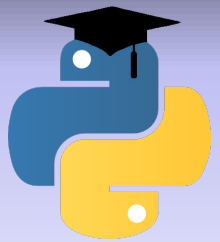
Langage et algorithmique

Abdelmalek TOUMI

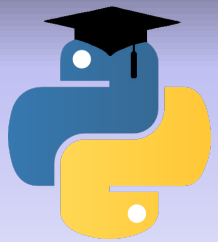
toumiab@ensta-bretagne.fr

8 septembre 2016

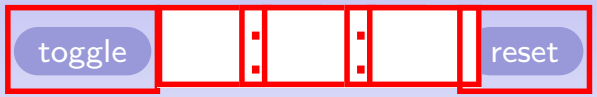
ENSTA Bretagne



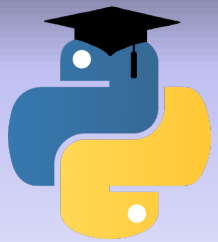
Sommaire



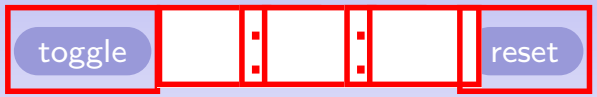
Déroulement



- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)

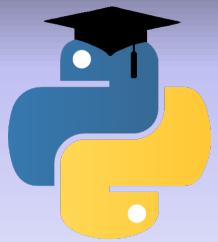


Déroulement

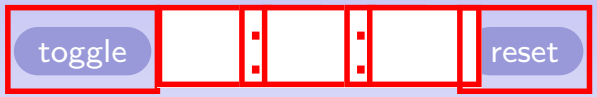


- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)

- Programmation impérative et objet
- Notions d'architecture logicielle



Déroulement

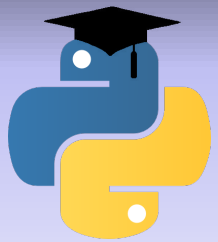


- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)

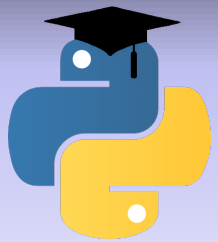
- Programmation impérative et objet
- Notions d'architecture logicielle

- Le cours est découpé : CM (30min) + Pause (10min) + CM (35 min)
- Utilisation des cartons (vert/rouge) pour les exercices Q/R

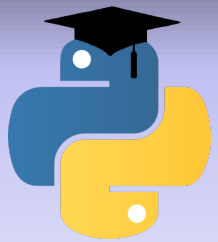
Plateforme : `https://moodle.ensta-bretagne.fr`



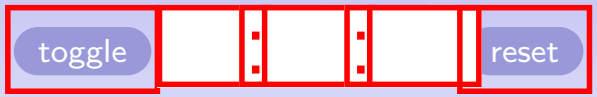
Sommaire



Sommaire

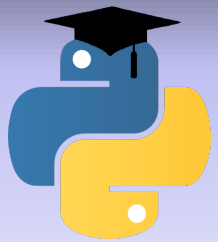


Histoire des langages

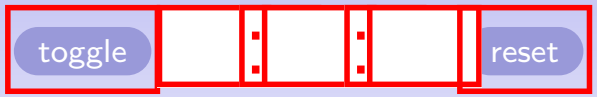


- Début : langage machine
- Assembleur en 1950
- Fortran en 1954
- Matlab 1970
- C en 1971
- Python en 1994 (première idée en 1990)
- Java en 1995

Historique : http ://www.levenez.com

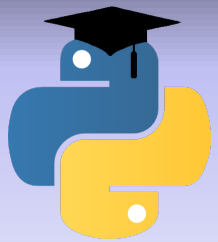


Histoire des langages

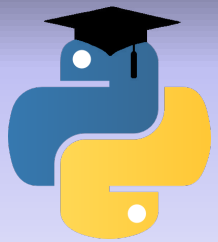


- Début : langage machine
- Assembleur en 1950
- Fortran en 1954
- Matlab 1970
- C en 1971
- Python en 1994 (première idée en 1990)
- Java en 1995

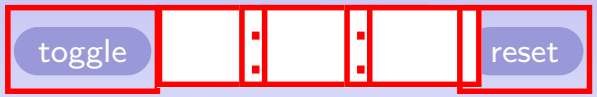
Historique : http ://www.levenez.com
Langage choisi : Python (version 3.5)



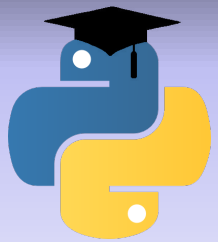
Sommaire



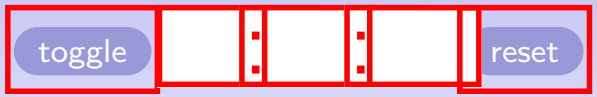
Langage Python



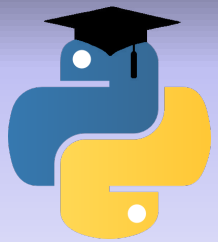
- Python est interprété



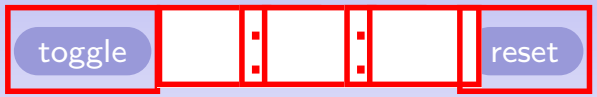
Langage Python



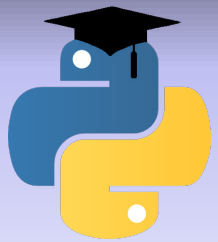
- Python est interprété
- Python est orienté objet et permet une programmation impérative



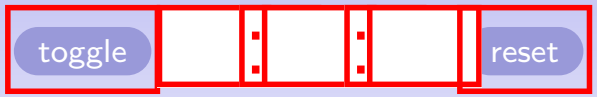
Langage Python



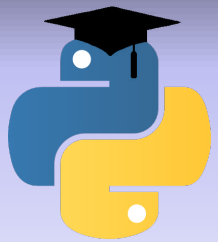
- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé



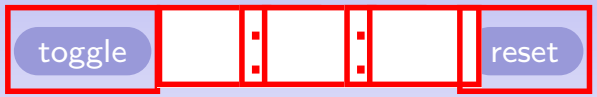
Langage Python



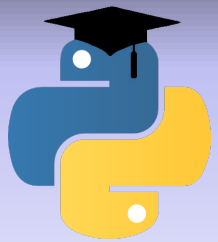
- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement



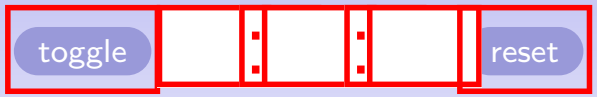
Langage Python



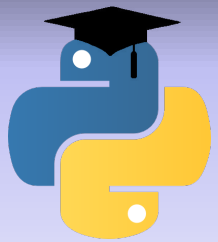
- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire



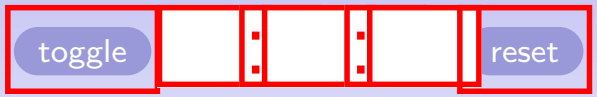
Langage Python



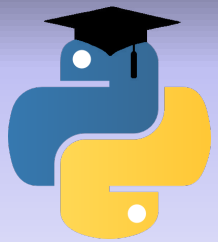
- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche



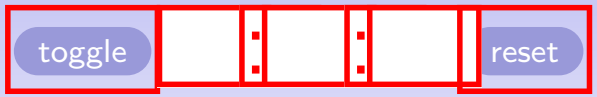
Langage Python



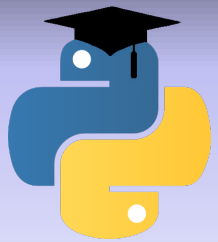
- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages



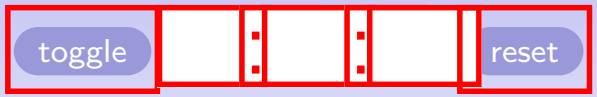
Langage Python



- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages
- Python est indépendant de toute plate-forme

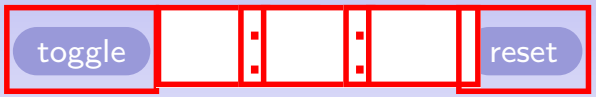


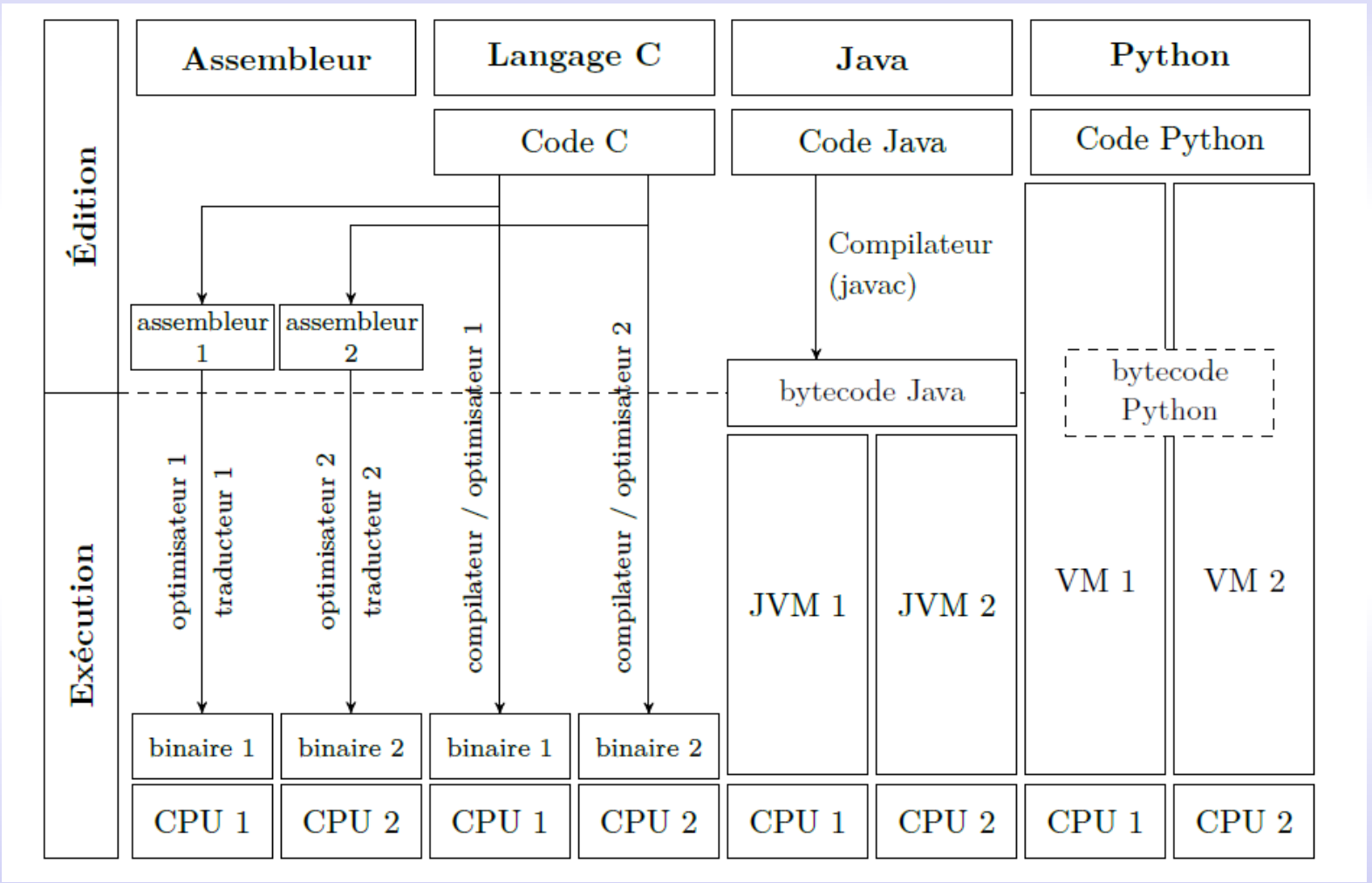
Langage Python



- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages
- Python est indépendant de toute plate-forme
- Python est agréable et facile à lire

Langage Python





Exécution d'un programme Python

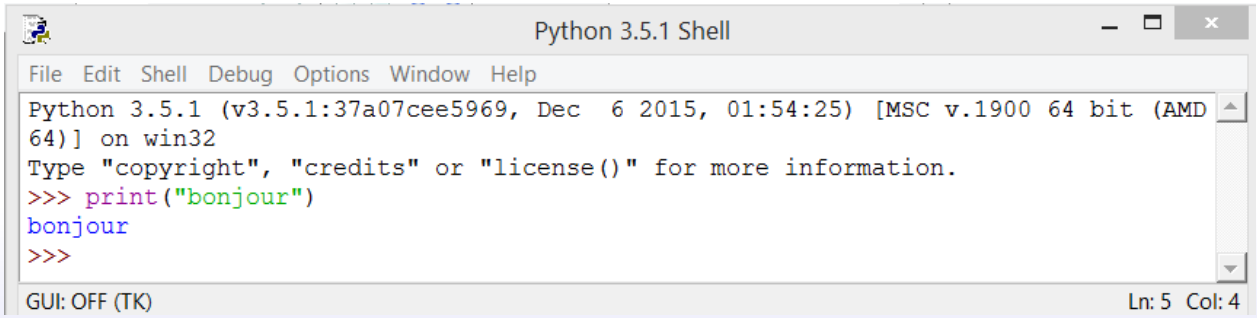
Remarque (Un programme Python)

Un programme est une suite d'instructions Python qui peut être exécuté de différentes manières :

- **En mode interactif**
- **En script**

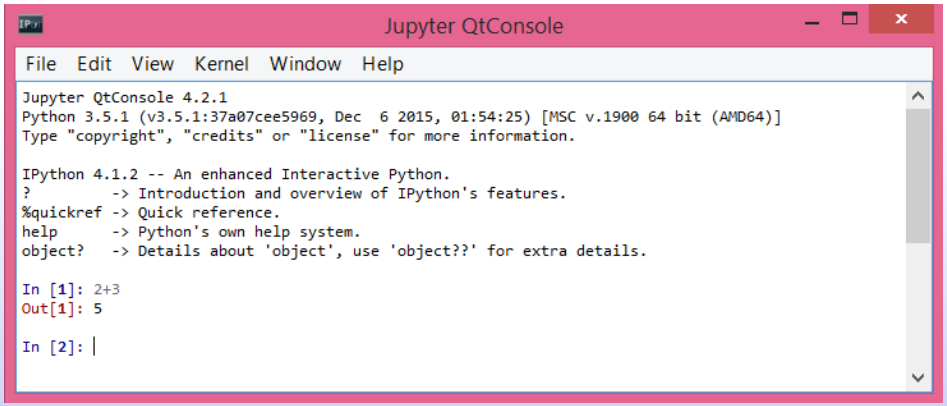
Exécution d'un programme Python

- **En mode interactif** :
  - avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
    - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
    - Les instructions sont exécutées immédiatement
    - Le résultat d'une instruction est affiché sur le terminal



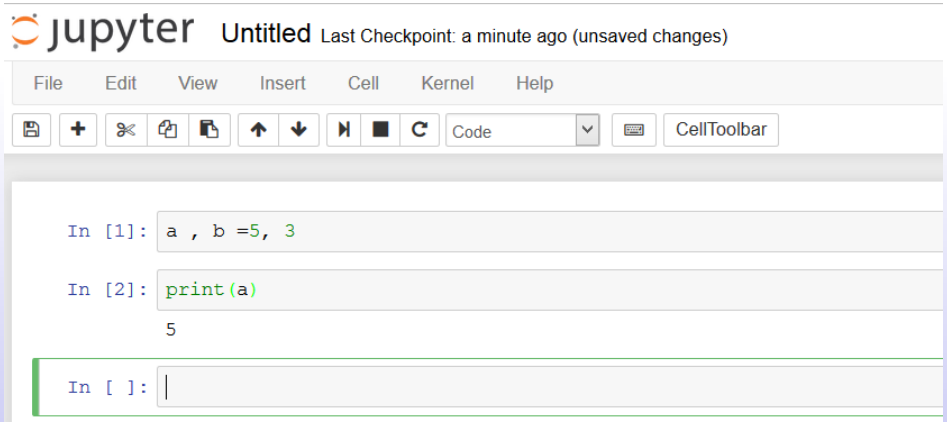
Exécution d'un programme Python

- **En mode interactif** :
  - avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
    - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
    - Les instructions sont exécutées immédiatement
    - Le résultat d'une instruction est affiché sur le terminal
  - avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**



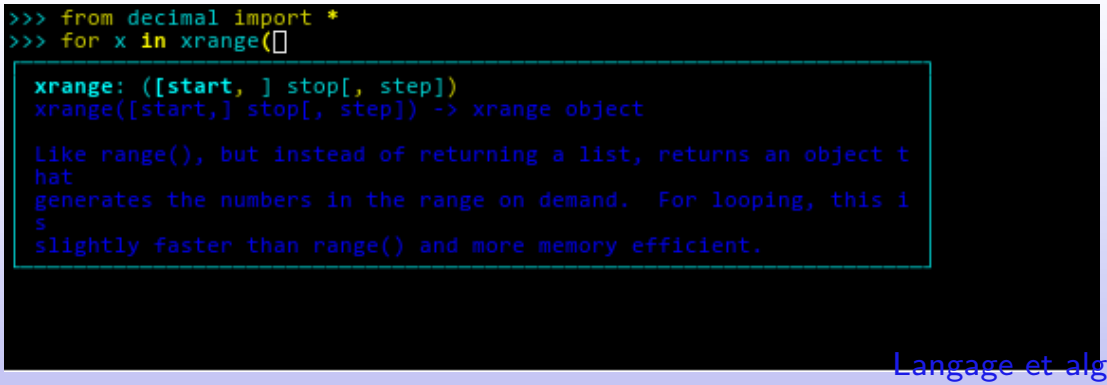
Exécution d'un programme Python

- **En mode interactif** :
  - avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
    - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
    - Les instructions sont exécutées immédiatement
    - Le résultat d'une instruction est affiché sur le terminal
  - avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**



Exécution d'un programme Python

- **En mode interactif** :
  - avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
    - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
    - Les instructions sont exécutées immédiatement
    - Le résultat d'une instruction est affiché sur le terminal
  - avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**
  - avec **bpython** : analogue à ipython en plus léger et plus commode, il propose une coloration syntaxique sur la ligne de commande

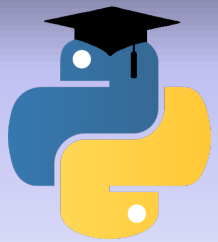


Exécution d'un programme Python

- **En mode script** (Programming mode) :
  - Écriture du programme Python (éditeur de texte) dans un fichier portant l'extension **.py**

Exécution d'un programme Python

- **En mode script** (Programming mode) :
  - Écriture du programme Python (éditeur de texte) dans un fichier portant l'extension **.py**
  - Exécution
  - `$ python monprogramme.py # pour windows`
  - `$ python3 monprogramme.py # pour linux`



Sommaire

Installation d'un environnement Python

- Sous Windows, Python n'est pas installé par défaut.

Installation d'un environnement Python

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés

Installation d'un environnement Python

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Pyhon v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7

Installation d'un environnement Python

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Pyhon v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7
- Installation :
  - Windows : **WinPython 3.5.1.3** distribution spécifique à Windows et très facile à mettre en œuvre. Il intègre notamment : IPython, Spyder, NumPy, SciPy, MatplotLib, Pandas, SymPy, PIP...
  - $+ \longrightarrow$ `http://www.lfd.uci.edu/~gohlke/pythonlibs/`

Installation d'un environnement Python

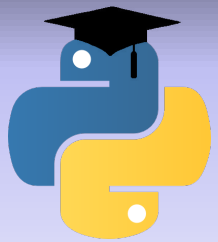
- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Pyhon v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7
- Installation :
  - Windows : **WinPython 3.5.1.3** distribution spécifique à Windows et très facile à mettre en œuvre. Il intègre notamment : IPython, Spyder, NumPy, SciPy, MatplotLib, Pandas, SymPy, PIP...
  - $+ \longrightarrow$ `http://www.lfd.uci.edu/~gohlke/pythonlibs/`
  - GNU/Linux Ubuntu $\geq$ 16.04 : installation standard via les dépôts officiels Ubuntu de Canonical
  - Mac OS X : **Anaconda** distribution Python multiplateforme très répandue dans les milieux scientifiques

Environnement de développement

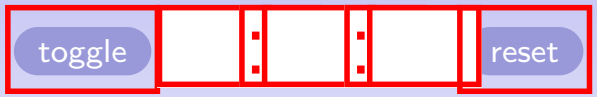
- bloc-note, notepad++, vim, emacs ;
- IDE (Integrated Development Environment) : `https://wiki.python.org/moin/IntegratedDevelopmentEnvironments`
  - Netbeans (Sun) avec Python/jPython
  - Eclipse (IBM) avec le plugin PyDev
  - **Spyder**
  - Komodo IDE
  - LiCipse avec PyDev
  - pyCharm
  - ....

Python et le calcul scientifique

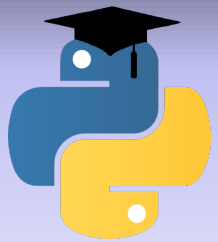
- rapide à apprendre (mais long à maîtriser) ;
- alternative (très) sérieuse à Matlab, Scilab, Octave ;
- bibliothèques de calcul et de visualisation très complète et performante ;
- parallélisation possible facilement (efficacité, etc.) ;
- communauté nombreuse et active ;
- multi-plateformes (scripts python sur Abaqus sur vos machines – Windows – et sur le cluster – Linux)



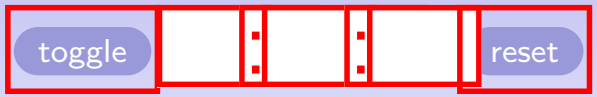
Les autres librairies



- numexpr :https://code.google.com/p/numexpr/
- NLopt : http://ab-initio.mit.edu/wiki/index.php/NLopt
- scikits : https://scikits.appspot.com/
- pyQt4 : http://www.riverbankcomputing.co.uk/software/pyqt/
- PIL : http://www.pythonware.com/products/pil/
- sympy : http://sympy.org/fr/index.html
- guiqwt/guidata : https://code.google.com/p/guiqwt,guidata
- pyqtgraph : http://www.pyqtgraph.org/
- SfePy : http://sfepy.org/doc-devel/index.html
- FEniCS : http://fenicsproject.org/

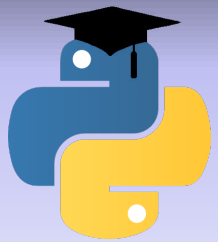


Les autres librairies

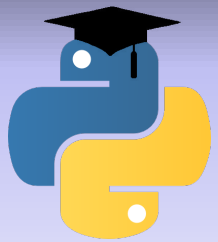


- numexpr :https://code.google.com/p/numexpr/
- NLopt : http://ab-initio.mit.edu/wiki/index.php/NLopt
- scikits : https://scikits.appspot.com/
- pyQt4 : http://www.riverbankcomputing.co.uk/software/pyqt/
- PIL : http://www.pythonware.com/products/pil/
- sympy : http://sympy.org/fr/index.html
- guiqwt/guidata : https://code.google.com/p/guiqwt,guidata
- pyqtgraph : http://www.pyqtgraph.org/
- SfePy : http://sfepy.org/doc-devel/index.html
- FEniCS : http://fenicsproject.org/

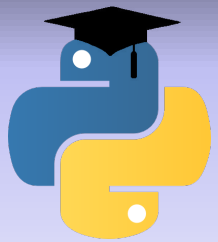
WinPython : http://www.lfd.uci.edu/~gohlke/pythonlibs/



Sommaire



Sommaire

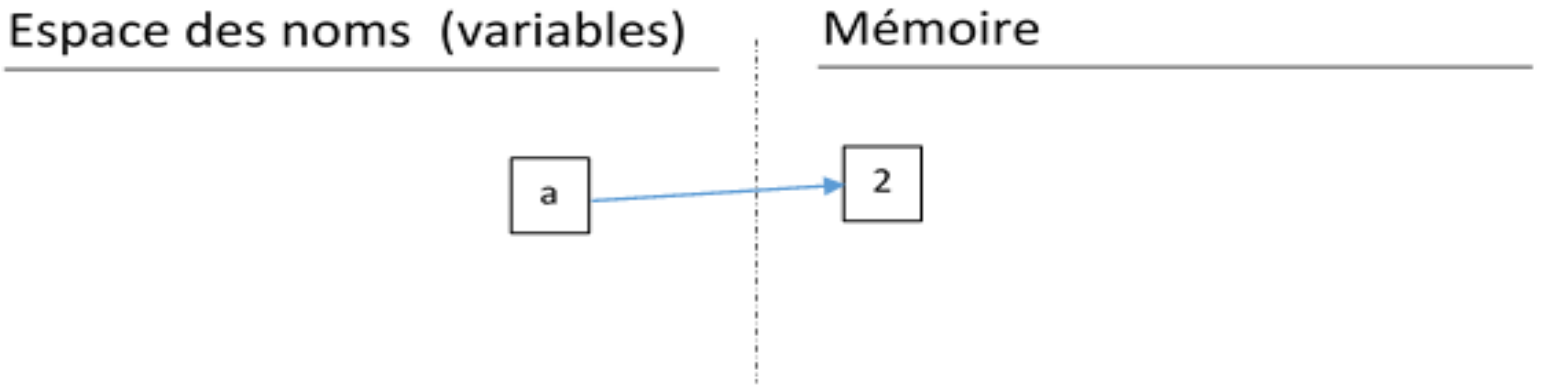


Variables et données

- Tous les éléments manipulés dans un programme python sont des objets (données).
- Une variable n'est que la référence à l'objet.
- Toute variable référence un objet d'un type bien connu (entier, réel, chaîne de caractères, ...)
- Un emplacement mémoire (adresse binaire) contient des données (binaire)
- Données de taille variable (selon le type défini : entier, réel, ...)
- Python gère ses ressources (mémoire, descripteurs de fichiers...) sans intervention du programmeur, par un mécanisme de comptage de références

Variables et données

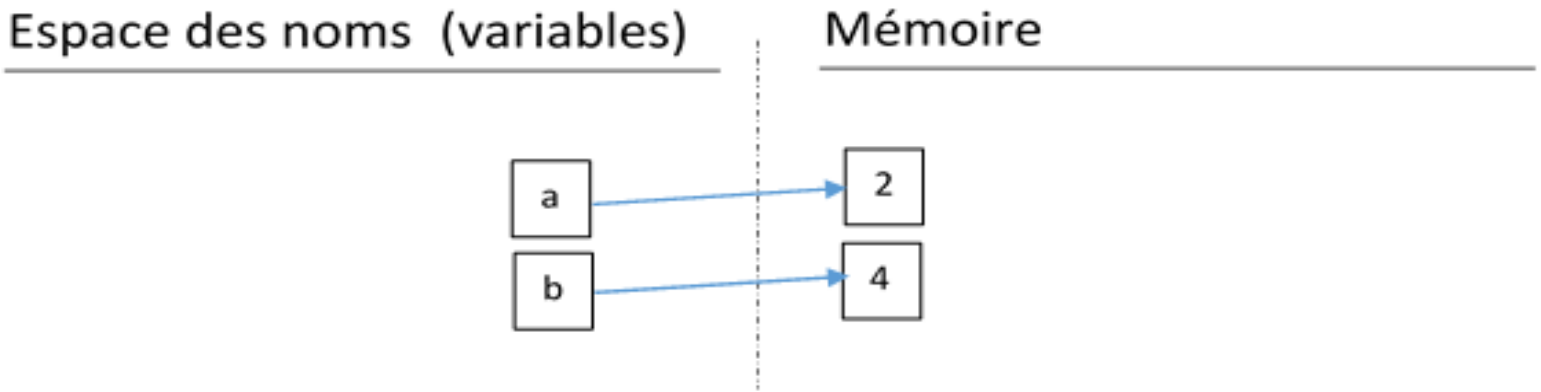
- Exemple :




```
>>> a = 2
```

Variables et données

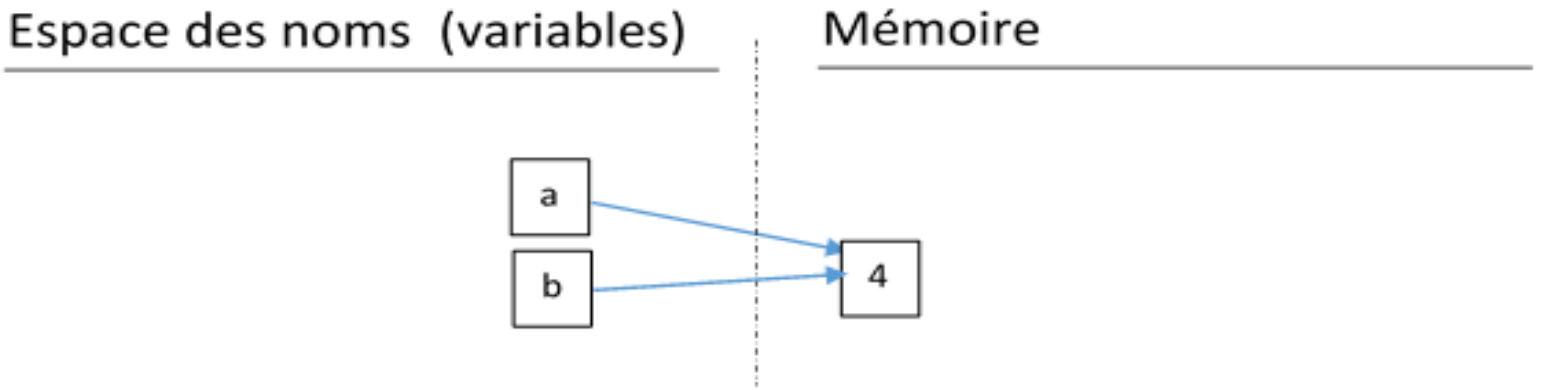
- Exemple :




```
>>> a = 2
>>> b = 4
```

Variables et données

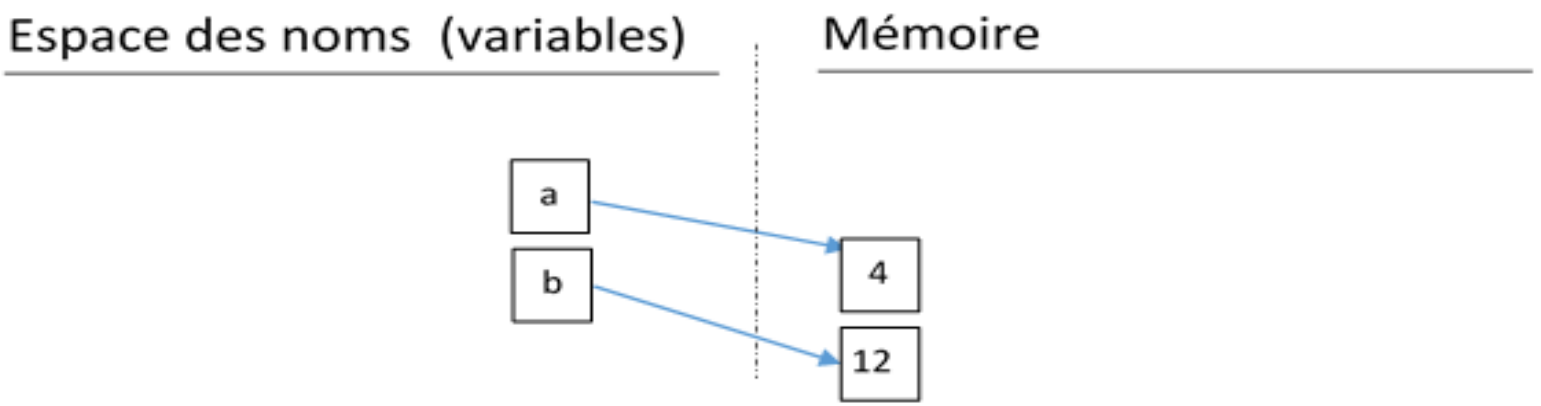
- Exemple :



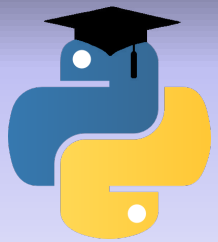

```
>>> a = 2
>>> b = 4
>>> a = b
```

Variables et données

- Exemple :




```
>>> a = 2
>>> b = 4
>>> a = b
>>> b = 12
```

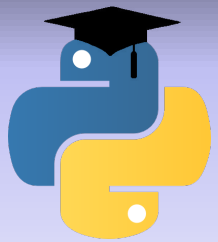


Variables en Python

- Il n'est pas nécessaire de déclarer une variable avant de l'utiliser.
- La déclaration d'une variable et son initialisation se fait en même temps.
- Une variable peut changer de type au cours de l'exécution d'un programme.
- Noms de variables sensibles à la casse

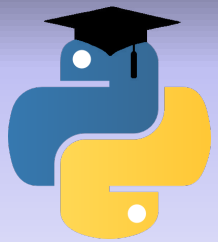
Exemple

```
>>> a = 42
>>> b = 33.5
>>> B = "une chaîne de caractères"
>>> a= a + 1.5
>>> a
43.5
```



Variables en Python

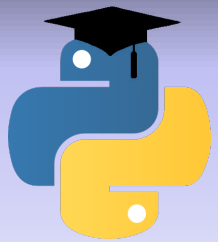
- Règles de nommage :
  - ne peut commencer par un chiffre ou un "_" ("_" est à réserver pour certaines situations)
  - peut contenir des chiffres, et "_"
  - ne peut pas contenir de séparateur
  - ne doit pas être un mot clé (33 mots clé en Python 3)
  - peut contenir des accents, mais déconseillé
- Choisir des noms parlants (rester raisonnable)



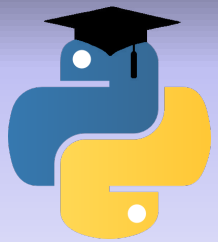
Variables en Python

- Règles de nommage :
  - ne peut commencer par un chiffre ou un “_” (”_” est à réserver pour certaines situations)
  - peut contenir des chiffres, et “_”
  - ne peut pas contenir de séparateur
  - ne doit pas être un mot clé (33 mots clé en Python 3)
  - peut contenir des accents, mais déconseillé
- Choisir des noms parlants (rester raisonnable)

```
surelyReachableObjectsWhichHaveToBeMarkedAsSuch
```



Sommaire

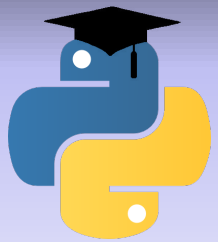


Fonctions utiles

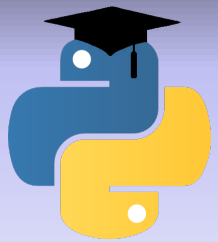
- Quelques fonctions utiles intégrées à Python :
  - la fonction **help()** donne l'accès à l'aide en ligne. **help(obj)** permet de consulter l'aide sur l'objet obj.
  - la fonction **dir()** la liste des noms accessibles dans la portée actuelle. **dir(obj)** retourne la liste de tous les attributs valides de l'objet obj.
  - la fonction **type(objet)** retourne le type de l'objet obj.
  - la fonction **print(objet)** permet d'afficher le contenu de l'objet obj.

Remarque

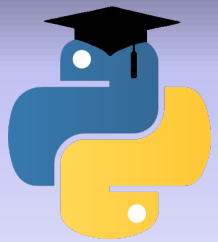
On invoque une fonction en utilisant l'opérateur **()** sur l'objet fonction et en plaçant les paramètres entre les parenthèses.



Sommaire

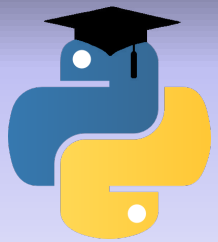


Sommaire



Typage des variables

- Python : langage typé dynamiquement



Typage des variables

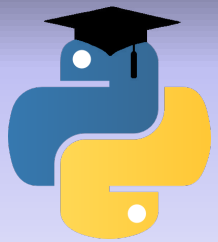
- Python : langage typé dynamiquement

$\Rightarrow$ typage lors de l'affectation

- Une variable peut changer de type au cours de l'exécution

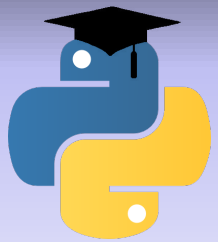
Remarque

- Pour connaître le type d'une variable ou expression, on peut utiliser la fonction **type()**
- La fonction **print()** permet d'afficher la valeur d'une variable



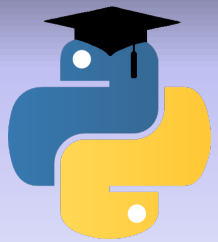
Typage des variables

- **Les types intégrés de Python**



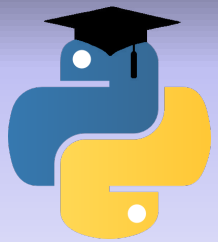
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**



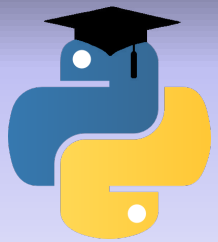
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**



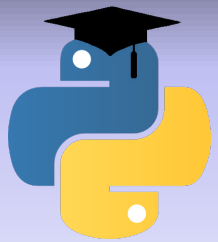
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**
  - Chaînes de caractères : type **str**



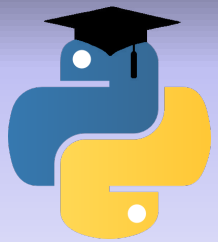
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**
  - Chaînes de caractères : type **str**
  - Listes : type **list**



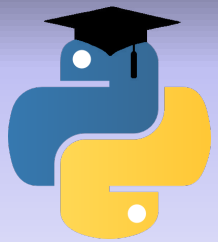
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**
  - Chaînes de caractères : type **str**
  - Listes : type **list**
  - Dictionnaires : type **dict**



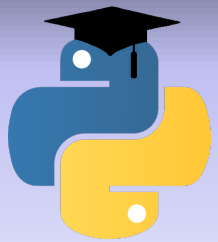
Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**
  - Chaînes de caractères : type **str**
  - Listes : type **list**
  - Dictionnaires : type **dict**
  - Tuples : type **tuple**

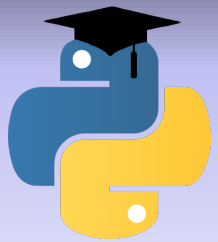


Typage des variables

- **Les types intégrés de Python**
  - Entiers signés (complément à deux) : le type **int**
  - Réels (IEEE 754) : le type **float** en double précision (64 bits)
  - Complexes : le type **complex**
  - Booléens : le type **bool**
  - Chaînes de caractères : type **str**
  - Listes : type **list**
  - Dictionnaires : type **dict**
  - Tuples : type **tuple**
  - Ensembles modifiables et immuables : le type **set** et le type **frozenset**



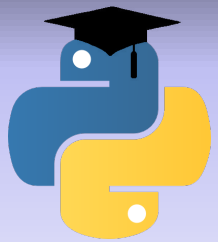
Sommaire



Contraintes

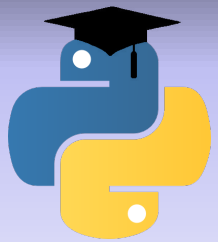
- Représenter des entiers relatifs
- Déterminer si le nombre est positif ou négatif
- Conserver les propriétés de l'addition

$\Rightarrow$ complément à deux



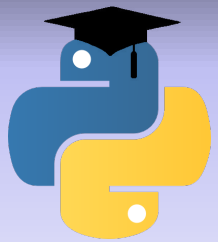
Complément à deux

- Bit de poids fort : signe (0$\rightsquigarrow$positif ou nul, 1$\rightsquigarrow$ négatif)



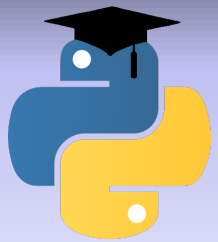
Complément à deux

- Bit de poids fort : signe (0$\rightsquigarrow$positif ou nul, 1$\rightsquigarrow$ négatif)
- Sur n bits : plus grand entier $2^{n-1}-1$, plus petit -2^{n-1}



Complément à deux

- Bit de poids fort : signe (0$\rightsquigarrow$positif ou nul, 1$\rightsquigarrow$ négatif)
- Sur n bits : plus grand entier $2^{n-1}-1$, plus petit -2^{n-1}
- Représentation d'un nombre négatif x :
  - considérer $-x$
  - inverser chaque bit
  - ajouter 1



Complément à deux

- Bit de poids fort : signe (0$\rightsquigarrow$positif ou nul, 1$\rightsquigarrow$ négatif)
- Sur n bits : plus grand entier $2^{n-1} - 1$, plus petit -2^{n-1}
- Représentation d'un nombre négatif x :
  - considérer $-x$
  - inverser chaque bit
  - ajouter 1
- Remarque : soit x un entier et $\tilde{x}$ son complémentaire. $x + \tilde{x} = 0$

Exemple (Représentation de -5 en complément à deux)

On désire coder la valeur -5 sur 8 bits. Il suffit :

- d'écrire 5 en binaire : 00000101 ;
- de complémenter à 1 : 11111010 ;
- d'ajouter 1 : 11111011 ;
- la représentation binaire de -5 sur 8 bits est 11111011.

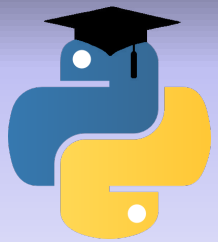
Exemple (Représentation de -5 en complément à deux)

On désire coder la valeur -5 sur 8 bits. Il suffit :

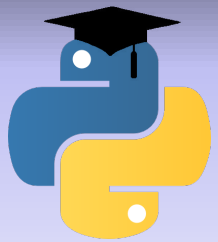
- d'écrire 5 en binaire : 00000101 ;
- de complémenter à 1 : 11111010 ;
- d'ajouter 1 : 11111011 ;
- la représentation binaire de -5 sur 8 bits est 11111011.

Remarque

- le bit de poids fort est 1 : un nombre négatif
- 5 + -5 (00000101 + 11111011) donne 0 (retenue de 1)



Sommaire

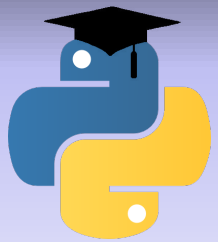


Objectif

- Représenter des réels en binaire
- Nécessité d'approximer les nombres
- Codage du nombre sur 32 bits en simple précision (64 en double précision)
  - signe
  - valeur
  - exposant

$\Rightarrow$ norme IEEE754

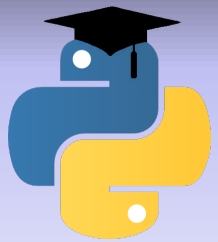
Ex : $5.25 \rightsquigarrow 1.0101 * 2^2$



norme IEEE754

Représentation (poids fort vers poids faible) :

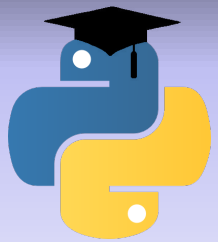
- 1 bit de signe



norme IEEE754

Représentation (poids fort vers poids faible) :

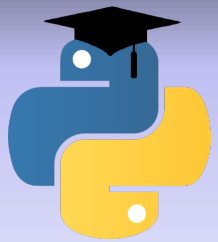
- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)



norme IEEE754

Représentation (poids fort vers poids faible) :

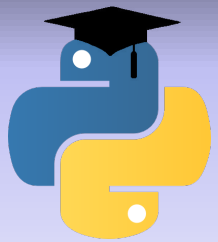
- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)
- 23 bits de mantisse (52 en double précision)



norme IEEE754

Représentation (poids fort vers poids faible) :

- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)
- 23 bits de mantisse (52 en double précision)
- En simple précision :
  [⇒]seeeeeeeemmmmmmmmmmmmmmmmmmmmmm

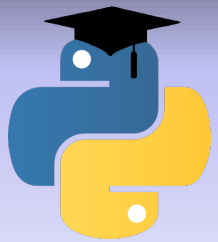


norme IEEE754

Représentation (poids fort vers poids faible) :

- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)
- 23 bits de mantisse (52 en double précision)
- En simple précision :
  [⇒]seeeeeeeemmmmmmmmmmmmmmmmmmmmmmm

$= (-1)^s \times (1.M) \times 2^{E-127}$

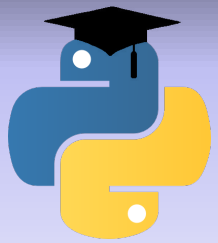


norme IEEE754

Représentation (poids fort vers poids faible) :

- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)
- 23 bits de mantisse (52 en double précision)
- En simple précision :
  [⇒]seeeeeeeemmmmmmmmmmmmmmmmmmmmmmm

$= (-1)^s \times (1.M) \times 2^{E-127}$



norme IEEE754

Représentation (poids fort vers poids faible) :

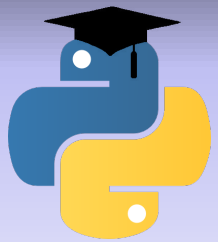
- 1 bit de signe
- 8 bits d'exposant pour la simple précision (11 pour la double précision)
- 23 bits de mantisse (52 en double précision)
- En simple précision :
  [⇒]seeeeeeeemmmmmmmmmmmmmmmmmmmmmmm

$= (-1)^s \times (1.M) \times 2^{E-127}$

Conditions sur les exposants :

- 00000000 interdit
- 11111111 $\rightsquigarrow$ NaN (Not a Number)

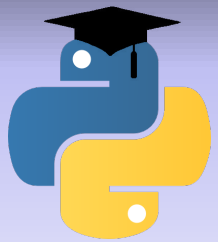
$\Rightarrow$ exposants de -126 à 127 (-1023 à 1024 pour la double précision)



Exemple

Exemple (Représentation de 525.5 en simple précision)

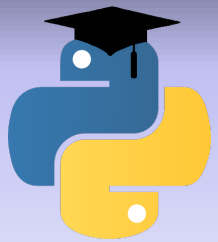
- $525.5 \xrightarrow{base2} 1000001101.1$



Exemple

Exemple (Représentation de 525.5 en simple précision)

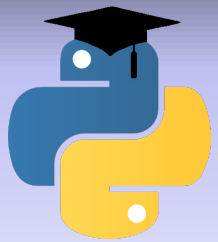
- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^{9}$



Exemple

Exemple (Représentation de 525.5 en simple précision)

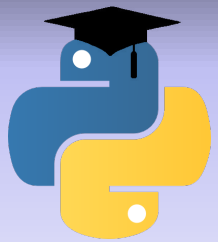
- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^9$
  - signe : 0



Exemple

Exemple (Représentation de 525.5 en simple précision)

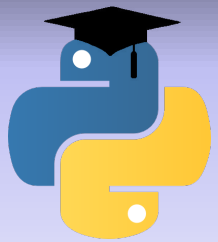
- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^{9}$
  - signe : 0
  - exposant : 127+9=136 $\rightsquigarrow$ 10001000



Exemple

Exemple (Représentation de 525.5 en simple précision)

- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^9$
  - signe : 0
  - exposant : 127+9=136 $\rightsquigarrow$ 10001000
  - mantisse : 0000011011

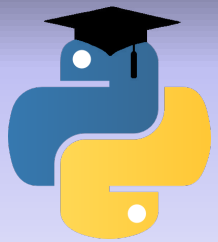


Exemple

Exemple (Représentation de 525.5 en simple précision)

- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^9$
  - signe : 0
  - exposant : 127+9=136 $\rightsquigarrow$ 10001000
  - mantisse : 0000011011

  $\Rightarrow$ 01000100000000110110000000000000



Exemple

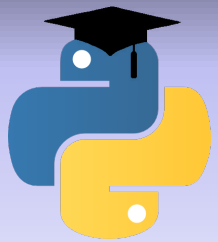
Exemple (Représentation de 525.5 en simple précision)

- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^9$
  - signe : 0
  - exposant : 127+9=136 $\rightsquigarrow$ 10001000
  - mantisse : 0000011011

  $\Rightarrow$ 01000100000000110110000000000000

Remarque

- Le type **float** en Python utilise une double précision (64 bits).



Exemple

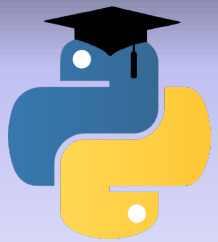
Exemple (Représentation de 525.5 en simple précision)

- $525.5 \xrightarrow{base2} 1000001101.1$
- $1000001101.1 = +1.0000011011 \times 2^9$
  - signe : 0
  - exposant : 127+9=136 $\rightsquigarrow$ 10001000
  - mantisse : 0000011011

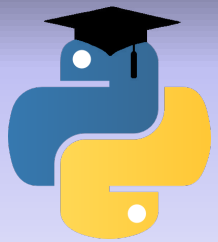
  $\Rightarrow$ 01000100000000110110000000000000

Remarque

- Le type **float** en Python utilise une double précision (64 bits).
- Les autres biothèques utilisées dans Python (ex. numpy) utilisent autres formats.



Sommaire

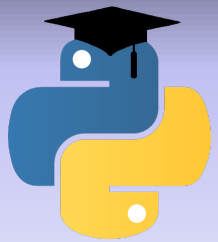


ASCII

- ASCII : American Standard Code for Information Interchange
- Standard sur 7 bits, étendu à 8 bits (ex : iso8859-1)

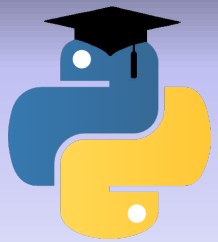
ASCII

	30	40	50	60	70	80	90	100	110	120
0 :		(	2	<	F	P	Z	d	n	x
1 :		)	3	=	G	Q	[	e	o	y
2 :		*	4	>	H	R	&	f	p	z
3 :	!	+	5	?	I	S	]	g	q	{
4 :	”	,	6	@	J	T	^	h	r	—
5 :	#	-	7	A	K	U	_	i	s	}
6 :	$	.	8	B	L	V	‘	j	t	∼
7 :	%	/	9	C	M	W	a	k	u	DEL
8 :	&	0	:	D	N	X	b	l	v	
9 :	’	1	;	E	O	Y	c	m	w	



ASCII

- ASCII : American Standard Code for Information Interchange
- Standard sur 7 bits, étendu à 8 bits (ex : iso8859-1)
- **Python** utilise la norme **Unicode** : représentation sur 16 bits

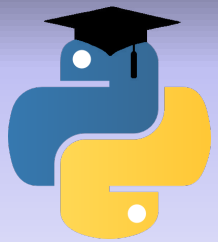


ASCII

- ASCII : American Standard Code for Information Interchange
- Standard sur 7 bits, étendu à 8 bits (ex : iso8859-1)
- **Python** utilise la norme **Unicode** : représentation sur 16 bits

Remarque (Le type caractère en Python)

- Contrairement à d'autre langage, il n'existe pas en Python un type spécifique pour un caractère.

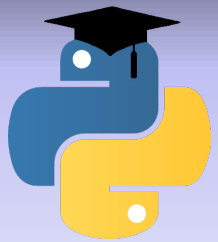


ASCII

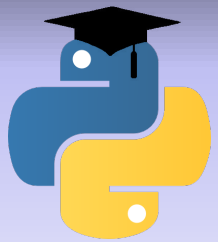
- ASCII : American Standard Code for Information Interchange
- Standard sur 7 bits, étendu à 8 bits (ex : iso8859-1)
- **Python** utilise la norme **Unicode** : représentation sur 16 bits

Remarque (Le type caractère en Python)

- Contrairement à d'autre langage, il n'existe pas en Python un type spécifique pour un caractère.
- Un caractère n'est rien qu'une chaîne de caractères (le type **str**) de longueur 1.



Sommaire



horloge