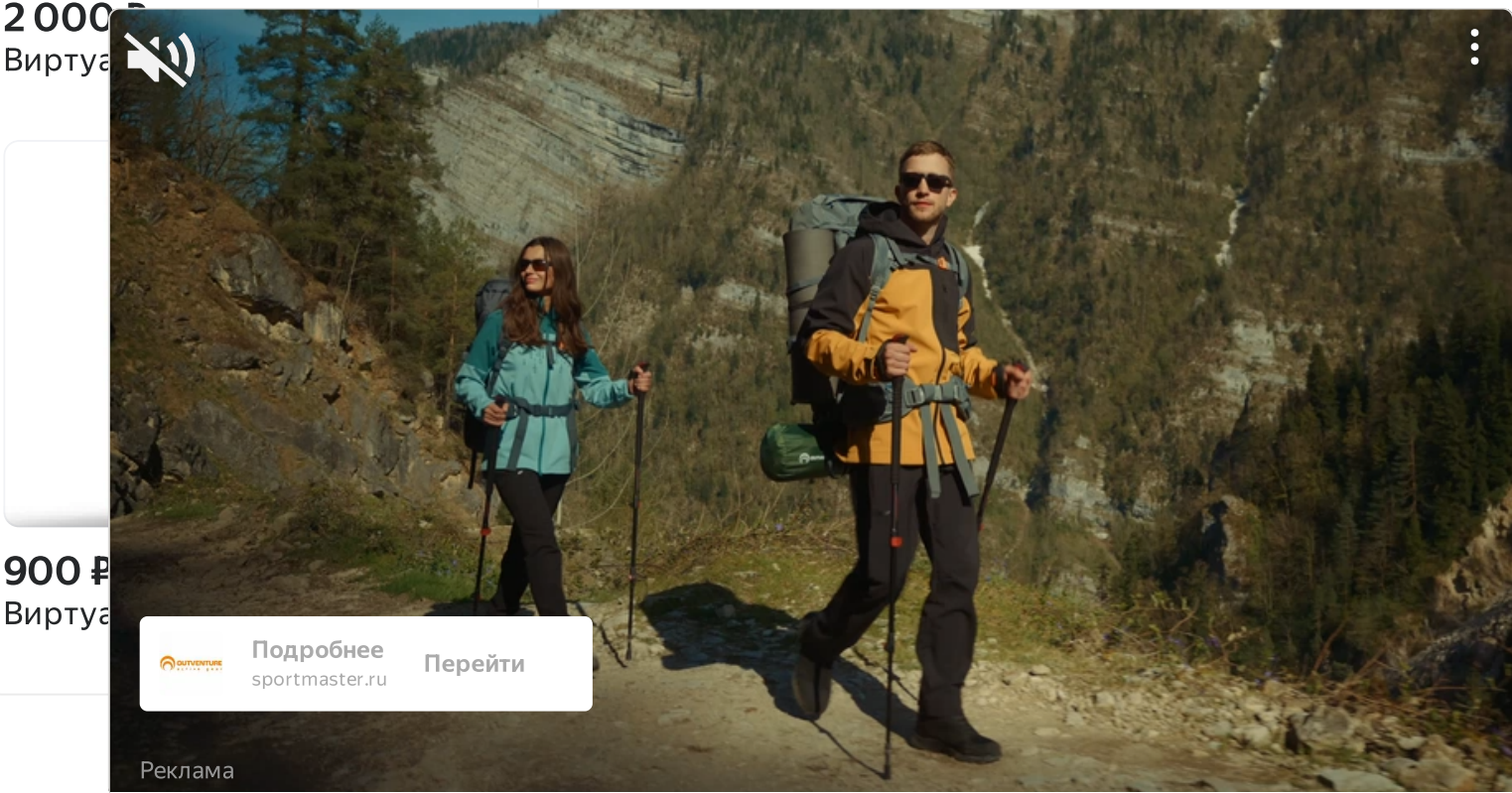
```
def test():
    """Stupid test function"""
    L = [i for i in range(100)]
    if __name__ == '__main__':
```

```
import timeit
print(timeit.timeit("test()", setup="from __main__ import test"))
```

Другой вариант - передать функцию `globals()` аргументу `globals`, что приведет к выполнению кода в текущем глобальном пространстве имен. Это может быть удобнее, чем указание импорта:



```
for func in (f,g,h)]:', globals=globals()))
```



Содержание раздела:

- [КРАТКИЙ ОБЗОР МАТЕРИАЛА.](#)
- [Функция timeit\(\) модуля timeit](#)
- [Функция repeat\(\) модуля timeit](#)
- [Класс Timer\(\) модуля timeit](#)
- [Интерфейс командной строки модуля timeit](#)