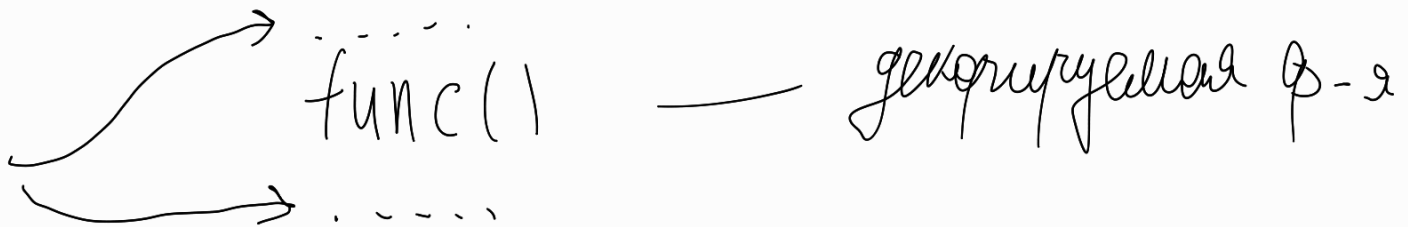


Decorators u. möglich.

```
def decorator(func): — p-2 gesprang  
    def decorated_func():
```

 `func()` — generiertes p-2

`return decorated_func` — p-2 / ergebnis

`func = decorator(func)`

`@decorator`
`def func():`
:
:

```
def time_decorator(func):  
    def decorated_func(*args, **kwargs):  
        import time  
        start_time = time.time()  
        res = func(*args, **kwargs)  
        print(time.time() - start_time)
```

return res
return decorated_func

закрепим
на замке в файле

Можно ли декорирование обернуть
векать? Технически можно, но
не оправдывает усилий.

Декораторы замедляют работу функций
из-за лишних вызовов и пространств имен.

-
- ✓ import math as m - импорт всего модуля
 - ✓ from module import function - импорт отдельных функций (основное пр-во имен)
 - ✗ from module import * - импорт всего модуля
основное пр-во имен

Любой файл .py - воспринимается как
модуль. При подключении модуля, он выполняется,
это может быть нежелательным.

if __name__ == "__main__":



не будем запускать или импортировать
модуль.

```
import sys
```

```
sys.path.insert(1, "full module address")
```

Таким образом с модулем, в котором есть
__init__.py (он может быть пустым)

```
import package.module
```

```
from package import module
```

-- all -- = ["module"] - список модулей,
импортирующихся через *