Python pour les scientifiques

Introduction

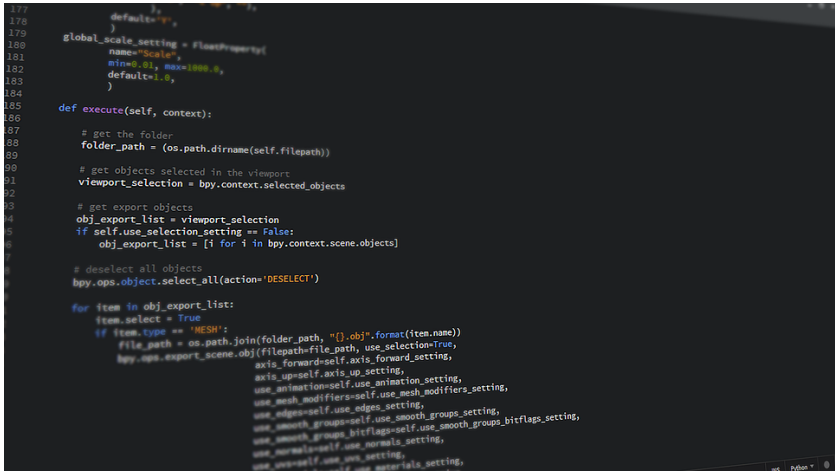
Cyril Desjouy

June, 2016
Updated : June 29, 2018

Ce cours est inspiré de documents de...

- Guido van Rossum (Python's father)
- Matt Huenerfauth (Penn State)
- Richard P. Muller (Caltech)
- SAO Telescope Data Center Team (Harvard)
- Jake VanderPlas (University of Washington)
- David Pine (New York University - Department of Physics)
- Scott Shell (UC Santa Barbara engineering)
- Gaël Varoquaux (Inria Saclay)
- Andrew M. C. Dawes (Pacific University of Oregon)
- Bruno Brouard (Le Mans Université)
- ...

La programmation



Extrait de Pixabay

Chapitre adapté du cours de programmation de Bruno Brouard

Comment résoudre un problème donné ?

Exemple : "*Faire une omelette avec 6 oeufs et la servir à 20h.*"

Prérequis :

- Trouver un exécutant : le **processeur**
- . . . Sachant exécuter certaines actions : les **instructions**

Comment résoudre un problème donné ?

Exemple : "*Faire une omelette avec 6 oeufs et la servir à 20h.*"

Prérequis :

- Trouver un exécutant : le **processeur**
- . . . Sachant exécuter certaines actions : les **instructions**

Les instructions sont énoncées en fonction du processeur :

- **Pour un enfant** (*instructions très détaillées*) :
  1. casser 6 œufs dans un bol à 19:30
  2. battre les œufs avec un fouet
  3. ajouter sel et poivre
  4. cuire à feu doux dans une poêle 15 minutes

Comment résoudre un problème donné ?

Exemple : "*Faire une omelette avec 6 oeufs et la servir à 20h.*"

Prérequis :

- Trouver un exécutant : le **processeur**
- . . . Sachant exécuter certaines actions : les **instructions**

Les instructions sont énoncées en fonction du processeur :

- **Pour un enfant** (*instructions très détaillées*) :
  1. casser 6 œufs dans un bol à 19:30
  2. battre les œufs avec un fouet
  3. ajouter sel et poivre
  4. cuire à feu doux dans une poêle 15 minutes
- **Pour un adulte** (*instructions concises*) :
  1. nombre d'œufs = 6
  2. heure du diner = 8PM

Comment résoudre un problème donné ?

Exemple : "*Faire une omelette avec 6 oeufs et la servir à 20h.*"

Prérequis :

- Trouver un exécutant : le **processeur**
- . . . Sachant exécuter certaines actions : les **instructions**

Les instructions sont énoncées en fonction du processeur :

- **Pour un enfant** (*instructions très détaillées*) :
  1. casser 6 œufs dans un bol à 19:30
  2. battre les œufs avec un fouet
  3. ajouter sel et poivre
  4. cuire à feu doux dans une poêle 15 minutes
- **Pour un adulte** (*instructions concises*) :
  1. nombre d'œufs = 6
  2. heure du diner = 8PM
- **Pour un ordinateur** (*séquence de 0 et de 1*) :
  1. 10011110100110011101001010010010100100101100111...

Comment résoudre un problème donné ?

Exemple : "*Faire une omelette avec 6 oeufs et la servir à 20h.*"

Prérequis :

- Trouver un exécutant : le **processeur**
- . . . Sachant exécuter certaines actions : les **instructions**

Les instructions sont énoncées en fonction du processeur :

- **Pour un enfant** (*instructions très détaillées*) :
  1. casser 6 œufs dans un bol à 19:30
  2. battre les œufs avec un fouet
  3. ajouter sel et poivre
  4. cuire à feu doux dans une poêle 15 minutes
- **Pour un adulte** (*instructions concises*) :
  1. nombre d'œufs = 6
  2. heure du diner = 8PM
- **Pour un ordinateur** (*séquence de 0 et de 1*) :
  1. 10011110100110011101001010010010100100101100111...

Instruction : Action encodée dans un langage connu par le processeur, contenant ou manipulant des données	**Algorithme :** Séquence d'instructions pouvant être exécutée par un processeur et permettant de résoudre un problème

Ordinateur : Rapide et puissant, mais dénoué d'intelligence et d'imagination *(pour le moment !)*

Pas de solution proposée à un problème donné !

Programmeur : Lent mais doué d'intelligence et d'imagination

Explique à l'ordinateur ce qu'on veut qu'il fasse

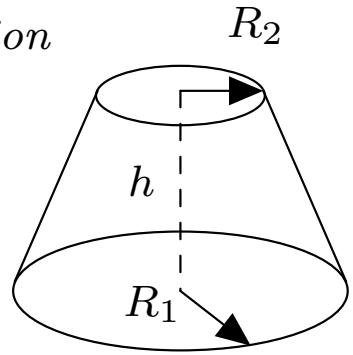
Pour arriver au programme, il existe trois étapes importantes :

1. **Réflexion**: Trouver une solution au problème
2. **Élaboration**: Expliquer comment obtenir cette solution à partir de rien, à travers un (ou des) algorithme(s)
3. **Encodage**: Choisir un langage compris par l'ordinateur, et traduire la solution algorithmique dans ce langage

Programme : Traduction d'un ou d'un ensemble d'algorithmes dans un langage compréhensible par l'ordinateur

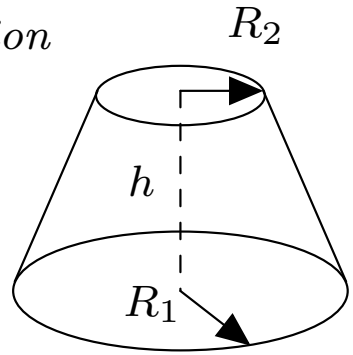
1. **Réflexion** : *trouver une solution au problème*

2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*



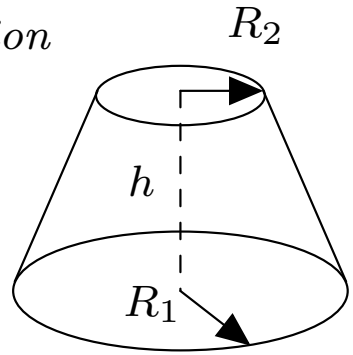
3. **Encodage** : *Choisir un langage et traduire l'algorithme*

1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*

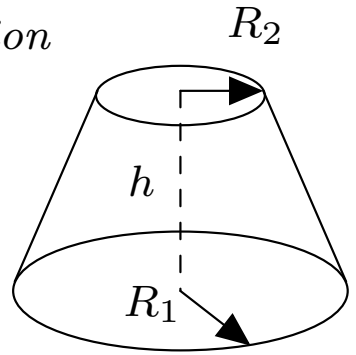


3. **Encodage** : *Choisir un langage et traduire l'algorithme*

1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*
   - 2.1 Lire le petit rayon et le sauvegarder sous le nom R2
   - 2.2 Lire le plus grand rayon et le sauvegarder sous le nom R1
   - 2.3 Lire la hauteur et la sauvegarder sous le nom H
   - 2.4 Calcul du volume et sauvegarde sous le sous VOL
   - 2.5 Écrire le résultat à l'écran
3. **Encodage** : *Choisir un langage et traduire l'algorithme*



1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*
   - 2.1 Lire le petit rayon et le sauvegarder sous le nom R2
   - 2.2 Lire le plus grand rayon et le sauvegarder sous le nom R1
   - 2.3 Lire la hauteur et la sauvegarder sous le nom H
   - 2.4 Calcul du volume et sauvegarde sous le sous VOL
   - 2.5 Écrire le résultat à l'écran

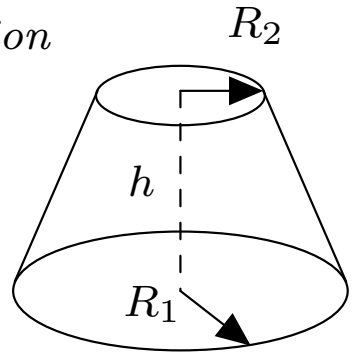


3. **Encodage** : *Choisir un langage et traduire l'algorithme*

Langage : Pyhon

```
from  math import  pi
R2 = float(input("Petit rayon ?  "))
R1 = float(input("Grand rayon ?  "))
H = float(input("Hauteur ?  "))
VOL = H*pi/3*(R1**2+R2**2+R1*R2)
print("Volume du cône tronqué :  ", VOL)
```

1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*
   - 2.1 Lire le petit rayon et le sauvegarder sous le nom R2
   - 2.2 Lire le plus grand rayon et le sauvegarder sous le nom R1
   - 2.3 Lire la hauteur et la sauvegarder sous le nom H
   - 2.4 Calcul du volume et sauvegarde sous le sous VOL
   - 2.5 Écrire le résultat à l'écran



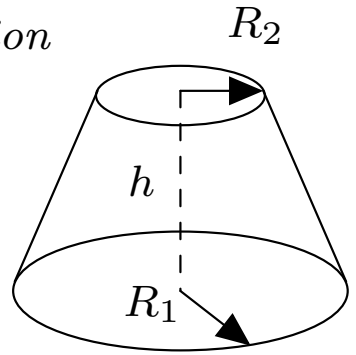
3. **Encodage** : *Choisir un langage et traduire l'algorithme*

Langage : Pyhon

```
from  math import  pi
R2 = float(input("Petit rayon ?  "))
R1 = float(input("Grand rayon ?  "))
H = float(input("Hauteur ?  "))
VOL = H*pi/3*(R1**2+R2**2+R1*R2)
print("Volume du cône tronqué :  ", VOL)
```

Variables

1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*
   - 2.1 Lire le petit rayon et le sauvegarder sous le nom R2
   - 2.2 Lire le plus grand rayon et le sauvegarder sous le nom R1
   - 2.3 Lire la hauteur et la sauvegarder sous le nom H
   - 2.4 Calcul du volume et sauvegarde sous le sous VOL
   - 2.5 Écrire le résultat à l'écran



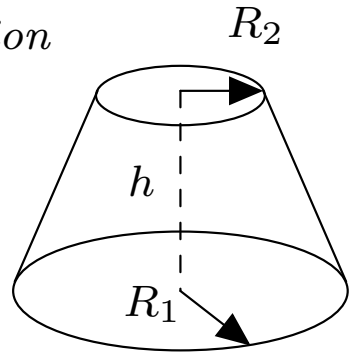
3. **Encodage** : *Choisir un langage et traduire l'algorithme*

Langage : Pyhon

```
from math import pi
R2 = float(input("Petit rayon ?  "))
R1 = float(input("Grand rayon ?  "))
H = float(input("Hauteur ?  "))
VOL = H*pi/3*(R1**2+R2**2+R1*R2)
print("Volume du cône tronqué :  ", VOL)
```

Mots clés

1. **Réflexion** : *trouver une solution au problème*
   (depuis internet) $\Longrightarrow$ volume $= h\frac{\pi}{3}(R_1^2 + R_2^2 + R_1 R_2)$
2. **Élaboration de l'algorithme** : *Expliquer comment obtenir la solution*
   - 2.1 Lire le petit rayon et le sauvegarder sous le nom R2
   - 2.2 Lire le plus grand rayon et le sauvegarder sous le nom R1
   - 2.3 Lire la hauteur et la sauvegarder sous le nom H
   - 2.4 Calcul du volume et sauvegarde sous le sous VOL
   - 2.5 Écrire le résultat à l'écran



3. **Encodage** : *Choisir un langage et traduire l'algorithme*

Langage : Pyhon

```
from  math import  pi
R2 = float(input("Petit rayon ?  "))
R1 = float(input("Grand rayon ?  "))
H = float(input("Hauteur ?  "))
VOL = H*pi/3*(R1**2+R2**2+R1*R2)
print("Volume du cône tronqué :  ", VOL)
```

4. *(**Traduction** dans le langage du processeur : langage binaire)*

```
11010110101001011010101010111100111101011001010101010001110000100111
10001101010101000001010000101000101010100010010111011000110001001011O
10100100101010010010101010010101010101001001000010101010110100111O...
```

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire
- **Langage assembleur (*intermediate level*) :** quelques commandes basiques

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire
- **Langage assembleur (*intermediate level*) :** quelques commandes basiques
- **Langage haut niveau (*human level*) :** très proche de l'anglais parlé

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire
- **Langage assembleur (*intermediate level*) :** quelques commandes basiques
- **Langage haut niveau (*human level*) :** très proche de l'anglais parlé

Un outil de traduction est nécessaire !

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire
- **Langage assembleur (*intermediate level*) :** quelques commandes basiques
- **Langage haut niveau (*human level*) :** très proche de l'anglais parlé

Un outil de traduction est nécessaire !

Deux types de traducteurs

- **Compilateur**: le programme est traduit entièrement

1. **Analyse lexicologique :** vocabulaire et orthographe corrects ?
2. **Analyse syntaxique :** grammaire correcte ?
3. **Analyse sémantique :** signification de chaque instruction ?
4. **Création d'un objet optimal :** traduction optimisée du code en binaire

Trois différents niveaux de programmation

- **Langage machine (*processor level*) :** le plus bas niveau, tout est binaire
- **Langage assembleur (*intermediate level*) :** quelques commandes basiques
- **Langage haut niveau (*human level*) :** très proche de l'anglais parlé

Un outil de traduction est nécessaire !

Deux types de traducteurs

- **Compilateur**: le programme est traduit entièrement

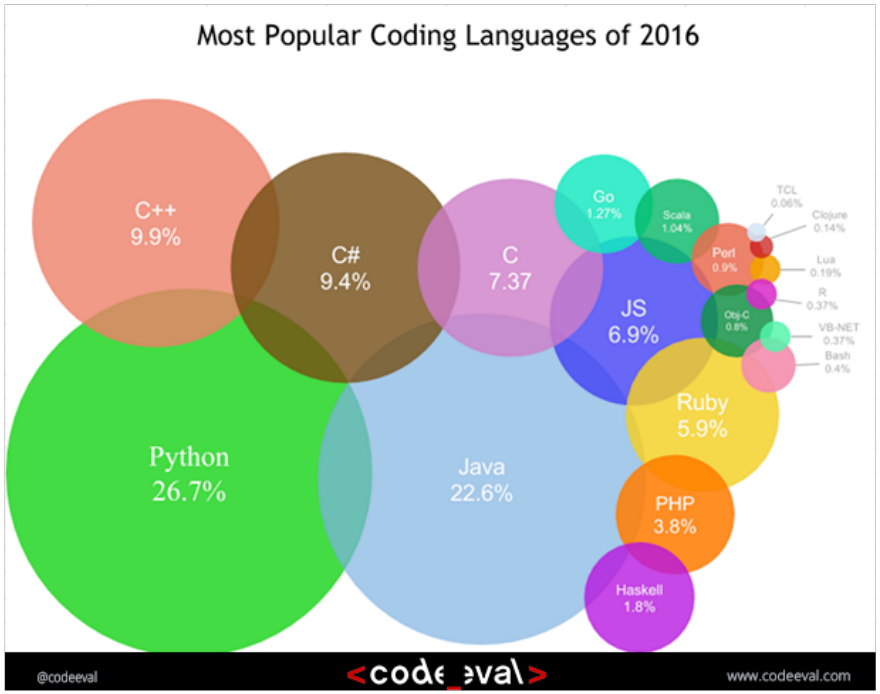
1. **Analyse lexicologique :** vocabulaire et orthographe corrects ?
2. **Analyse syntaxique :** grammaire correcte ?
3. **Analyse sémantique :** signification de chaque instruction ?
4. **Création d'un objet optimal :** traduction optimisée du code en binaire

- **Interpréteur**: Instructions lues, traduites, exécutées **une à une immédiatement**

La langage Python

C'est quoi Python ?

- Langage de programmation haut niveau crée par **Guido Van Rossum** en 1991.
- Langage libre et *open source*
- Langage interprété orienté objet
- Langage le plus populaire depuis 2010 !



Pourquoi l'utiliser ? Pourquoi est il si populaire ?

- Libre et *open source* : énorme communauté
- Interprété et orienté objet : grande flexibilité
- Généraliste : grande variété d'applications
- Facile à lire : développement simplifié, coûts de maintenance réduits
- Syntaxe simple : apprentissage rapide et productivité d'écriture de code
- Support de modules et packages : modularité et réutilisation de code
- Esthétique, et souvent plus compact que les autre langages
- Standardisé sur différentes plateformes (Windows / MacOS / Linux)
- Standard en calcul scientifique
- Extrêmement populaire dans le monde académique et dans les *tech companies*

"Python has been an important part of Google since the beginning, and remains so as the system grows and evolves. Today dozens of Google engineers use Python, and we're looking for more people with skills in this language.", `Peter Norvig, director of search quality at Google, Inc.`

"Python is fast enough for our site and allows us to produce maintainable features in record times, with a minimum of developers,", `Cuong Do, Software Architect, YouTube.com`.

"Programmers fall in love with Python because of the increased productivity it provides.", *Extract from Python executive summary*.

Hmmm... Et donc ?
Les désavantages ?

- Faible développement mobile : peu d'applications smartphone développées en Python
- Langage interprété : codes intensifs plus lents qu'en langage compilé

... Mais,

- De nombreux packages ont été optimisés et s'exécutent à la vitesse de C
- De nombreux packages permettent de compiler du code Python *just in time* (JIT) ou de précompiler du code
- Pour les scientifiques, **NumPy/SciPy** fournissent un environnement comparable et probablement plus compétitif que des application commerciales coûteuses telles que MatLab™

The easy way : Charm the snake

The snake charmer by Zachsmithson

Anaconda – Python distribution

- Large-scale data processing
- Predictive analytics
- Scientific computing
- Easy-to-use package manager `conda`

Spyder – **S**cientific **PY**thon **D**evelopment **E**nvi**R**onments

- Interactive environment
- Edition, testing, debugging, introspection
- Matlab-Like interface

Jupyter – Open-source notebook

- Web application
- Create and share documents
- Handle live code, text and LaTeX equations

The hard way : Ride the snake

The snake rider by Czarine

I'm a geek, I wanna control all my Python environment !

- Start from scratch : download and install last Python version on `www.python.org`
- Download and install manually a package manager like `pip`
- Update, upgrade, install whatever package you need by using

```
» pip install some-package
» pip uninstall some-package
```

- Use your favorite text editor (*vim*, ~~*emacs*~~ ...) or IDE (*vim*, *spyder*, *PyCharm*, ...)
- And then "***Qwant***[1] *is your friend" !*

[1] Great alternative to Google ! No trackers, Respect of privacy !

Python 2 Vs. Python 3

Python 2.7

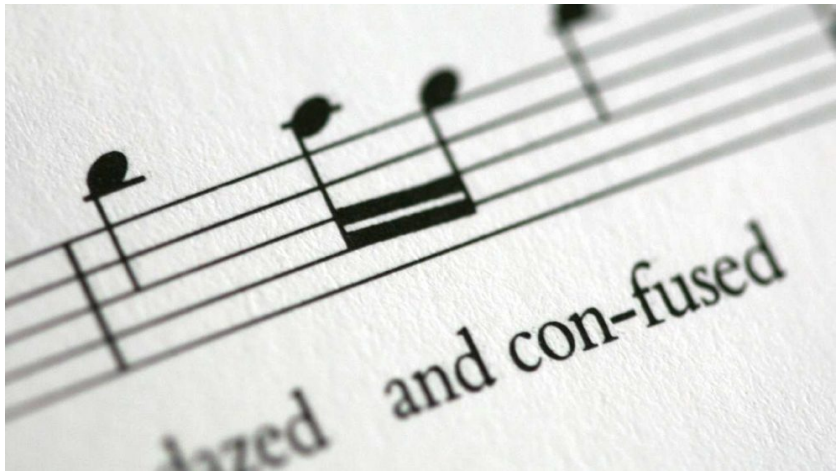
- Last major release of 2.x series
- Supported until at least 2020
- Used for a long time by scientist even if Python 3 existed

Python 3.x

- Last version is Python 3.6.1 (03.21.17)
- Broke the backward compatibility
- Intensive development in the last years
- `six` and `python-future` packages for Python 2/3 support

Today, there is no reason not to work with Python 3.x !
Especially for beginners !

A brief presentation



Dazed and confused by Led Zeppelin

Introduction à l'interpréteur Python

- Interactive interface to Python

```
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
» 3*2
6
```

- Python interpreter evaluates inputs
- Python prompts with "»"
- To exit Python : CTRL-D or `quit()`

Introduction à Ipython

- Interactive Python !
- Advanced completion
- Magic commands (with % : `ls`, `cd`, `who`, `whos`, `clear`, `timeit`, `run`, ...)

Introduction à Jupyter Notebook

- Application web
- Facilite la création et partage de documents
- Gère du code *live*, du texte et des équations LaTeX

Important : Le lancement de Jupyter s'accompagne du lancement d'un kernel IPython localement (fenêtre de commande ms-dos sous windows). Si vous fermez cette fenêtre, Jupyter ne fonctionnera plus correctement

Introduction à Spyder

- Environnent interactif de travail
- Édition, test, debugging, introspection

The beginnings

Photography by J.J.

Structure de lignes sous Python

- **Ligne physique :** séquence de caractères se terminant par un retour charriot
- **Ligne logique:** ensemble de plusieurs *lignes physiques* suivant les règles de :
  - ➤ ***Raccordement explicite :***
    - ➥ lignes physiques assemblées en ligne logique avec "\"
    - ➥ une fin de ligne avec "\" ne peut pas porter de commentaire
  - ➤ ***Raccordement implicite :***
    - ➥ expressions entre `()`, `{}`, ou `[]` divisibles sans utiliser "\"
    - ➥ les lignes assemblées implicitement peuvent porter un commentaire

Commentaires sous Python

- Tout ce qui suit le caractère "#" est ignoré par l'interpréteur
- Termine une ligne logique sauf si la règle *implicite* de raccordement est invoquée

Illustration

```
# Raccordement implicite :
a = [1, 2, 3,           # Commentaire possible à la fin...
           4, 5, 6]     # ...de chaque ligne physique

# Raccordement explicite :
b = [1, 2, 3, \         <- Commentaire interdit ici !
           4, 5, 6]     # Seulement autorisé à la fin de la ligne logique
```

Fonctions et instructions natives

☞ **68 fonctions natives :**

abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

☞ **31 instructions et 2 booléens :**

True	assert	del	for	in	or	while
False	break	elif	from	is	pass	with
None	class	else	global	lambda	raise	yield
and	continue	except	if	nonlocal	return	
as	def	finally	import	not	try	

Et c'est tout ?