Developing Python Apps

Part 1: Basics C

Kauko Kolehmainen

2020

Contents

[Functions 2](#_Toc42326403)

[Introduction to functions 2](#_Toc42326404)

[Function definition (implementation) 3](#_Toc42326405)

[Learning by Examples 3](#_Toc42326406)

[Passing by value or passing by reference 7](#_Toc42326407)

# Functions

## Introduction to functions

Functions are often called also subprograms, routines, procedures, methods…

Functions   
do one well defined task

Instead of putting all the code the main body of the program, we can use  
functions and call them when needed.

Why functions?

Can be called several times from other parts of the program  
Can be reused in other programs  
Program is better organized (better structure)  
No need for repeating same code  
SO, When some code is to be used more than once, it is good to create a function

## Function definition (implementation)

def functionName(parameters):

function body (the code, implementation)

## Learning by Examples

Example 1

Our function prints out “Good Morning”

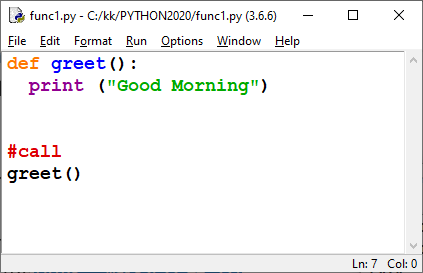
def greet():

print (“Good Morning”)

Function call

greet()

Test run



Result



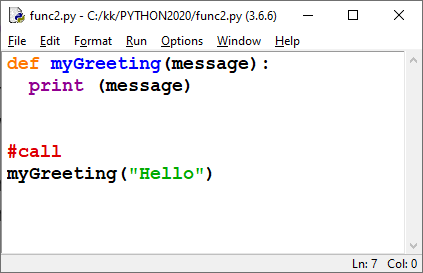
Example 2

We want do decide ourselves what to print!

def myGreeting(message):

print (message)

Test run



Result:



Example 3  
We have had only 1 parameter – let's try with different kinds of function parameters now...

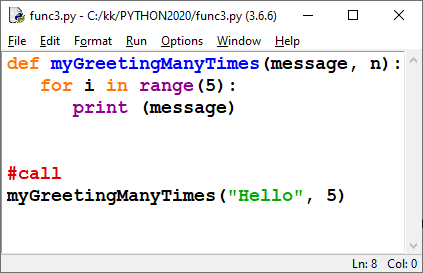
Example 4  
Our function prints our greeting n times.

def myGreetingManyTimes(message, n):

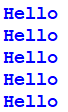
for i in range(5):

print (message)

Test run



Result



Functions can also return values. Let's take a look at that feature.

Example 5  
Next function returns the sum of 3 whole numbers.

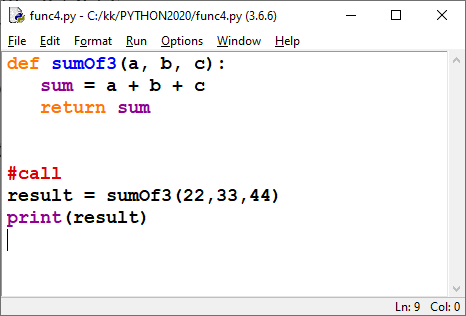
def sumOf3(a, b, c):

sum = a + b + c

return sum

Note, we have return statement there!

Test run



Result



We can call that function also like this:

print (sumOf3(55,66,88))

OR

x = 10  
y = 20  
z = 20

print (sumOf3(x, y, z))

Example 6

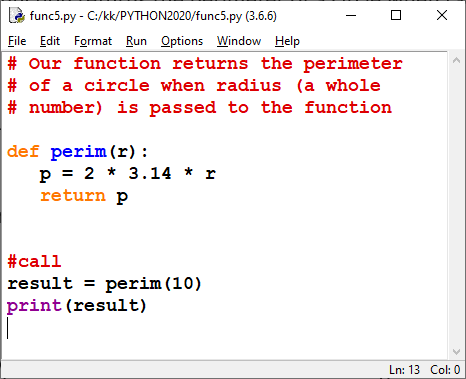
Our function returns the perimeter of a circle when radius (a whole number) is passed to the function

def perim(r):

p = 2 \* 3.14 \* r

return p

Test run



Result



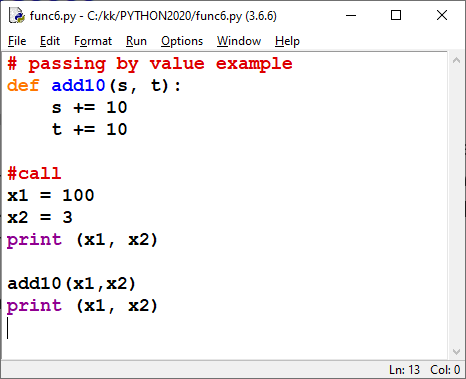
Note: we could have taken the value of pi from math, but libraries are discussed later...

## Passing by value or passing by reference

Here is an example where function has normal parameters. Values of are modified inside the function but

original variables are not changed. This is called passing by value: only the value of the variable is passed to the function and original variable cannot be modified by the function (function does not an access to original memory place).

Example 7



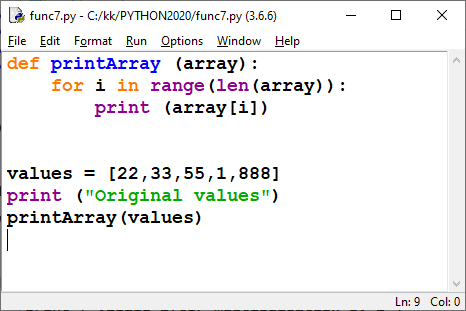
Test run shows that values of x1 and x2 are not changed:



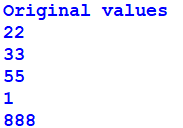
Arrays are passed to functions as references – they can be modified by the function.

Example 7

First we only print an array



Result



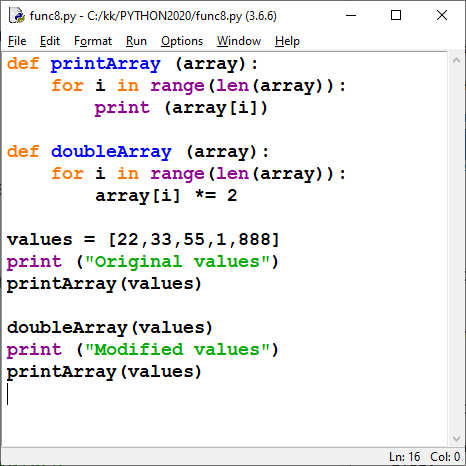
Example 8  
Here arrays values are multiplied by 2. You can see that original array has changed.

def doubleArray (array):

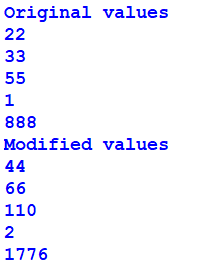
for i in range(len(array)):

array[i] \*= 2

Test run



AND



We return to functions even later.

This was part one...