

# Introduction à Python

Septembre 2010

# Plan

- 1 Découverte de l'environnement Linux
  - Le bureau
  - Arborescence
- 2 Premiers pas
  - Python en mode consol
  - Python avec Eric4
- 3 Outils de base
  - Variables et écriture
  - Boucles, tests
  - Fonctions

# Ouverture d'une session

- login : aeinstein
- mot de passe : \*\*\*\*\*

# Le bureau

- Les applications
- Les raccourcis
- Le système
- Tableau de bord du bas
  - La barre de tâches
  - Les bureaux virtuels

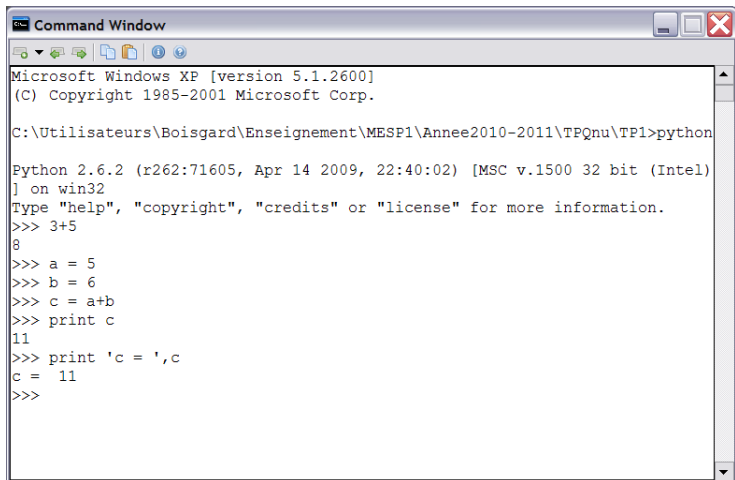
# Structure des fichiers

# Utilisation du Terminal

## Les commandes de base

- `dir` ou `ls` : affiche le contenu d'un répertoire
- `cd` (pour change directory) :
  - `cd nom_du_repertoire_enfant` : descend d'un cran dans l'arborescence
  - `cd ..` : remonte vers le répertoire « père »
- astuce : taper les premières lettres puis « `tab` » pour compléter

# Lignes de commande



```
Microsoft Windows XP [version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Utilisateurs\Boisgard\Enseignement\MESPl\Annee2010-2011\TPQnu\TP1>python

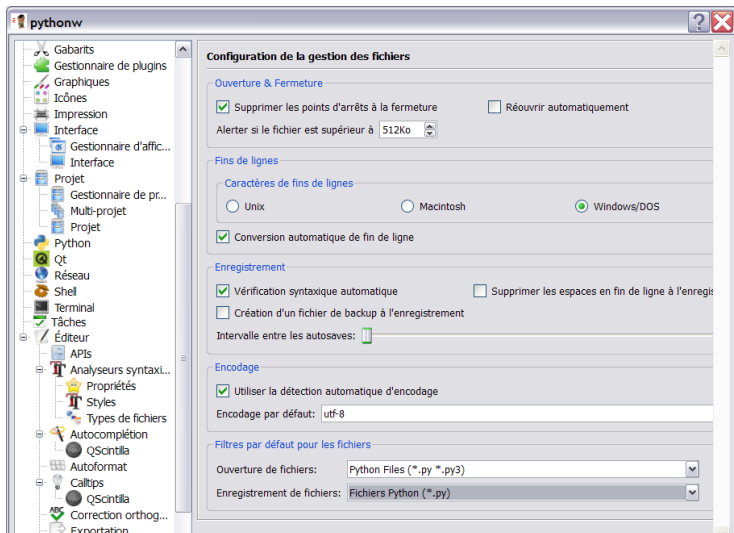
Python 2.6.2 (r262:71605, Apr 14 2009, 22:40:02) [MSC v.1500 32 bit (Intel)
] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 3+5
8
>>> a = 5
>>> b = 6
>>> c = a+b
>>> print c
11
>>> print 'c = ',c
c = 11
>>>
```

# Attention à la division !

```
C:\Utilisateurs\Boisgard\Enseignement\I
Python 2.6.2 (r262:71605, Apr 14 2009,
] on win32
Type "help", "copyright", "credits" or
>>> 2/3
0
>>> 2./3
0.66666666666666663
>>> █
```



# Configuration de l'IDE



# Premiers scripts

```
script1.py
1  """
2  Commentaire
3  """
4  g = 9.8
5  v0z = 10
6  t = 10
7
8  z = -0.5*g*t**2+v0z*t
9  print 'z =', z
10
```

A rajouter automatiquement

```
script1.py  script2.py
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  Calcul de la position verticale d'un mobile lancé
5  avec une vitesse initiale v0z
6  """
7  g = 9.8  # accélération de la pesanteur
8  v0z = 10  # vitesse initiale
9  t = 10  # instant
10
11  z = -0.5*g*t**2+v0z*t
12  print 'z =', z
13
```

- Enregistrer votre script modèle dans « fichiers étiquetés »

# Les bibliothèques (1)

bib1.py	bib2.py	bib3.py	bib4.py
1	#!/usr/bin/env python		
2	# -*- coding: utf-8 -*-		
3	"""		
4	racine carrée 1		
5	"""		
6	<b>import</b> math		
7			
8	a = math.sqrt(2)		
9	<b>print</b> a		
10			

re horizontale

```
1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237
```

bib1.py	bib2.py	bib3.py	bib4.py
1	#!/usr/bin/env python		
2	# -*- coding: utf-8 -*-		
3	"""		
4	racine carrée 1		
5	"""		
6	<b>import</b> math <b>as</b> m		
7			
8	a = m.sqrt(2)		
9	<b>print</b> a		
10			

re horizontale

```
1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237
```

# Les bibliothèques (2)

bib1.py	bib2.py	bib3.py	bib4.py
1	#!/usr/bin/env python		
2	# -*- coding: utf-8 -*-		
3	"""		
4	racine carrée 1		
5	"""		
6	from math import sqrt		
7			
8	a = sqrt(2)		
9	print a		
10			

erre horizontale

```
1 Python 2.6.2 (r262:71605, Apr 14 2006)
2 >>> 1.41421356237
```

bib1.py	bib2.py	bib3.py	bib4.py
1	#!/usr/bin/env python		
2	# -*- coding: utf-8 -*-		
3	"""		
4	racine carrée 1		
5	"""		
6	from math import *		
7			
8	a = sqrt(2)		
9	print a		
10			

horizontale

```
1 Python 2.6.2 (r262:71605, Apr 14 2006)
2 >>> 1.41421356237
```

## Bibliothèque (3)

On utilisera essentiellement

- ❶ la bibliothèque `scipy` : `from scipy import *`  
contient toutes les fonctions mathématiques standards + sous-bibliothèques (FFT, solution équations,...)
- ❷ la bibliothèque `matplotlib` : `from pylab import *`  
permet de tracer des plots 2D (et aussi un peu de 3D)

# Variables

## types

### ① numériques :

- entier :  $x = 3$ 
  - réel :  $x = 3.14156$
  - complexe :  $x = 5 + 4j$

### ② texte : $T = \text{"toto"}$

### ③ ensembles :

- liste :  $L = [4,6,8] \rightarrow$ modifiable
- tuple :  $t = (4,6,8) \rightarrow$  non modifiable
- tableau :  $T = \text{array}([1,2,3]) \rightarrow$  type non standard (bibliothèque Numpy)

# Affectations et opérations (1)

```
op_num.py
3 - """
4   opérations
5   """
6   x = 2
7   y = 4
8   z = x + y
9   print z
10
```

---

```
1 Python 2.6.2 (r262:
2 >>> 6
3
```

```
op_texte.py
3 - """
4   opérations
5   """
6   T1 = 'toto'
7   T2 = 'titi'
8   T = T1 + ' ' + T2
9   print T
10
```

---

```
1 Python 2.6.2 (r262:71605
2 >>> toto titi
3
```

## Affectations et opérations (2)

```
liste1.py
3  """
4  opérations sur les listes
5  """
6  L1 =[1, 2, 3]
7  L2 =[11, 12, 13]
8  L = L1 + L2
9  print L
10 print L[0], L[3]
```

erre horizontale

```
1 Python 2.6.2 (r262:71605, Apr 10 2008, 14:56:05) [AMD64] on win32
2 >>> [1, 2, 3, 11, 12, 13]
3 1 11
```

```
array1.py
4  """
5  opérations sur les listes
6  """
7  from scipy import *
8  A1 =array([1, 2, 3])
9  A2 =array([11, 12, 13])
10 A = A1 + A2
11 print A
```

erre horizontale

```
1 Python 2.6.2 (r262:71605, Apr 10 2008, 14:56:05) [AMD64] on win32
2 >>> [12 14 16]
```



# Listes et tableaux sur mesure

```

liste2.py
4 opérations sur les listes
5 """
6 L1 = range(10)
7 print L1
8 L2 = range(3, 12, 2)
9 print L2
10 L3 = range(10, 2, -1)
11 print L3

erre horizontale
1 Python 2.6.2 (r262:71605, Apr 14 2009, 22:40:02) [MSC v.15006480 64-bit x86]
2 >>> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
3 [3, 5, 7, 9, 11]
4 [10, 9, 8, 7, 6, 5, 4, 3]

```

```

array2.py
3 - """
4 opérations sur les listes
5 """
6 from scipy import *
7 A1 = arange(2, 5, 0.3)
8 print A1
9 A2 = linspace(0, 10, 7)
10 print A2

e horizontale
1 Python 2.6.2 (r262:71605, Apr 14 2009, 22:40:02) [MSC v.15006480 64-bit x86]
2 >>> [ 2.  2.3  2.6  2.9  3.2  3.5  3.8  4.1  4.4  4.7]
3 [ 0.          1.66666667  3.33333333  5.          6.66666667
4  8.33333333 10.          ]

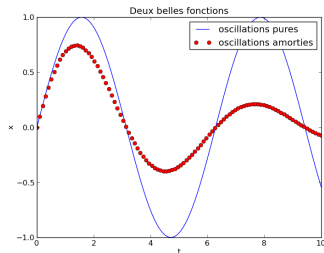
```

# Applications : plots 0

```
1  #! /usr/bin/env python
2  # -*- coding: utf-8 -*-
3  """
4  plots de base
5  """
6  from scipy import *
7  from pylab import *
8
9  t = linspace(0, 10, 100)
10 x1 = sin(t)
11
12 plot(t, x1)
13
14 show()
15
```

# Application : plots

```
4 plots de base
5 """
6 from scipy import *
7 from pylab import *
8
9 t = linspace(0, 10, 100)
10 x1 = sin(t)
11 x2 = exp(-t/5.)*sin(t)
12
13 plot(t, x1)
14
15 figure(2)
16 plot(t, x1, color='b', label=ur'oscillations pures')
17 plot(t, x2, 'o', color='r', label=ur'oscillations amorties')
18 title(ur'Deux belles fonctions')
19 xlabel(ur't')
20 ylabel(ur'x')
21 legend()
22
23 show()
```



# Boucle for

```
boucle_for.py
2  # -*- coding: utf-8 -*-
3  """
4  opérations sur les listes
5  """
6  L = []
7  for n in range(10):
8      L = L + [2**n]
9  print L
```

erre horizontale

```
1 Python 2.6.2 (r262:71605, Apr 14 2009, 22:
2 >>> [1, 2, 4, 8, 16, 32, 64, 128, 256, 512]
3
```

# If then else

```
test_if.py
4 tests
5 """
6 res1 = 2==3
7 print res1
8 res2 = 20==20
9 print res2
10 - for n in range(5):
11 -     if n%2==0:
12 -         print n, ' est pair'
13 -     else:
14 -         print n, ' est impair'
```

---

```
2 >>> False
3 True
4 0 est pair
5 1 est impair
6 2 est pair
7 3 est impair
8 4 est pair
```

# Fonctions 1

```
"""  
initiation aux fonctions  
"""  
  
from scipy import *  
from pylab import *  
  
def somme(a, b):  
    res = a + b  
    return res  
  
# somme de tableaux  
t = linspace(0, 10, 100)  
x = sin(t)  
y = cos(t)  
z = somme(x, y)  
plot(t, x, t, y, t, z)  
  
show()
```

## Fonctions 2

```
6  from scipy import *
7  from pylab import *
8
9  - def somme(a, b):
10      res = a + b
11      return res
12
13  # somme de tableaux
14  t = linspace(0, 10, 100)
15  x = sin(t)
16  y = cos(t)
17  z = somme(x, y)
18  plot(t, x, t, y, t, z)
19
20  show()
```

