



Программирование  
на Python

Презентация занятия

# Изучение возможностей и синтаксиса Python: Функции.

4 занятие



**инжинириум**<sup>®</sup>

МГТУ им. Н.Э. Баумана

2019

## Тема: Изучение возможностей и синтаксиса Python: Функции.

Инструкция `def` создает объект функции и связывает его с именем. В общем виде инструкция имеет следующий формат:

```
def <name>(arg1, arg2,... argN):  
    <statements>  
    ...  
    return <value>
```

# Определение функции

```
def min_sec(x): # Создать функцию и связать ее с именем  
    return x * 60 # Тело, выполняемое при вызове функции`
```

```
value = min_sec(30) # Вызов функции  
print(value) # Что выведет print?
```



## Тема: Изучение возможностей и синтаксиса Python: Функции.

```
x = 10
def my_func(a, b):
    print(x)
    z = 5
my_func(1,4)
print(z)
```

Вывод:  
10

**NameError: name 'z' is not defined**

```
x = 10
def my_func(a, b):
    x = x + 1
    print(x)
my_func(1,4)
```

**UnboundLocalError: local variable 'x'  
referenced before assignment**

```
# Если не нашел, то
ищет здесь
строка кода
строка кода
```

Блок 1

```
# Если не нашел,
то ищет здесь
строка кода
строка кода
```

Блок 2

```
#Ищет здесь
строка кода
строка кода
строка кода
```

Блок 3

**строка кода**  
**строка кода**



## Тема: Изучение возможностей и синтаксиса Python: Функции.

```
def degree(x, a = 2):  
    f = x**a  
    return f  
print(degree(5), degree(5, 3), degree(2, a = 5))  
# Что выведет программа?
```

Вывод:

25 125 32

```
print(degree(a = 5))
```

**TypeError: func() missing 1 required positional argument: 'a'**



## Тема: Изучение возможностей и синтаксиса Python: Функции.

```
def unknown(*args):  
    for argument in args:  
        print ( argument )
```

```
unknown('Что ', 'происходит', '?')  
unknown('Не знаю!')
```

Вывод программы:

Что

происходит

?

Не знаю!



## Тема: Изучение возможностей и синтаксиса Python: Функции.

Анонимные функции могут содержать лишь одно выражение, но и выполняются они быстрее. Анонимные функции создаются с помощью инструкции `lambda`.

```
func = lambda x, y: x + y  
func(1, 2) # 3  
func('a', 'b') # 'ab'
```

```
(lambda x, y: x + y)(1, 2) # 3  
(lambda x, y: x + y)('a', 'b') # 'ab'
```

`lambda` функции, в отличие от обычной, не требуется инструкция `return`, а в остальном, ведет себя точно так же.

