Introduction to programming

Lecture 06

Table of contents

01

FUNCTIONS

Functions - the basics

```
# A function is a block of code which only
runs when it's called
# Functions can take data in the form of
arguments
# Functions can return data
# In python a function is defined using the
def keyword

def function_name(argument_identifier,
other_argument_identifier):
    # body
    returned_value = 10
    return returned_value


# call the function
function_name(20, 30)
```

Functions - arguments

```
# Arguments have a name, and a type

def greet(name):
    print("Welcome: " + name)


greet("pesho")
greet("gosho")
greet("ivan")
```

Functions – multi arguments

```
# By default, the functions should be called with the correct number of arguments
# Multiple arguments are separated by a comma (arg1, arg2)

def greet_by_full_name(first_name, last_name):
    print("Welcome: " + first_name + " " + last_name)


greet_by_full_name("Ivan", "Dimitrov")
```

Functions - default arguments

```
# If you call a function without an argument
it will use the default value

def repeat(string_to_repeat, times=1):
    for _ in range(times):
        print(string_to_repeat)


repeat("pesho")
repeat("gosho", 3)
```

Functions - return value

```
# All functions in python return a value. To
let the function return a value use thereturn
keyword.

def addition(x, y):
    return x + y


print("5 + 10 = ", addition(5, 10))
```

Functions - return value

```
# You can return the value of a variable
defined inside the function

def addition(x, y):
    result = x + y
    return result


print("5 + 10 = ", addition(5, 10))
```

Functions - return value

```
# By default, if you don't have a return
statement the function will return None

def no_return_example(x, y):
    result = x * y


print("Is result equal to None?: ",
no_return_example(10, 20) is None)
```

Functions - calls

```
# A lot of the times we'll define a bunch of
functions that are going to be chained and
used all together.

def my_addition(x, y):
    return x + y


def my_sum(numbers):
    s = 0
    for number in numbers:
        s = my_addition(s, number)

    return s
```

Functions - Keyword arguments and type hinting

```
def repeat(string_to_repeat: str, times: int) -> str:
    return string_to_repeat * times


print(repeat(times=10, string_to_repeat="*"))
```


EXERCISE
You can find them in the github repository
https://github.com/genchev99/ia-lectures

THANKS !

CREDITS: This presentation template was created by Slidesgo, incluiding icons by Flaticon, and infographics & images by Freepik.