\n - перенос строки

\t - табуляция

—————————————————

{ } без значений создают словарь, а со значениями, в зависимости от синтаксиса могут создавать как множество, так и словарь:

a = {}

print(type(a)) # <class 'dict'>

b = {1, 2, 3}

print(type(b)) # <class 'set'>

c = {'a': 1, 'b': 2}

print(type(c)) # <class 'dict'>

————————————————————————-

Метод title() возвращает строку с заглавной первой буквой Каждого Слова В Строек В Верхний Регистр.

Источник: <https://pythonstart.ru/string/title-python>

Метод [str.capitalize()](https://docs-python.ru/tutorial/operatsii-tekstovymi-strokami-str-python/metod-str-capitalize/) вернет копию строки str с первым символом в верхнем регистре, а остальные символы будут в нижнем регистре.

<https://replit.com/@LiubovSaltykova> - good

Я правильно понял:

что функция возвращает return None, если внутри ее только print ( ) и не создаются новые переменные или значения.

Если print внутри функции выводит итерируемый список, то return только возвращает одну строку, либо первую, либо вторую, в зависимости от отступа return. Чтобы после списка не выдавало сообщение None реализовал так:

if get\_list\_all\_documents(documents) == None:

print("\n" )

А список не экран выводится из самой функции.

*def* get\_list\_all\_documents(*documents*):

    for document in documents:

        all\_list\_number\_name = ('{} "{}" "{}"'.format(document['type'], document['number'], document['name']))

        print(all\_list\_number\_name)

Обратил внимание, что если добавлять новый документ и прописывать так:

документ: **q** , №: **q** , владелец: **q** , № полки: **q** ,

документ: **q** , №: **q** , владелец: **q** , № полки: **1**,

документ: **q** , №: **q** , владелец: **1** , № полки: **1**,

документ: **q** , №: **1** , владелец: **1** , № полки: **1**,

документ: **q** , №: **1** , владелец: **1** , № полки: **1**,

документ: **1** , №: **q** , владелец: **1** , № полки: **1**,

То при удалении документа с выводится сообщение: ValueError: list.remove(x): x not in list

Если так:

документ: **q**, №: **1**, владелец: **q**, № полки: **1**

документ: **q**, №: **1**, владелец: **q**, № полки: **q**

документ: **11**, №: **qq**, владелец: **11**, № полки: **11**,

– то все работает, документ удаляется, почему так происходит, и от чего зависит

## **Таблица "методы списков"**

|  |  |
| --- | --- |
| **Метод** | **Что делает** |
| **list.append**(x) | Добавляет элемент в конец списка |
| **list.extend**(L) | Расширяет список list, добавляя в конец все элементы списка L |
| **list.insert**(i, x) | Вставляет на i-ый элемент значение x |
| **list.remove**(x) | Удаляет первый элемент в списке, имеющий значение x. ValueError, если такого элемента не существует |
| **list.pop**([i]) | Удаляет i-ый элемент и возвращает его. Если индекс не указан, удаляется последний элемент |
| **list.index**(x, [start [, end]]) | Возвращает положение первого элемента со значением x (при этом поиск ведется от start до end) |
| **list.count**(x) | Возвращает количество элементов со значением x |
| **list.sort**([key=функция]) | Сортирует список на основе функции |
| **list.reverse**() | Разворачивает список |
| **list.copy**() | Поверхностная копия списка |
| **list.clear**() | Очищает список |

Нужно отметить, что методы списков, в отличие от [строковых методов](https://pythonworld.ru/tipy-dannyx-v-python/stroki-funkcii-i-metody-strok.html), изменяют сам список, а потому результат выполнения не нужно записывать в эту переменную.

пример —————————————————————————————

number1 = [2, 10, 11, 25, 80, 100]

**for** i **in** number1:

    # print(i)

**if** **len**(**str**(i)) != 2:

**print**(i)

        number1.remove(i)

**print**(number1)

>>>[10, 11, 25, 80] # удаляет все где в списке < 2 и > 3 литералов, оставляет только 2х значные

number = [1, 11, 25, 80, 100]

**for** i **in** **reversed**(number):

**if** **len**(**str**(i)) == 2:

        number.remove(i)

**print**(number)

>>>[1, 100] # удаляет все где в списке > 2х значных литералов

——————————————————————————- пример

## **Методы словарей**

**dict.clear**() - очищает словарь.

**dict.copy**() - возвращает копию словаря.

classmethod **dict.fromkeys**(seq[, value]) - создает словарь с ключами из seq и значением value (по умолчанию None).

**dict.get**(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а возвращает default (по умолчанию None).

**dict.items**() - возвращает пары (ключ, значение).

**dict.keys**() - возвращает ключи в словаре.

**dict.pop**(key[, default]) - удаляет ключ и возвращает значение. Если ключа нет, возвращает default (по умолчанию бросает исключение).

**dict.popitem**() - удаляет и возвращает пару (ключ, значение). Если словарь пуст, бросает исключение KeyError. Помните, что словари неупорядочены.

**dict.setdefault**(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а создает ключ со значением default (по умолчанию None).

**dict.update**([other]) - обновляет словарь, добавляя пары (ключ, значение) из other. Существующие ключи перезаписываются. Возвращает None (не новый словарь!).

**dict.values**() - возвращает значения в словаре.

Лямбда-функция может иметь только одно выражение.