4 - Functions

1. Definition of function
2. Function call
3. Global and local variables
4. Modules of functions

4.1 – Definition of function

- Functions are used to decompose programs into fragments that are re-usable.
- A function is a Python program defined from **input parameters** and it possibly returns an **output value**.
- The syntax if a function definition in Python is the following:

  def *<name of the function> (<list of parameters>) :*

  <block of instructions>

- An instruction **return** *<expression>* inside *<block of instructions>* causes the stopping of the execution of the function and the return of the value of *<expression>*.

4.1 - Definition of function

```
>>> def count_letter(letter, text) :
      n=0
      for c in text :
            if c == letter :
                  n += 1
      return "number of occurences of letter " + letter + " : " + `n`

>>> print count_letter('e', 'je reviens')
number of occurrences of letter e : 3
>>>
```

4.2 – Function call

- After it was defined, a function can be used in a particular expression named function call with the following syntax $f(v_1, v_2, \ldots, v_n)$, where $v_1, v_2, \ldots, v_n$ are expressions whose values are transmitted to parameters. With this instantiation of parameters, the subprogram constituting the body of the function is then executed.

- Python offers a default mechanism of parameter instantiation. The list of parameters can be written in the head of a function definition as follows:

$$p_1, p_2, \ldots, p_k, p_{k+1} = expr_1, p_{k+2} = expr_2, \ldots, p_{k+n} = expr_n$$

The k first parameters have to be instantiated at function call but the n last ones not necessarily. A function call is executed with k arguments at least and k+n arguments at most. If some parameter p_{k+i} is not instantiated explicitely, it takes the default value $expr_i$

4.2 - Function call

```
>>>def plural(word, family = 'standard'):
    if family == 'standard' :
        return word + 's'
    if family == 's':
        return word
    if family == 'oux' :
        return word + 'x'
    if family == 'al' :
        return word[:-1] +'ux'
```

```
>>> print plural(‘maison’)
‘maisons’

>>> print plural(‘souris’, ‘s’)
‘souris’

>>> print plural(‘chou’, ‘oux’)
‘choux’

>>> print plural(‘cheval’, ‘al’)
‘chevaux’

>>>
```

4.3 - Local variables and global variables

- The variables and parameters that are introduced in the definition of a function can be used in the rest of the definition but not outside. These variables are called **local variables** by opposition to **global variables** that are introduced ouside the definition of any function. Global variables can be used everywhere.

- If the same name is used for a global variable and for a local variable, Python distinguishes two variables, but inside the function definition where the local variable has been introduced, the common name only refers to the local variable.

4.3 - Local variables and global variables

```
>>> def f(x):
      y=2
      return x + y

>>> print f(3)
5
>>> print y

Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    print y
NameError: name 'y' is not defined
>>> u = 7
>>> def g(v):
      return u * v

>>> print g(2)
14
```

```
>>> def h(u):
      return u

>>> print h(3)
3
>>> print u
7
>>> def k(w) :
      u = 5
      return w+u

>>> print k(3)
8
>>> print u
7
>>>
```

4.4 - Function modules

- Function definitions that are related to the same application, can be stored in a common script called **module**. A module has the form of a file with the name in the form *<nom du module>.py*.

- A module is used through the instruction *import <nom du module>.* Running this instruction consists in executing the script constituting the module (this script may contain other instructions than function definitions).

- To run the instruction *import*, Python needs to be aware of the path used to access the module file. This path must appear in the list of potential paths stored in the *path* variable of module *sys.*

4.4 - Function modules

- Then, objects introduced in the module are used being referred to as *<name of module>.<name of object>.*

- To avoid prefixing object names with module names, one uses the following instruction ***from*** *<name of module>* ***import*** *<name of the object>.*
  To apply the instruction to all objects of the module, one uses the instruction ***from*** *<nom of module>* ***import*** *

- Modules can be gathered in **packages** and the operation can be iterated at the level of packages themselves. The instruction *import* can be uses to import packages in the same way as modules.

- Python provides predefined functions but also a standard library of modules, the description of which is available at the Web address: *http://docs.python.org/lib/lib.html*

4.4 - Function modules

```
# module related to circle located in the file “ /Users/perrier/Desktop/figures/circle.py”
# « /Users/perrier/Desktop/figures » is a package of modules
pi = 3.14

def surface(radius):
    return pi*radius**2

def length(radius):
    return 2*pi*radius
```

```
>>> import sys
>>> sys.path.append('/Users/perrier/Desktop/')
>>> from figures import circle
>>> r = 5
>>> print ’surface of a circle with ', r, ' cm as radius : ', circle.surface(r), '
      cm2'
surface of a circle with 5 cm as radius :  78.5  cm2
>>> from figures.circle import surface
>>> print ’surface of a circle with ', r, ' cm as radius : ', surface(r), ' cm2'
Surface of a circle with 5 cm as radius :  78.5  cm2
```

4.5 - Exercises

1. The Soundex algorithm is used in libraries to represent names of persons. It has the advantage to be not very sensitive to variations of name writing; for instance, Jurafsky, Jarofsky, Jarovsky et Jarovski are all represented with J612.

   Write functions performing the following tasks composing the algorithm:

   a) In a proper name, keep the first letter and delete all other letters that are different from a, e, h, i, o, u, w, y.

   b) From the output of the first function, replace the other letters than the first one with natural numbers according to the following rules:

      b, f,p,v → 1   c,g,j,k,q,s,x,z → 2   d,t → 3   l → 4   m,n → 5
      r → 6

4.5 - Exercises

c) From the output of the previous function, replace the sequences of identical digits with a unique digit (666 → 6)

d) From the output of the previous function, delete the digits beyond the three first ones or if digits are missing to have three digits, complete with zeros.

With all these functions, build a function that takes a text as its input and returns the same text where all proper nouns are replaced with their representation given by the Soundex algorithm.

4.5 - Exercises

2. Binary trees are represented as triples (label of the root, left sub-tree, right sub-tree). The empty tree is regarded as a particular binary tree represented with the empty triple (). For instance, here is a binary tree: (2, (4, (), (1, (), ())), (4, (), ())) Define the functions that manipulate binary trees and et return :

   a) The depth of a tree.

   b) The list of the labels for the tree leaves.

   c) The list of the labels for all nodes.

   d) A tree that is the initial tree where all occurrences of a given label are replaced with another given label.

   e) The tree that is the symetrical one of the initial tree with respect to its vertical axis.