

**kwargs

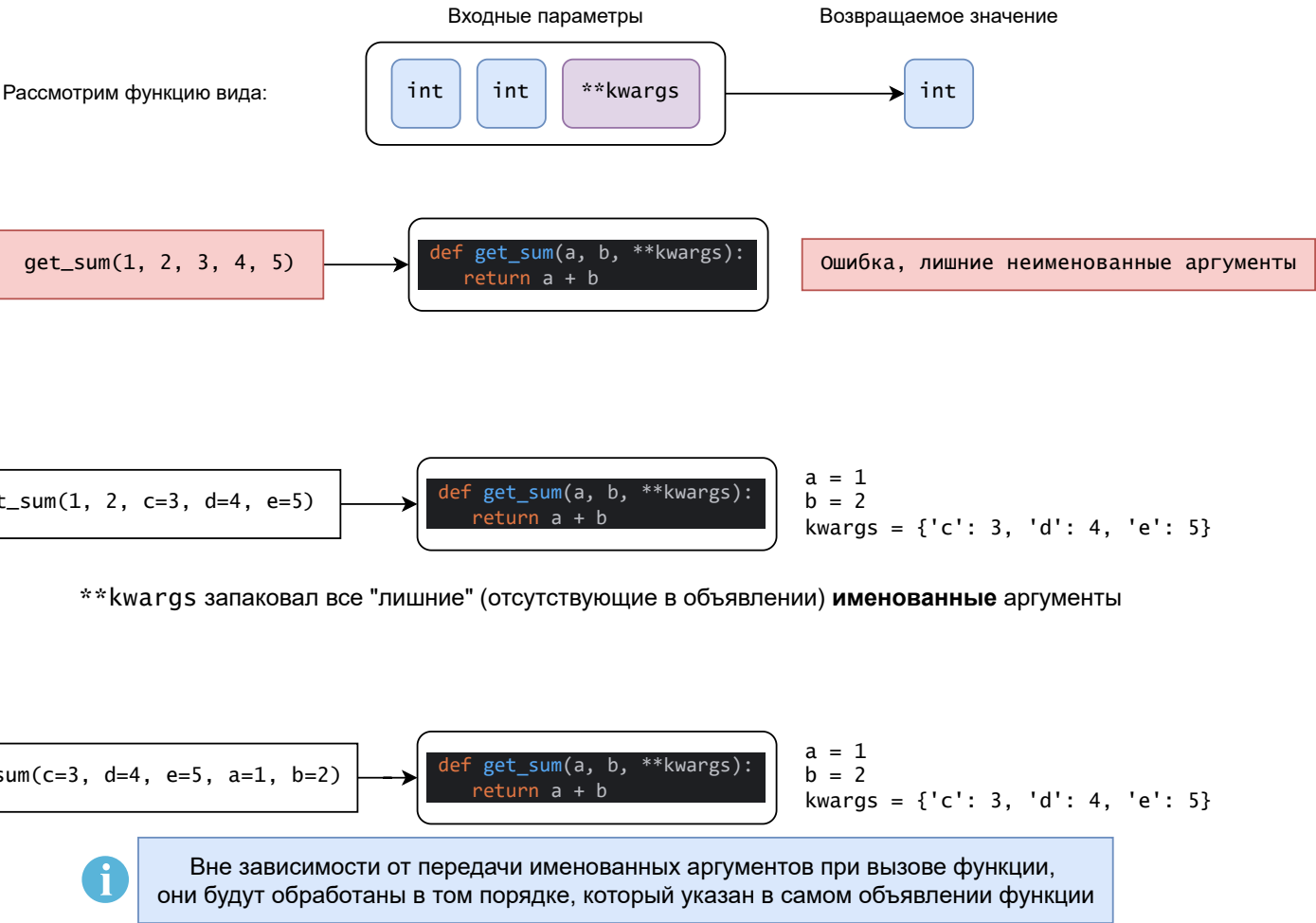
**kwargs запаковывает все оставшиеся ИМЕННОВАННЫЕ аргументы

Имя также может быть другим

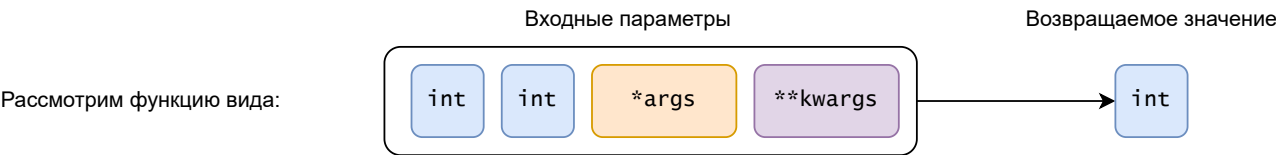
```
def get_sum(a, b, **kwargs):  
    return a + b
```

```
def get_sum(a, b, **other):  
    return a + b
```

i внутри функции kwargs это словарь



*args, **kwargs вместе



get_sum(1, 2, 3, 4, 5)

```
def get_sum(a, b, *args, **kwargs):  
    return a + b + sum(args) + sum(kwargs.values())
```

a = 1
b = 2
args = (3, 4, 5)
kwargs = {} # пустой

get_sum(1, 2)

```
def get_sum(a, b, *args, **kwargs):  
    return a + b + sum(args) + sum(kwargs.values())
```

a = 1
b = 2
args = tuple() # пустой
kwargs = {} # пустой

get_sum(1, 2, 3, tr=4, var=5)


```
def get_sum(a, b, *args, **kwargs):  
    return a + b + sum(args) + sum(kwargs.values())
```

a = 1
b = 2
args = (3,)
kwargs = {'tr': 4, 'var': 5}

get_sum(tr=4, a=1, var=5, b=2)

```
def get_sum(a, b, *args, **kwargs):  
    return a + b + sum(args) + sum(kwargs.values())
```

a = 1
b = 2
args = tuple() # пустой
kwargs = {'tr': 4, 'var': 5}



Таким образом *args, **kwargs **вместе** дают возможность передавать в функцию **бесконечное кол-во аргументов**

get_sum(10, 20, tr=4, a=1, var=5, b=2)

позиционный

именованный

```
def get_sum(a, b, *args, **kwargs):  
    return a + b + sum(args) + sum(kwargs.values())
```

Конфликт

a = ?
b = ?

Ошибка, дважды переданы аргументы a и b