introduction

November 23, 2023

1 The Python language

- Python is an interpreted language (by opposition to a compiled language like C or Fortran)
- Interpreted language are easier to use than compile language but slower. This is not a problem for scientific calculation because complex algorithms are programmed in C or Fortran.
- There are two versions of Python, the version 2 (currently 2.7) and the version 3 (currently 3.11). There are small differences in the syntax. The version 2.7 is obsolete since Jan. 2020 but still in use.
- We strongly advise to install the Anaconda distribution `https://www.anaconda.com/download/`. This distribution was build for scientific calculation. It is available for different platforms (Linux, Mac or Windows).

2 A taste of Python

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

[23]:
```
x = 3.14
epsilon = 1E-6
result = 0
n = 1
term = 1 # Initial value
while abs(term)>epsilon :
    result = result + term
    term  = term * x/n
    n = n+1
print(result)
```

```
23.103865905895475
```

3 How to execute Python code

https://docs.anaconda.com/anaconda/install/verify-install/

- python command
- IPython

- Spyder
- Jupyter notebook

[ ]:

4 Variable in Python

The name of a variable is any sequence of letters or numbers or _ which does not starts with a number. Variable are case sensitive

[11]:
```
this_is_a_variable = 23
this_is_a_variable = 2
this_is_a_variable
```

[11]: 2

5 Functions

[28]:
```
def exp(x):
    """ calculate e to the power x

    Use the power serie e^x = sum_i .....
    x is a float

    return a float"""
    epsilon = 1E-6
    result = 0
    n = 1
    term = 1
    while abs(term)>epsilon :
        result = result + term
        term  = term * x/n
        n = n + 1
    return result
```

[27]:
```
exp?
```

[22]:
```
exp(1)
```

[22]: 2.7182815255731922

6 Data types in Python

6.1 Numbers

- int (unlimited size)

- float (64 bits)
- complex (two floats)

[32]:
```
2**1034
```

[32]:
```
1840837770099011489514808515367961327224808426436921930479924031055182600248329862478934807781453168856269966129883067982426007232659626214326757689748215033628343228670622569229334728716760003783199569429350459072902662987186819906292870251938070908552709229410163693977059798410032294961514048815352055165091840
```

[33]:
```
2 + 4/5 *4
```

[33]:
```
5.2
```

[39]:
```
1345//15 # integer divition
1345%15 # modulo
```

[39]:
```
10
```

[41]:
```
a = 1E-6
a
```

[41]:
```
1e-06
```

[46]:
```
a = 1
print( (a + 1E-15) - a)
```

```
1.1102230246251565e-15
```

[47]:
```
5*2**(-52)
```

[47]:
```
1.1102230246251565e-15
```

[49]:
```
z = 1 + 1J
z**3
```

[49]:
```
(-2+2j)
```

[52]:
```
z.imag
(1+1J).imag
```

[52]:
```
1.0
```

[51]:
```
z.real
```

[51]:
```
1.0
```

6.2 Boolean and comparison

- True; False
- and, or, not
- And or are functions, not operator
- Be carreful of priority
- Avoid binary operator (&, ^, |)

[53]:
```
True
False
```

[53]: False

[55]:
```
(1==1)
(1<5)
(1<=5)
(1!=4)
```

[55]: True

[56]:
```
not True
```

[56]: False

[61]:
```
x = 1
(x>0.5) and (x<2)

#if (x>0.5):
#    if (x<2):
#        return True
#return False
from math import sqrt

x = -1
if (x>0) and sqrt(x)>2:
    print('Hello')
```

[66]:
```
False or (True and False)
```

[66]: False

[67]:
```
True & False
```

[67]: False

[72]:
```
6 ^ 4
```

[72]: 2

[ ]:

[74]:
```
print(7==4 | 3==7)
```

```
True
```

[75]:
```
x = -1
if x>0 and log(x)>4:
    print("Hello")
```

6.3 Strings

- format
- concatenation

[76]:
```
s = "Peter"
s = 'Peter'
s = "Peter's dog"
s = 'Peter\'s dog say "hello"'
s = """Hello
What time is it ?"""
s
```

[76]:
```
'Hello\nWhat time is it ?'
```

[77]:
```
s="""1
2"""
s
```

[77]:
```
'1\n2'
```

[78]:
```
len(s)
```

[78]:
```
3
```

[80]:
```
print(s)
```

```
1
2
```

[81]:
```
s = '1'
s
```

[81]:
```
'1'
```

[83]:
```
a = 1
print(s)
print(a)
```

```
print(repr(s))
```

```
1
1
'1'
```

[91]:
```
s = """Hello
What time is it ?"""

s[15:17] + s[17:20]
s[20:]
N = len(s)

s[N-1]
s[-1]
```

[91]: '?'

[93]:
```
a = exp(1)
'The value of a is '+str(a)+' m/s'
```

[93]: 'The value of a is 2.7182815255731922 m/s'

[ ]:
```

```

[98]:
```
hour = 15
minute = 30
#s = "It's {h}:{mn}".format(h=hour, mn=minute)
s = f"It's {hour}:{minute}"
print(s)
```

```
It's 15:30
```

[103]:
```
from math import pi
print(f'{pi:010.5f}') # '3.14159'
c = 299792458. # Speed of light in m/s
print(f'c = {c:.3e} m/s') # '2.998e+08'
```

```
0003.14159
c = 2.998e+08 m/s
```

[104]:
```
hour = 15
minute = 3
#s = "It's {h}:{mn}".format(h=hour, mn=minute)
s = f"It's {hour:02d}:{minute:02d}"
print(s)
```

```
It's 15:03
```

Common methods on string are already implemented

- split
- strip
- join
- startswith, endswith
- lower(), upper()
- comparison (alphabetic order) : 'Peter'>'John' is True.

[107]:
```
s = 'hello'
s.upper()
```

[107]:
```
'HELLO'
```

[109]:
```
if s.startswith('he'):
    print('Bonjour')
```

```
Bonjour
```

[110]:
```
s = "1, 2, 5, 7"
s.split(',')
```

[110]:
```
['1', ' 2', ' 5', ' 7']
```

[111]:
```
s = "    bonjou     "
s.strip()
```

[111]:
```
'bonjou'
```

[112]:
```
' and '.join(['a', 'b', 'c'])
```

[112]:
```
'a and b and c'
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:
```
s = "  Where is Brian? Brian is in the kitchen.   \r\n"
s = s.strip() # string with leading and trailing whitespaces characters removed
word_list = s.split() # list containing the words
word_set = set(word_list)
print("The sentence contains {0} different words".format(len(word_set)))
print("The words are {}.".format(' and '.join(list(word_set))))
if "Brian" in s:
```

```
    print("The word Brian is in the sentence")
```

[ ]:

6.4 List in python

- List creation
- Modification of an element
- The command `range(n)`
- A list can contain elements of any type (list containing a list)
- *list comprehension*
- List comprehension can be used to filter a list
- There are two convenient ways to loop through a list

[113]:
```
l = [1, 5, 'hello']
l
```

[113]:
```
[1, 5, 'hello']
```

[116]:
```
l = []
l.append('Bonjour')
l.append('Hello')

l.insert(1, 'coucou')
l[1] = 'Coucou'
l
```

[116]:
```
['Bonjour', 'Coucou', 'Hello']
```

[118]:

[118]:
```
[1, 2, [4, 5], <function print>]
```

[121]:
```
l = [1, 2, [4, 5], print]
l.append('Bonjour')
l.append(l)
l[-1][-1][-1]
```

[121]:
```
[1, 2, [4, 5], <function print>, 'Bonjour', […]]
```

[ ]:

[122]:
```
l = [1, 2, 3, 4]
l = [] # Empty list
l.append(3) # now l==[3]
l.append(4) # now l==[3, 4]
l.insert(0,3.24+1j) # now l==[3.24+1j,3,4]
```

[153]:
```
list('Hello')
```

[153]:
```
['H', 'e', 'l', 'l', 'o']
```

[ ]:
```

```

[124]:
```
l = [1, 3, 5, "Pierre"]
for elm in l:
    print(elm)

# for page in book:
```

```
1
3
5
Pierre
```

[128]:
```
for i, list_item in enumerate(l):
    print(list_item, " is the item number ",i," of the list")
```

```
1  is the item number  0  of the list
3  is the item number  1  of the list
5  is the item number  2  of the list
Pierre  is the item number  3  of the list
```

[129]:
```
#for i in range(len(l)):
#    print(l[i])
```

```
1
3
5
Pierre
```

6.5 Tuple

- Tuples are used to collect few objects together
- Tuple are used when a function returns more that one value

[131]:
```
t = (1, 2)
t[0]
```

[131]:
```
1
```

[132]:
```
t[0] = 4
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-132-08a0409f2124> in <module>
```

```
----> 1 t[0] = 4

TypeError: 'tuple' object does not support item assignment
```

[133]:
```
def test():
    return 1, 2

test()
```

[133]: (1, 2)

[135]:
```
t = (2, 3)
x, y = t
print(y)
```

```
3
```

[136]:
```
x, y = test()
y
```

[136]: 2

[157]:
```
t = (1, )
t
```

[157]: (1,)

[ ]:

[ ]:

6.6 Dictionary

[138]:
```
d = {'a':'Hello', 1:"Bonjour", "age":199}
print(d['a'])
print(d[1])
```

```
Hello
Bonjour
```

[142]:
```
person1 = {'name':'Dupont', 'age':46, 'phone':"12345678"}
person2 = {'name':'Dupond', 'age':42, 'phone':"87654321"}
person1['age']
```

[142]: 46

[141]:
```
for key, value in person1.items():
    print(key, value)
```

```
name Dupont
age 46
phone 12345678
```

[147]:
```
phone_book = [person1, person2]

for person in phone_book:
    print(f'{person["name"]} is {person["age"]} years old.')
```

```
Dupont is 46 years old.
Dupond is 42 years old.
```

[149]:
```
for person in phone_book:
    birth_year = 2023 - person["age"]
    print(f'{person["name"]} was born in {birth_year}.')
```

```
Dupont was born in 1977.
Dupond was born in 1981.
```

[ ]:
```

```

6.7 Set

[150]:
```
a = set([1,2,3])
b = set([3,5,6])

c = a | b # union
d = a & b # intersection
print(c)
print(d)
```

```
{1, 2, 3, 5, 6}
{3}
```

[151]:
```
pwd = input('Enter a password with at least one punctuation :')
punctuation = set("?,.;:!")
if (punctuation & set(pwd)) == set():
    print("The password should contain at least one punctuation")
```

```
Enter a password with at least one punctuation :azerty
The password should contain at least one punctuation
```

[ ]:
```

```

6.8 Index in Python

- start:stop:step
- slice(start, stop, step)

[159]:
```
l = ['bonjour', 'Hello', 'Hallo']
l[1:3]
```

[159]:
```
['Hello', 'Hallo']
```

[160]:
```
s = 'Bonjour'
s[0:4:2]
```

[160]:
```
'Bn'
```

[ ]:
```

```

[ ]:
```

```

6.9 None

[163]:
```
def test():
    print('Bonjour')

a = test()
print(a)
```

```
Bonjour
None
```

[165]:
```
def my_sqrt(x):
    if x<0:
        return None
    else:
        return sqrt(x)

if my_sqrt(-1)==None:
    print('Bonjour')
```

```
Bonjour
```

[ ]:
```

```

[ ]:
```

```

7 Mutable objects / arguments in functions

[ ]:
```
a = 3 # Python creates the object #1 containing 3.
b = a + 4 # Python creates the object #2 containing 7
c = a # The symbol c point to object #1
a = b # The symbol a point to object #2
c = 3.14 # The symbol c point to a third object. There is no way
        # to point to object #1. Python can delete it.
```

[167]:
```
a = [2,3,7]
b = a
print(b[1])
a[1] = 4
print(b[1])
a = [5,6,7,8]
print(b[1])
```

```
3
4
4
```

[169]:
```
def exemple(arg):
    print(arg[1])
    arg[1] = 4
    print(arg[1])
    arg = [5,6,7,8]

a = [1,2,3,4]
exemple(a)
print(a[1])
```

```
2
4
4
```

[170]:
```
# which number is displayed ?
a = [1, 2, 34, 45]
b = a
c = a[1]
a[2] = 1
a[1] = 5
print(b[2]+c)
```

```
3
```

[ ]:
```

```

7.1 Local and global variable

[171]:
```
pi = 3.141592
def function():
    print(pi*x)

x = 1
function()
x = 3
function()
```

```
1
3
```

[174]:
```
def function():
    x = 1
    print(x)

x = 3
function()
print(x)
```

```
1
3
```

[175]:
```
def function():
    print(x)
    x = 1
    print(x)

x = 3
function()
```

```
---------------------------------------------------------------------------
UnboundLocalError                         Traceback (most recent call last)
<ipython-input-175-9aa2db775929> in <module>
      5
      6 x = 3
----> 7 function()

<ipython-input-175-9aa2db775929> in function()
      1 def function():
----> 2     print(x)
      3     x = 1
      4     print(x)
      5
```

```
UnboundLocalError: local variable 'x' referenced before assignment
```

[176]:

[176]: 1.0

[ ]:

7.2 The global instruction in Python

Forget it !

[ ]:

8 Control structure

8.1 For loop

- enumerate
- zip

[177]:
```
X = [1,3,4,7]
Y = [3,5,1,2]
for x,y in zip(X,Y):
    print(x,y)
```

```
1 3
3 5
4 1
7 2
```

[178]:
```
l = [1, 2, 4, 6, 7, 8, 10]
for start, stop in zip(l[:-1], l[1:]):
    print('length = {}'.format(stop-start))
```

```
length = 1
length = 2
length = 2
length = 1
length = 1
length = 2
```

[179]:
```
m = 2023
```

[181]:
```
from math import ceil, sqrt
p_max = int(ceil(sqrt(m)))
for p in range(p_max+1):
```

```
    if p<=1:
        continue
    if m%p==0:
        is_prime = False
        break
else:
    is_prime = True
print(is_prime)
```

```
7
False
```

[183]:
```
def is_prime(m):
    p_max = int(ceil(sqrt(m)))
    for p in range(p_max+1):
        if p<=1:
            continue
        if m%p==0:
            return False
    return True
is_prime(2023)
```

[183]:
```
False
```

8.2 Generators

[187]:
```
def simple_generator():
    print('A')
    yield 1
    print('B')
    yield 2
    print('C')
    yield 3
    print('D')


for item in simple_generator():
    print(item)
    if item>=2:
        break
```

```
A
1
B
2
```

[189]:
```
for i in range(10000000000):
    print(i)
    if i>2:
        break
```

```
0
1
2
3
```

[ ]:
```
def concatenate(liste1, liste2):
    for elm in liste1:
        yield elm
    for elm in liste2:
        yield elm
```

[190]:
```
def matrix_index_generator(N1, N2):
    for i in range(N1):
        for j in range(N2):
            yield (i, j)

list(matrix_index_generator(3, 4))
```

[190]:
```
[(0, 0),
 (0, 1),
 (0, 2),
 (0, 3),
 (1, 0),
 (1, 1),
 (1, 2),
 (1, 3),
 (2, 0),
 (2, 1),
 (2, 2),
 (2, 3)]
```

[ ]:
```

```

8.3 Function

[194]:
```
from scipy.integrate import quad

quad?
```

[195]:
```
from math import exp
y, _ = quad(exp, 0, 10, epsrel=1E-3)
y
```

[195]: 22025.465794806725

[196]:
```
option = {"epsrel":1E-3, 'limit':100}
quad(exp, 0, 1, **option)
```

[196]: (1.7182818284590453, 1.9076760487502457e-14)

8.3.1 Lambda function

[205]:
```
def function(x):
    return x + 1

g = function

g.__name__
```

[205]: 'function'

[208]:
```
f = lambda x:x+1
f(1)
f.__name__

(lambda x:x+1)(1)
```

[208]: 2

[ ]:
```

```

8.3.2 Variable length argument list

[197]:
```
print(1, 2, 'Hello')
```

```
1 2 Hello
```

[199]:
```
def my_function(a, b, *args, **kwd):
    print(args)
    print(kwd)

my_function(1,2,3,4, my_variable='Hello', g=45)
```

```
(3, 4)
{'my_variable': 'Hello', 'g': 45}
```

[203]:
```
data = (1, 2, 3,)

my_function(*data)
```

```
(3,)
{}
```

[ ]:

[ ]:

[213]:
```
def erf(x, **kwd):
    return quad(lambda x:exp(-x**2), 0, x, **kwd)

print(erf(1, epsrel=1E-4))
```

```
(0.7468241328124271, 8.291413475940725e-15)
```

[ ]:

9 Accents and non Latin letters

[214]:
```
name = "Pierre Cladé"
name
```

[214]: 'Pierre Cladé'

[215]:
```
s = " "
print(s)
```

[225]:
```
'\u03b1'
```

[225]: ' '

[222]:
```
hex(ord(' '))
```

[222]: '0x3b1'

[ ]:
```
  = 1/137.035990
```

[226]:
```
A = 1
A = 2
print(A)
print(A)
```

```
1
2
```

[ ]:

[ ]:

[227]:
```
print(name.encode('latin-1'))
print(name.encode('utf-8'))
```

```
b'Pierre Clad\xe9'
b'Pierre Clad\xc3\xa9'
```

[228]:
```
" = 1/137".encode('utf-8')
b'\xce\xb1 = 1/137'.decode('utf-8')
```

[228]:
```
' = 1/137'
```

[ ]:

10 Files and file like objects

10.1 Files

- `write(str)` : To write one string in the file
- `read` : To read all the file. `read(n)` to read a given number of characters.
- `readline` : read one line of the file
- `readlines` : return a list with one item per line.

[240]:
```
f = open('test.txt')
for line in f.readlines():
    print(line.strip())
f.close()
```

```
Bonjour !
Hello!!!
```

10.2 With statement

[241]:
```
with open('test.txt') as f:
    print(f.read(5))
```

```
Bonjo
```

[244]:
```
with open('test.txt', 'a') as f:
    f.write('Hello')
```

[ ]:

[ ]:

11 Json

- json.dump(obj, f)
- json.load(f)

[245]:
```
import json

person1 = {'name':'Dupont', 'age':46, 'phone':"12345678"}
person2 = {'name':'Dupond', 'age':42, 'phone':"87654321"}
phone_book = [person1, person2]

phone_book
```

[245]:
```
[{'name': 'Dupont', 'age': 46, 'phone': '12345678'},
 {'name': 'Dupond', 'age': 42, 'phone': '87654321'}]
```

[247]:
```
with open('my_phone_book.json', 'w') as f:
    json.dump(phone_book, f, indent=2)
```

[248]:
```
with open('my_phone_book.json') as f:
    data = json.load(f)

data
```

[248]:
```
[{'name': 'Dupont', 'age': 46, 'phone': '12345678'},
 {'name': 'Dupond', 'age': 42, 'phone': '87654321'}]
```

[ ]:
```

```

12 Modules

12.1 Creating a module

12.2 Importing from a module

[ ]:
```

```

13 Package

13.1 Installation of a package

13.2 Create your package

13.3 Local import

13.4 Distribute your package

[ ]:

[ ]:
```
from distutils.core import setup
__version__ = "alpha"

long_description="""This is a very nice package

"""

setup(name='my_package',
      version=__version__,
      description='A very nice package',
      author=u'François Pignon',
      author_email='francois.pignon@trucmuch.fr',
      url='',
      packages=['my_package'],
     )
```

[ ]:

14 Error

Solution of the equation : $ax^2 + bx + c = 0$

[270]:
```
from math import sin, sqrt

a = sin(1)

b = cos(2)

a,b,c = 2,8,4
Delta = b**2 - 4*a*c
root1 = (-b + sqrt(Delta)/(2*a) )
root_number_2 = (-b - sqrt(Delta)/(2*a) )
root_number_2
```

[270]:
```
-9.414213562373096
```

[257]:
```
mylist = [1,2,34]
print(mylist(2))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-257-0100b3b922c2> in <module>
      1 mylist = [1,2,34]
----> 2 print(mylist(2))

TypeError: 'list' object is not callable
```

[262]:
```
if 1==1:
    print('Hello')
    print('World')

if 1==1:
    print('Hello World')
```

```
Hello
World
Hello World
```

[ ]:

14.1 Exceptions

Generate you own exceptions

Exemple : solution of $ax^2 + bx + c = 0$

[ ]:

15 Python Standard Library

- string — Common string operations
- re — Regular expression operations
- datetime — Basic date and time types
- math — Mathematical functions
- shutil — High-level file operations
- os — Miscellaneous operating system interfaces
- logging — Logging facility for Python
- email — An email and MIME handling package
- sys — System-specific parameters and functions

- urllib — Open arbitrary resources by URL
- time — Time access and conversions

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: