Functions

Defining and Using Functions

SoftUni Team

Technical Trainers

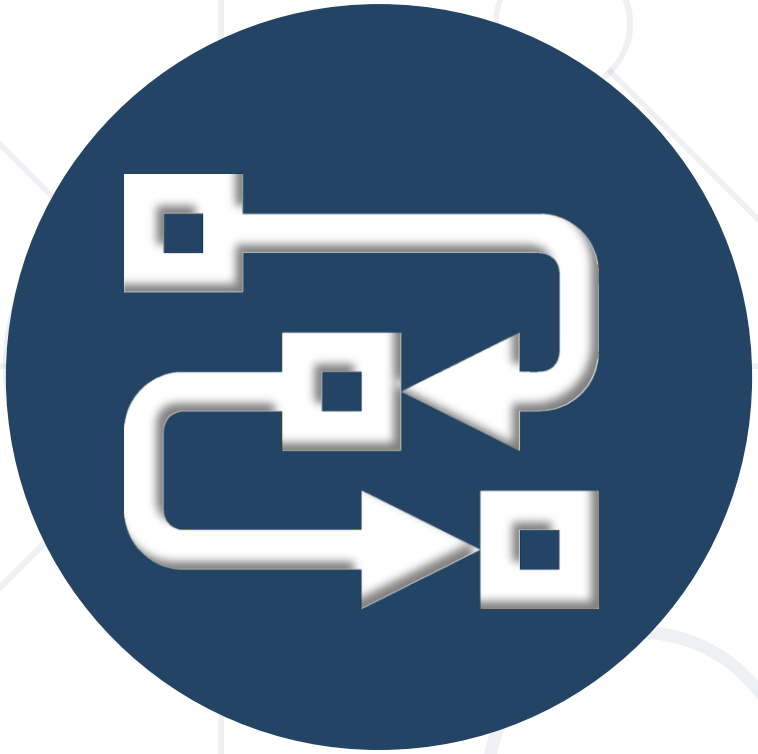
Software University

SoftUni

Software University

https://softuni.bg

Table of Contents



Functions Overview

Declaring and Invoking Functions

Functions

- Function == named piece of code
  - Can take parameters and return result

Use snake_case

Function parameter

```
def function_name(parameter: type):
    statement(s)
```

Type of the parameter

Why Use Functions?

- More **manageable programming**
  - Splits large problems into small pieces
  - Better organization of the program
  - Improves code readability
  - Improves code understandability
- Avoiding **repeating code**
  - Improves code maintainability
- Code **reusability**
  - Using existing functions several times

Built-In Functions

- Python has a set of **built-in functions** that we can call at **any time**
- List of some built-in functions

```
abs()
min()
max()
round()
```

```
sum()
filter()
map()
sorted()
```

Declaring and Invoking Functions

Declaring Function

Software University

Function Name

Parameters

```
def print_text(text):
    print(text)
```

Function Body

- Using the **def** statement is the most common way to define a function in Python
- Functions can have **several parameters**
- It is possible for the function to **not** return a value

Invoking a Function

- Functions are **first** declared, then **invoked** (many times)

```
def print_header():
    print("This is header")
```

Function Declaration

- Functions can be **invoked** (called) by their name

```
print_header()
```

Function Invocation

Invoking a Function

- A function can be invoked from:
  - Other functions

```
def print_header():
    print_header_top()
    print_header_bottom()
```

Function invoking functions

  - Itself (recursion)

```
def crash():
    crash()
```

Function invoking itself

Function without Parameters

- Executes the code after
- Does not return result

```
def multiply_numbers():
    result = 5 * 5
    print(result)
multiply_numbers()  #25
```

Prints result on the console

Return Values

The Return Keyword

return Keyword

Software University

- Functions can return a value that you can use directly:

```
def give_me_five():
    return 5
print(give_me_five())  # Print the returned value
#Out: 5
```

- or save the value for later use:

```
num = give_me_five()
print(num) #Print the saved returned value
#Out: 5
```

return Keyword

Software University

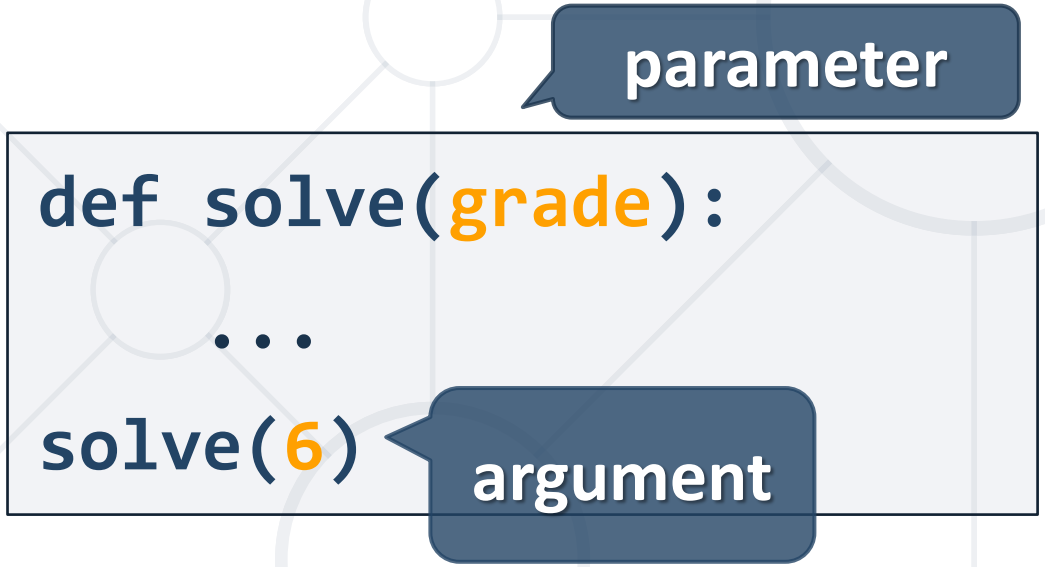
- If **return** is encountered in the function the function will be exited immediately

```
def give_me_another_five():
    return 5
    print('This statement will not be printed.')
print(give_me_another_five()) #Out: 5
```

Parameters vs Arguments

Parameters vs Arguments

- **Parameter** is a variable defined in a function definition, while the **argument** is an actual value passed to the function



Default Arguments

- Function arguments can have **default** values
- If the function is called **without the argument**, the argument gets its default value

```
def person(first_name = 'George', last_name ='Brown'):
    print(first_name, last_name)
person('Peter') #'Peter Brown'
```

Keyword (Named) Arguments

Software University

- Functions can be called using **keyword arguments**
- When we use keyword/named arguments, it's the **name** that matters, not the **position**

```
def area(width, height):
    return width * height
print(area(height = 2, width = 1)
```

Lambda Functions

Lambda Definition

- Lambda is an **anonymous one-time** function
  - Like a function, it can take a parameter and return a result

key word | arguments | expression

```
x = lambda a: a + 10
print(x(5))  # 15
```

Lambda Example

- It can take multiple parameters

```
x = lambda a, b: a * b
print(x(3, 4))  # 12
```

```
full_name = lambda first, last: f'I am {first} {last}'
result = full_name('Guido', 'van Rossum')
print(result)  # I am Guido van Rossum
```

Summary

- Break large programs into simple functions that solve small sub-problems
- Consist of declaration and body
- Are invoked by their name
- Can accept parameters

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, softuni.org
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos, and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://softuni.org
- © Software University – https://softuni.bg