

# Python Starter

Функции

# Python Starter

После урока обязательно



Повторите этот урок в видео формате на [ITVDN.com](http://itvdn.com)

Доступ можно получить через руководство вашего учебного центра



Проверьте как Вы усвоили данный материал на [TestProvider.com](http://testprovider.com)

# Python Starter

Введение в Python

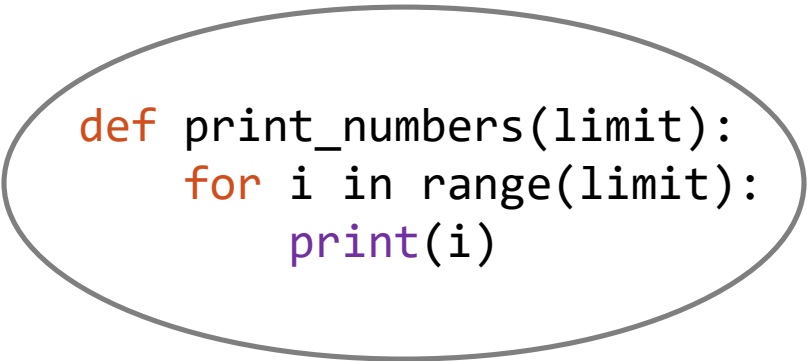
Функции

# Python Starter

## Понятие функции

*Функция* — это именованный участок кода, к которому можно неоднократно обращаться из других мест программы.

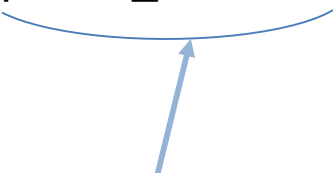
Функции могут принимать параметры (аргументы) и возвращать значения.



```
def print_numbers(limit):  
    for i in range(limit):  
        print(i)
```

объявление  
функции

```
n = int(input())  
print_numbers(n)
```



вызов  
функции

# Python Starter


## Передача параметров

Функции могут принимать параметры (аргументы).

*Формальные параметры* – параметры, указываемые при объявлении функции.

*Фактические параметры* – параметры, которые передаются в функцию при вызове.


формальный  
параметр



```
def print_numbers(limit):  
    for i in range(limit):  
        print(i)
```

```
n = int(input())  
print_numbers(n)
```

фактический  
параметр



# Python Starter

## Возврат результата

В некоторых языках есть чёткое разделение между подпрограммами, возвращающими результат (*функциями*), и теми, которые никакого результата не возвращают (*процедурами*).

В Python любая подпрограмма является функцией. Если функция явно не возвращает никакого значения, автоматически возвращается **None**.

Для возврата значения используется оператор **return** *значение*. Работа функции при этом завершается. Если не указать значение, которое нужно вернуть, автоматически возвращается None и это можно использовать для досрочного выхода из процедур.

```
def add_numbers(a, b):  
    return a + b
```

# Python Starter

## Создание функции

1. Следует выбрать подходящее имя для функции, продумать, какие параметры должна принимать функция (от каких значений она зависит) и что должна возвращать.
2. Методы определяются при помощи ключевого слова **def**, за которым следует *сигнатура функции* – её имя и список формальных параметров.
3. Имена параметров функции перечисляются в круглых скобках через запятую. Если функция не принимает никаких параметров, пустые круглые скобки всё равно указываются.
4. Если функция должна что-то возвращать, это делается при помощи ключевого слова **return**. Если оно не присутствует в функции или хотя бы одной возможной её ветви исполнения, функция автоматически вернёт **None**.

# Python Starter

## Вызов функции

1. Функция вызывается путём указания её имени и списка фактических параметров, разделённых запятыми, в круглых скобках. **Важно:** если функция не требует передачи её аргументов, всё равно указываются пустые круглые скобки, так как именно они являются признаком *вызова функции*. Если этого не сделать, то мы получим объект – *саму функцию*.
2. Вызов функции – выражение. Его значение совпадает со значением, которая вернула функция, и его можно присвоить переменной, использовать в качестве аргумента другой функции или использовать внутри иного выражения.
3. Если функция ничего не возвращает или её результат нам не интересен, можно ничему не присваивать её результат и использовать вызов этой функции как главный оператор в данной строке кода.



# Python Starter

## Вызов функции с именованными параметрами

- При вызове функции фактические параметры замещают формальные в том порядке, в котором они указаны.
- Можно изменить этот порядок, указав при вызове имена соответствующих формальных параметров:

`имя_функции(аргумент_2=значение, аргумент_1=значение)`

- При вызове функции можно использовать именованные и неименованные аргументы одновременно. В таком случае сначала указываются неименованные фактические параметры, которые замещают формальные в том порядке, в котором в заголовке функции описано соответствующие формальные параметров, а затем указываются остальные фактические параметры вместе с именами соответствующих формальных параметров в произвольном порядке.

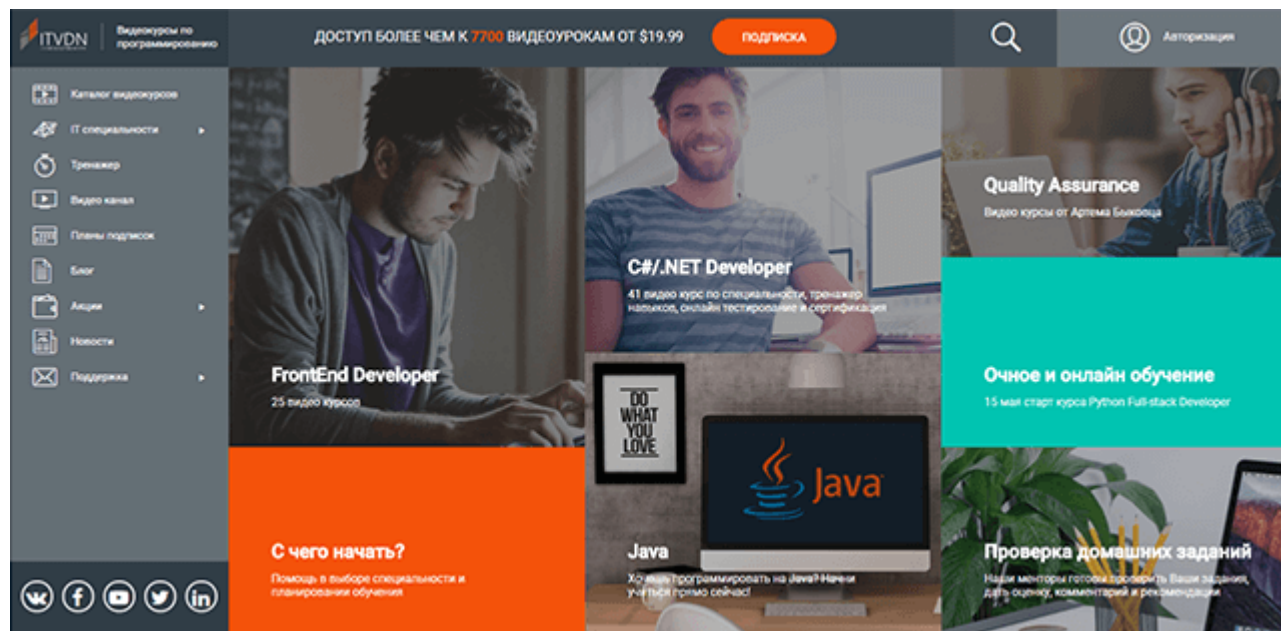
# Python Starter

## Опциональные параметры

- Можно часть параметров функции сделать необязательными для передачи при вызове.
- Для этого необходимо указать значения этих параметров по умолчанию, которые будут использованы в том случае, если при вызове соответствующие фактические параметры не указаны.
- Значения по умолчанию создаются только один раз при создании функции и связываются с именами соответствующих формальных параметров. Поэтому нужно быть осторожным при использовании значений по умолчанию, которые являются изменяемыми объектами, так как их можно изменить для всех последующих вызовов функции. С неизменяемыми объектами, такими как числа и строки, этой проблемы нет.

# Смотрите наши уроки в видео формате

ITVDN.com



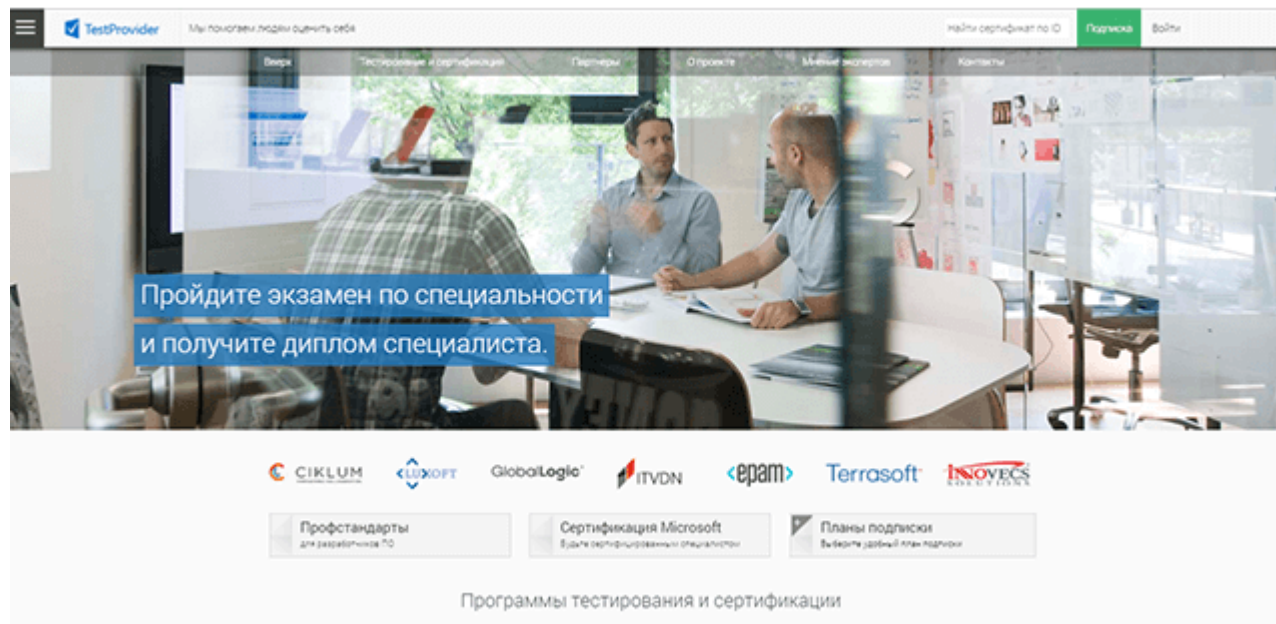
Посмотрите этот урок в видео формате на образовательном портале [ITVDN.com](http://ITVDN.com) для закрепления пройденного материала.

Курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics и другими высококвалифицированными разработчиками.



# Проверка знаний

TestProvider.com



TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT специалиста.

После каждого урока проходите тестирование для проверки знаний на [TestProvider.com](https://testprovider.com)

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.



# Python Starter

Q&A

# Информационный видеосервис для разработчиков программного обеспечения

