О функции min

В питоне есть много удобных функций, которые решают всякие мелкие задачки. Одна из таких функций - `min()`. Она находит минимальный элемент в списке. Например, ниже показан поиск наименьшего числа и поиск самой короткой строки (если коротких много — выберется первая по алфавитному порядку):

```
numbers = [4, 1, 2, 5]
min(numbers)  # 1

words = ['abcd' 'ab', 'a', 'b', 'abcde']
min(words)  # 'a'
```

Самое главное, что эта функция умеет искать минимальное значение среди любых типов данных, доступных в питоне. Давайте найдём автора с наименьшим количеством постов. Есть список авторов блога: словари с именем автора и количеством его постов:

```
users = [
  {'username': 'Даша', 'posts': 25},
  {'username': 'Валера', 'posts': 10},
  {'username': 'Ибрагим', 'posts': 56},
]
```

Здесь подошла бы функция `min()`, но Python не умеет сравнивать словари напрямую. Нужно сказать ему, что сравнивать нужно по значению ключа `'posts'`.

К счастью, создатели языка предвидели такой случай и ввели опциональный параметр `key`. Он принимает на вход функцию, которая для каждого автора вернёт количество его постов. `min` поймёт, что сравнивать нужно по ним:

```
def get_user_posts(user):
    return user['posts']

min(users, key=get_user_posts)  # Вернёт {'username': 'Валера', 'posts': 10}
```

Функция `get_user_posts`, получает на вход пользователя `user`, достаёт количество его постов и возвращает наружу `return user['posts']`. Затем, мы передаём эту функцию в `min()`. Обратите внимание, что `get_user_posts` **передаётся без скобок**, подобно коллбэку! Функция `min` будет вызывать её самостоятельно, во время поиска.