Функция map() в Python

**ОПУБЛИКОВАНО**[**13.05.2021**](https://pythonist.ru/funkcziya-map-v-python/)[**ФУНКЦИИ**](https://pythonist.ru/category/voprosy-s-sobesedovaniya-python/funkczii/)

Если вы учитесь программировать, то функция map() в Python — это ваша возможность повысить свой уровень и упростить решение некоторых задач.

Представьте себе: вы хотите стать более эффективным разработчиком. Вы хотите, чтобы ваш код компилировался быстрее. Вы хотите произвести впечатление на коллег своими глубокими знаниями в области программирования. Если что-то из этого находит отклик у вас в душе, значит, вы попали в нужное место.

Прежде чем двигаться дальше, давайте разберемся, что такое функции и итерируемые объекты. Функции — это код, который выполняет определенную задачу. Например: len(), print(), str(). Итерируемые объекты — это объекты, содержащие один или несколько членов. К итерируемым объектам относят списки, кортежи, словари и др.

Функция map() в Python — это функция, которая позволяет вам преобразовывать весь итерируемый объект с помощью другой функции. Ключевой концепцией здесь являются преобразования, которые могут включать:

* преобразование строк в числа
* округление чисел
* получение длины каждого повторяемого члена
* и пр.

На данном этапе может возникнуть вопрос: а почему нельзя просто сделать это с помощью цикла for? Ответ прост: можно, но использование функции map() сэкономит память (а это значит, что ваш код будет работать быстрее). Более того, решение с использованием map() делает ваш код более компактным.

Давайте рассмотрим пример, чтобы вы могли понять, что мы имеем в виду. Допустим, у вас есть список строк list\_of\_strings = [«5», «6», «7», «8», «9», «10»], которые на самом деле являются числами. Нужно преобразовать этот список в список целых чисел. Для этого можно использовать пустой список и цикл for:

list\_of\_strings = ["5","6","7","8","9", "10"]

result = []

**for** string **in** list\_of\_strings:

result.append(int(string))

print(result)

Если вы запустите этот пример, вы получите: [5, 6, 7, 8, 9, 10].

**Что при использовании цикла for происходит под капотом?**

Мы можем быть довольны результатом преобразования списка строк в список чисел, но давайте подумаем о том, что только что сделал наш код.

Мы сказали компьютеру пройтись по каждому члену («5», «6», «7» и т.д.), преобразовать член, а затем сохранить его в новом списке. Хотя использование цикла for для преобразования списка является функциональным, это не оптимальное решение.

**Функция map() в Python**

Чтобы получить такой же рабочий, но при этом более оптимальный результат, давайте используем функцию map().

Будем работать с тем же списком — list\_of\_strings = [«5», «6», «7», «8», «9», «10»]. А для преобразования списка строк в список целых чисел воспользуемся функцией Python map():

result = map(int,list\_of\_strings)

print(list(result))

Если вы запустите этот пример, вы получите результат, аналогичный полученному при помощи цикла for.

Прежде чем мы выясним, почему функция map() в Python более оптимальна, чем использование цикла for, давайте разберемся с тем, что мы только что сделали.

По сути, мы лишь создали переменную, хранящую список строк, которые мы хотим преобразовать в числа.

result = map(int, list\_of\_strings)

Давайте разберем, как работает приведенный выше код. Синтаксис функции map() в Python следующий:

map (функция, итерируемый объект)

map() — это просто имя функции, здесь ничего особенного.

Первым аргументом функция map() принимает другую функцию. В приведенном выше примере кода мы использовали функцию int. Мы могли бы использовать другую встроенную функцию, скажем, len(), или создать нашу собственную функцию и использовать ее здесь.

Вторым аргументом для map() идет сам итерируемый объект. В нашем примере это список list\_of\_strings. Наш преобразованный список мы запишем в переменную result.

Перейдем к последней строке кода. Опять же, будем работать изнутри:

print(list(result))

list() принимает наши недавно преобразованные итерируемые элементы и сообщает компьютеру, что эти элементы являются частью списка. print() выводит наш новый список.

**Что при использовании функции map() происходит под капотом?**

Вместо перебора каждого члена списка строк функция map() преобразовала весь список строк в список чисел. Вы сэкономили память, и ваш код сработал быстрее. Поставленная задача выполнена наиболее оптимальным способом.

**Заключение**

Наконец, функция map() более элегантна, чем цикл for. Она помогает быстрее скомпилировать код. Также использование функции map() поможет вывести ваши навыки программирования на новый уровень и стать более «подкованным» программистом.

Однако не стоит забывать что, функция map() в Python – это только начало. В этом языке есть еще множество различных трюков, которые помогут вам писать более элегантный код и эффективнее решать поставленные задачи.