Computational Programming with Python

Unit 4: Functions

Center for Mathematical Sciences, Lund University

Lecturer: Claus Führer, Alexandros Sopasakis
Functions in Mathematics

In Mathematics:
A function is written as a map, that uniquely assigns an element $y$ from the range $R$ to every element $x$ from the domain $D$.

$$f : x \mapsto y.$$ 

$f$ is the function
$x$ is its argument
$y$ is its (return) value

There can be several arguments of different type:

Consider

$$f(g, a, b) = \int_{a}^{b} g(x) \, dx$$

The arguments are not interchangable. Position matters.
Functions in Python

Definition of a function:

```python
def f(x1, x2, x3):
    .... # some computations
    y = ... #
    return y
```

Evaluation (call) of a function:

```python
f(17, 18, -2)
f([1, 2, 3], {'Tol': 1.e-10}, 'ro')
```

Wording:

- $x_1$, $x_2$, $x_3$ are called function parameters (needed for the definition)
- 17, 18, -2 are called arguments (needed for the evaluation)
Passing Arguments

Consider:

```python
def subtract(x1, x2):
    return x1 - x2
```

Passing arguments by position:

```python
subtract(1, 2)  # returns -1
```

Position matters.

Passing arguments by keyword:

```python
subtract(x2=2, x1=1)  # returns -1
```

Position doesn’t matter.
Scope of variables I

Variables defined inside the function are said to belong to the function’s scope. They are unknown outside the function.

Definition:

```python
def mult2(x):
c = 2.
return c*x
```

Evaluation:

```python
mult2(20.) # returns 40
c      # returns NameError: 'c' not defined
```
Scope of variables II

Compare:

a is a parameter of the function

```python
def multiply(x, a):
    return a*x
```

a is referenced from outside the function’s scope:

```python
a = 3
def multiply(x):
    return a*x
```
Local variables

Similarly, immutable variables are copied when calling a function. You cannot change an input variable by changing a local variable like this:

```python
a = 3
def f(x):
    x = 2
    return x
print(f(a))  # 2
print(a)     # 3
```

*However*, this *can* be done for mutable variables like lists, dictionaries, etc. As a principle, *never* change input variables within a function.
Default Arguments

default (eng) = standard (sv)

The standard way ...

```
def mult(x, c):
    return c*x
```

a typical call

```
y=mult(2., 13.)
# returns 26
y=mult(c=13., x=2.)
# returns 26
```

... now with defaults

```
def mult(x, c=1.0):
    return c*x
```

possible calls

```
y=mult(2., 13.)
# returns 26
y=mult(2.)
# returns 2
```

In the definition of the function mandatory parameters must precede optional parameters (those with default values).

Why?

C. Führer, A. Sopasakis, Lund University, Sweden
Arguments, Parameters – Summary

Make sure that you understood the difference

- between arguments and parameters
- between function definition and function evaluation (call)
- between positional arguments and keyword arguments
Docstrings

All functions (and everything else) should be documented carefully:

A docstring is the leading comment in a function (or class):

```python
def newton(f, x0):
    """
    Newton’s method for computing a zero of a function
    on input:
    f (function) given function f(x)
x0 (float) initial guess
    on return:
    y (float) the approximated zero of f
    """
    ...
```

`help(newton)` in Python or `newton?` in IPython displays the docstring as a help text.