

INTRODUCTION AU LANGAGE python™

PYTHON EST ...

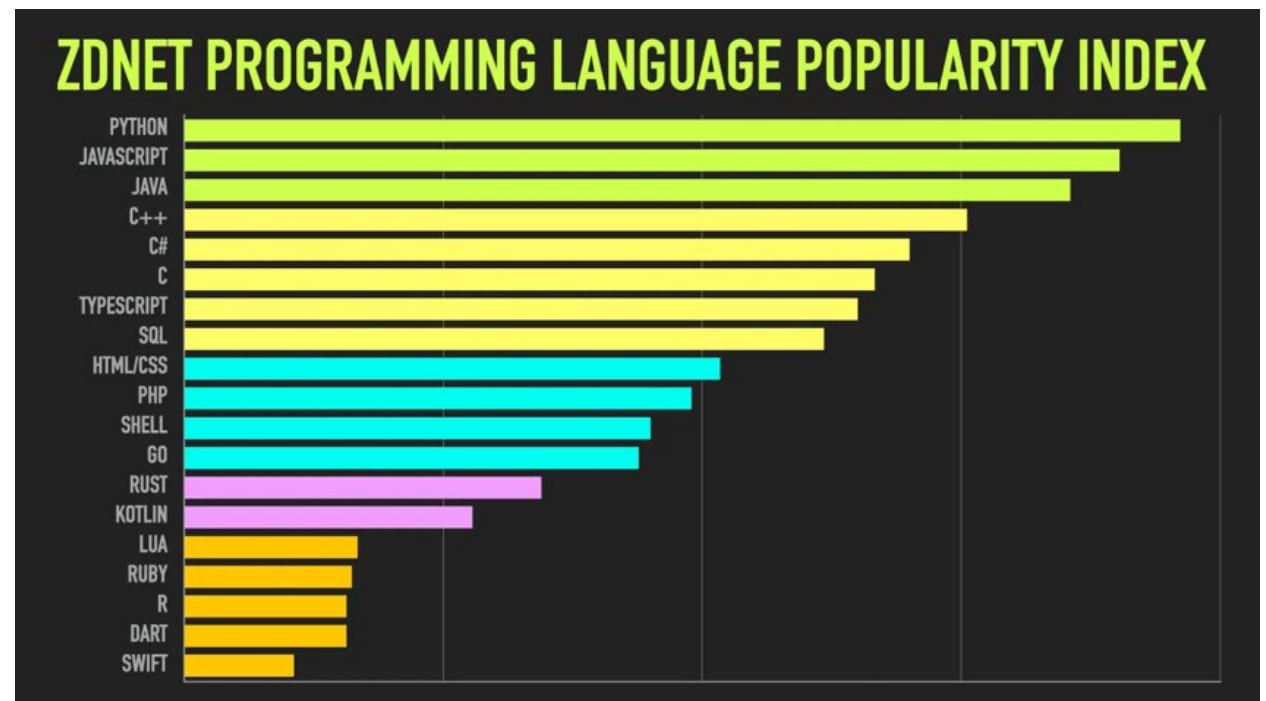
Simple

- Python est un langage simple, de haut-niveau et minimaliste par nature
- Sa nature proche du pseudo-code permet de mieux se concentrer sur le problème à résoudre plutôt que le langage

Facile à apprendre

Libre & Open source

Portable

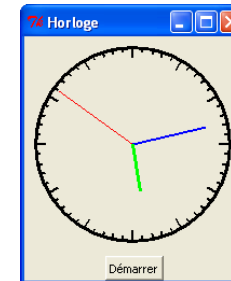
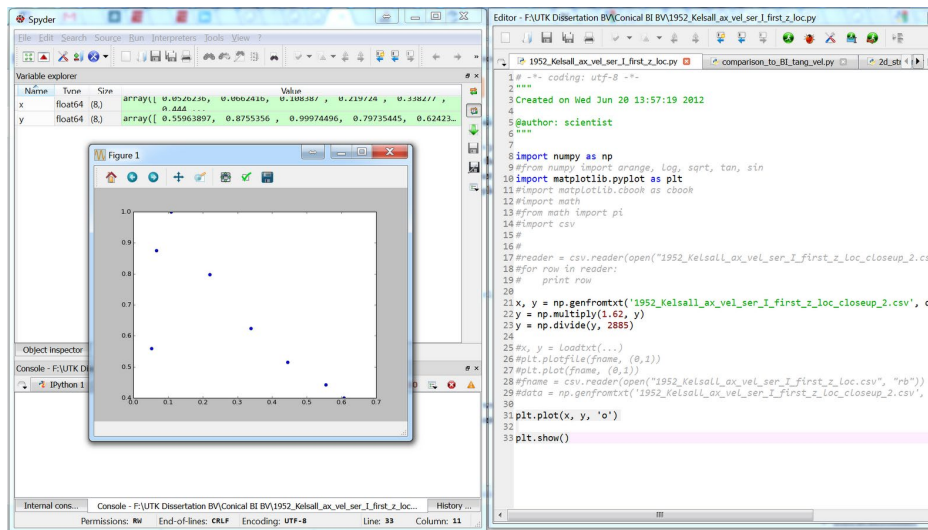


PYTHON EST ...

Un langage **interprété**

Extensible avec de nombreuses bibliothèques

- Réseau, image, son, cryptographie, calcul numérique – numpy, scipy, ... - etc. **Mais aussi** des bibliothèques graphiques - wxPython, tkinter, kivy, ...



HISTOIRE DE **Python**

Python a été conçu fin 1989 par Guido Van Rossum (code rendu public en 1991)

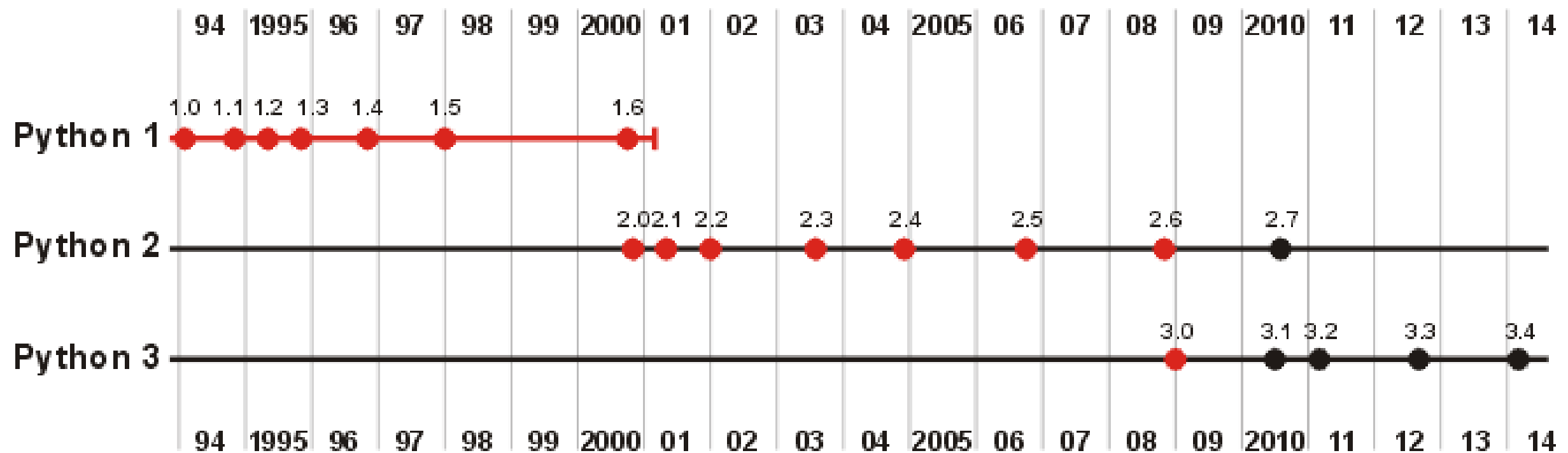
En Janvier 1994 sort Python 1.0. Avec la version 1.4, Python s'enrichit de nouvelles fonctionnalités dont notamment la gestion des nombres complexes.

Lance l'initiative "**Computer Programming for Everybody - CP4E** "
(financé par le DARPA)

- L'objectif est de rendre accessible la programmation au plus grand nombre notamment à ceux qui ne font pas de grandes études.



HISTOIRE DE PYTHON



HISTOIRE DE PYTHON

En 2008, apparition de Python 3.0 (a.k.a. “Py3k”)

- **Incompatible** avec les versions 2.x (**un premier problème**)

Aujourd’hui (**Avril 2025**) : 1 branche active incompatibles : Python **3.13.2**

Dernière version Python **2.7.18** (Avril 2020)

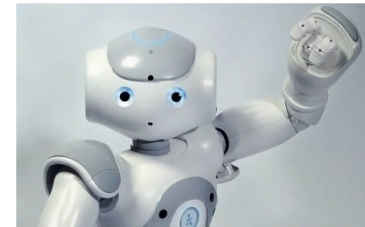
Les différentes améliorations et conventions du langage sont accessibles via les PEPs, **Python Enhancement Proposals** (<http://legacy.python.org/dev/peps>)

- PEP 8 : **Style Guide for Python Code** (<https://www.python.org/dev/peps/pep-0008>)
- PEP 20 : **The Zen of Python** (<https://www.python.org/dev/peps/pep-0020>)
- ...

PYTHON EST UTILISÉ ... (PARTOUT)

Dans des projets divers comme :

- blender
- SoftBank Robotics (Nao, Pepper, ...)
- micro Python pour micro-contrôleurs
- SciPy.org, (mathématiques et ingénierie)
- **Cybersécurité** et réseaux (<http://www.secdev.org/projects/scapy> ..., ...)



DES SHELLS INTERACTIFS EN LIGNE

<https://www.python.org>

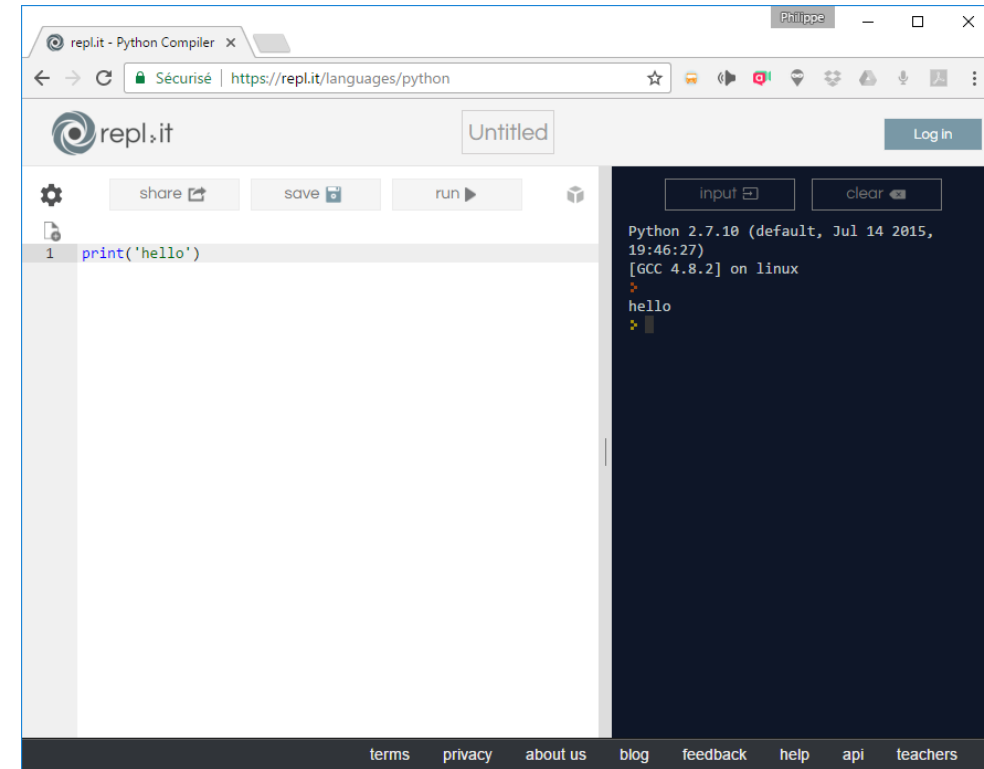
<https://repl.it/languages/python>

<https://repl.it/languages/python3>

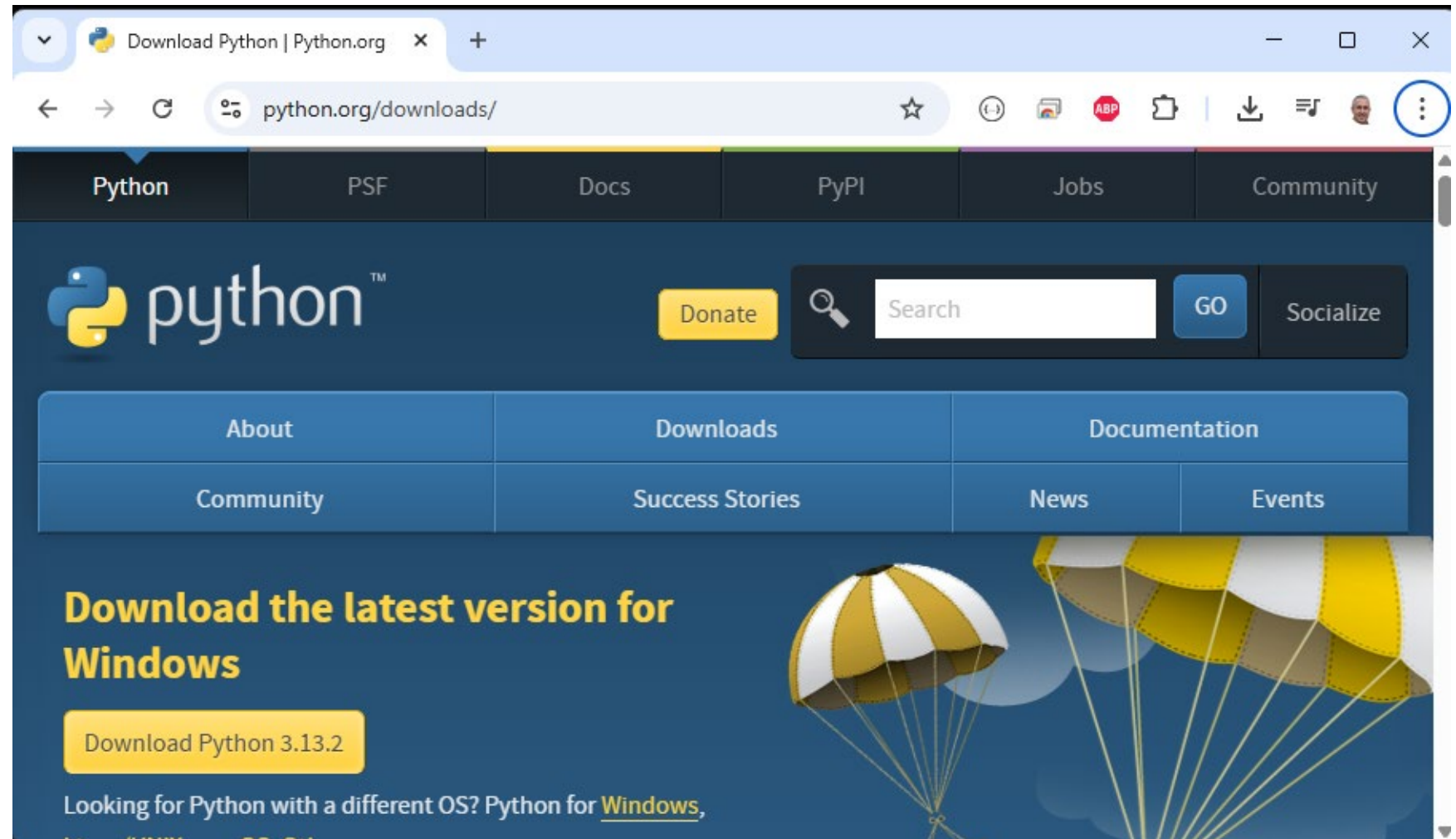
<http://www.pythontutor.com/visualize.html#mode=edit>

<https://www.pythonanywhere.com>

...

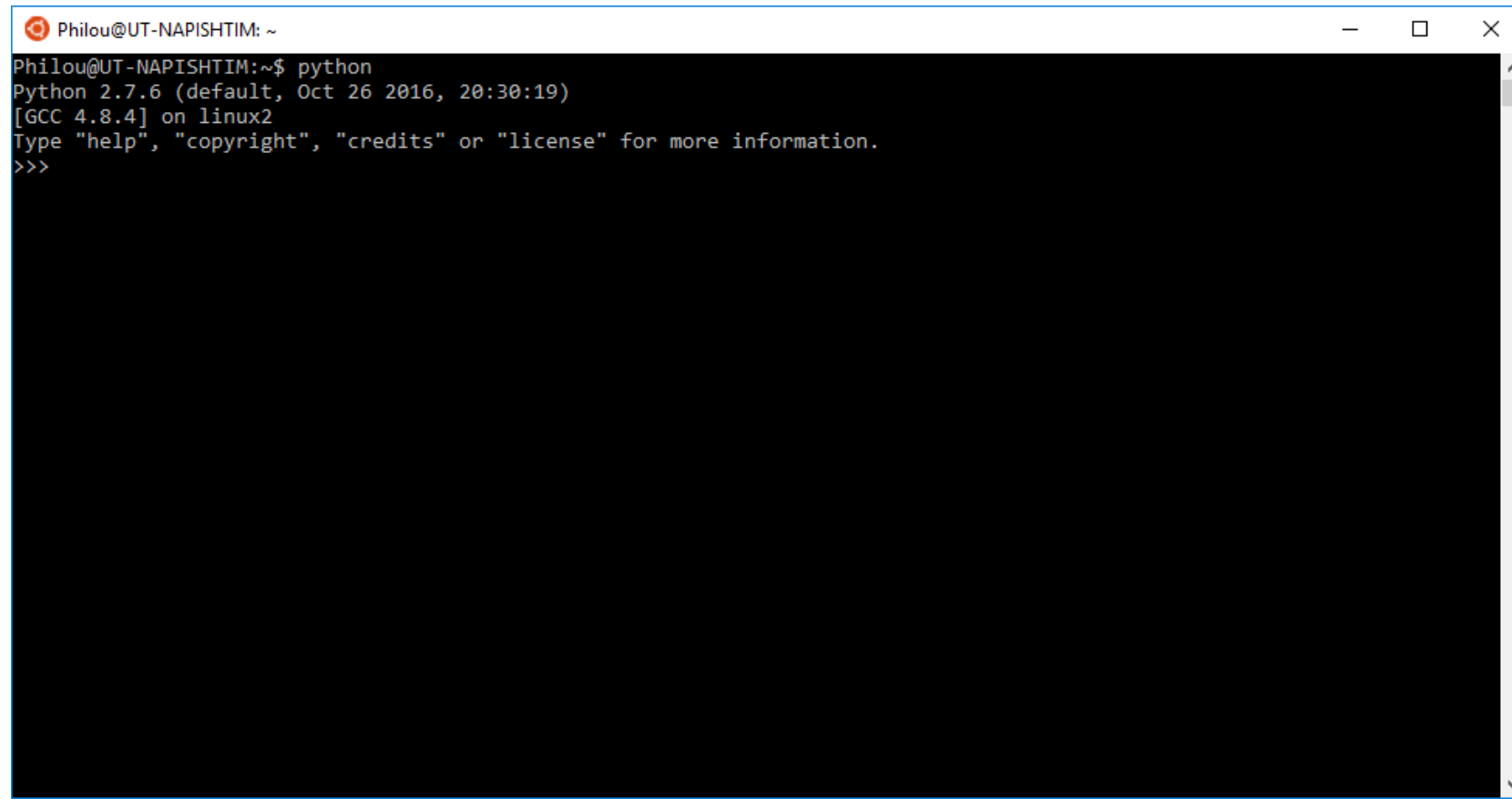


INSTALLER PYTHON



LANCER PYTHON

Il faut juste taper « **python** » ou « **python3** » (suivant les OS) dans un terminal

A terminal window with a blue title bar. The title bar contains a red icon, the text 'Philou@UT-NAPISHTIM: ~', and standard window controls (minimize, maximize, close). The terminal has a black background with white text. The text shows the command 'python' being executed, followed by the Python 2.7.6 startup banner: 'Python 2.7.6 (default, Oct 26 2016, 20:30:19)', '[GCC 4.8.4] on linux2', and 'Type "help", "copyright", "credits" or "license" for more information.' The prompt '>>>' is visible at the bottom of the first line of output.

```
Philou@UT-NAPISHTIM: ~  
Philou@UT-NAPISHTIM:~$ python  
Python 2.7.6 (default, Oct 26 2016, 20:30:19)  
[GCC 4.8.4] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>>
```

EXERCICE 0

Ouvrir un terminal, taper « **python** »

Taper « **import this** » puis « *touche entrée* »

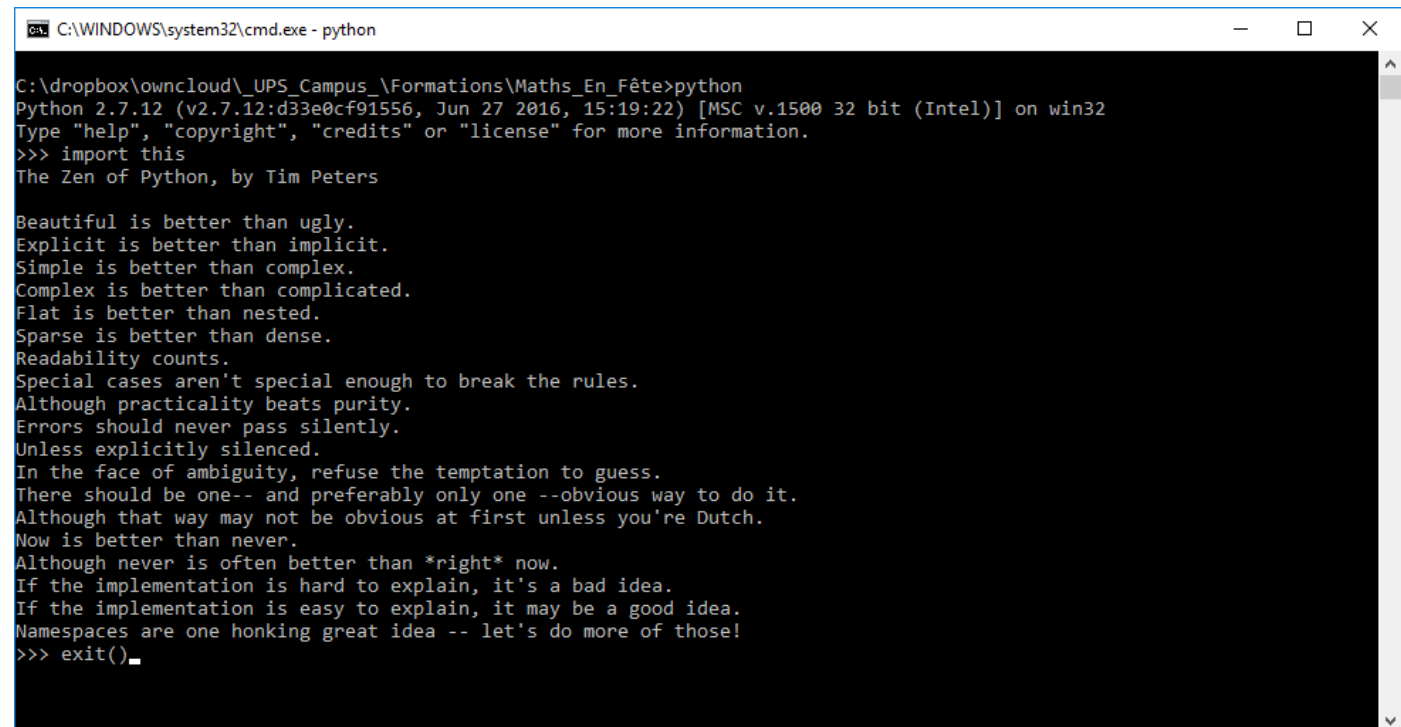
EXERCICE 0

Ouvrir un terminal, taper « **python** »

Taper « **import this** » puis « *touche entrée* »

Observer « l'easter egg PEP 20 »)

Taper « **exit()** » pour sortir



```
C:\WINDOWS\system32\cmd.exe - python
C:\dropbox\owncloud\UPS_Campus_Formations\Maths_En_Fête>python
Python 2.7.12 (v2.7.12:d33e0cf91556, Jun 27 2016, 15:19:22) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
>>> exit()_
```

EN RÉALITÉ ...

On utilise un **éditeur de texte** (intégré comme spyder, IDLE, processing.py, ...) ou un éditeur « *basique* » (nano, notepad++, ...)

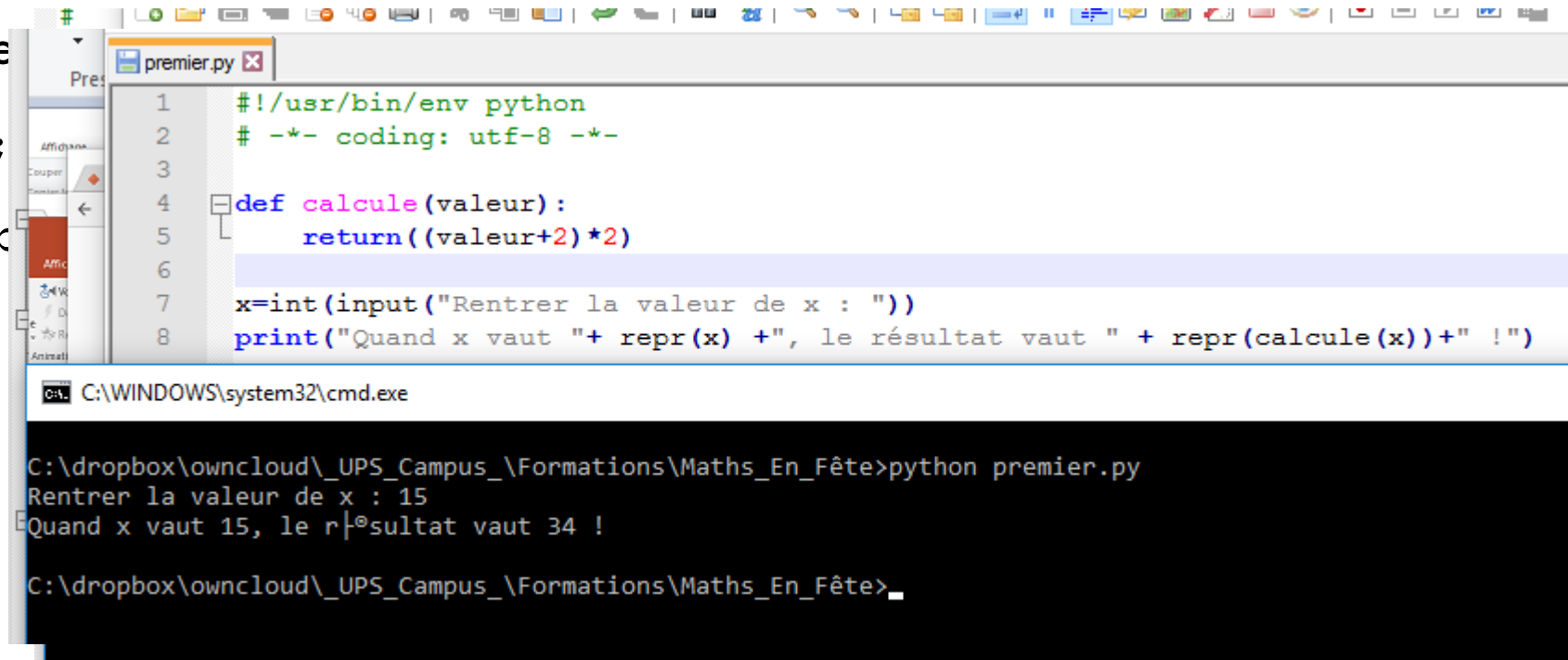
L'extension est **.py** (logique, non ?!)

Et on lance la commande : `python monfichier.py`

UN PREMIER EXERCICE

On veut simuler un programme comme ceux que l'on rencontre au collège :

- on choisit une
- on ajoute 2 ;
- on multiplie p
- on obtient le



```
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  def calcule(valeur):
5      return((valeur+2)*2)
6
7  x=int(input("Rentrer la valeur de x : "))
8  print("Quand x vaut "+ repr(x) +", le résultat vaut " + repr(calcule(x))+" !")
```

```
C:\WINDOWS\system32\cmd.exe

C:\dropbox\owncloud\_UPS_Campus_\Formations\Maths_En_Fête>python premier.py
Rentrer la valeur de x : 15
Quand x vaut 15, le résultat vaut 34 !

C:\dropbox\owncloud\_UPS_Campus_\Formations\Maths_En_Fête>
```

STRUCTURE D'UN PROGRAMME PYTHON

Des commentaires

Des structures ...

Des fonctions

Des répétitions

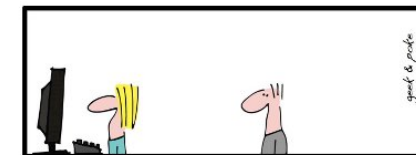
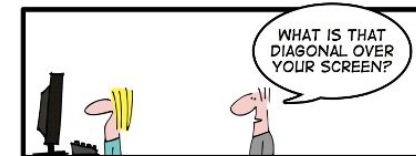
Des indentations

Structure de sélection

```
C:\dropbox\owncloud\_UPS_Campus_\Formations\Maths_En_Fête\ex1.py - Notepad++
Fichier Édition Recherche Affichage Encodage Langage Paramétrage Outils Macro Exécution Compléments Documents ?

ex1.py
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3  #
4  # Exercice 1 - encodage / décodage
5  #
6
7  # ddéfinir le dictinnaire
8  key = {'a':'n', 'b':'o', 'c':'p', 'd':'q', 'e':'r', 'f':'s', 'g':'t', 'h':'u', 'i':'v', 'j':'w', 'k':'x', 'l':'y', 'm':
9      'z', 'n':'a', 'o':'b', 'p':'c', 'q':'d', 'r':'e', 's':'f', 't':'g', 'u':'h', 'v':'i', 'w':'j', 'x':'k', 'y':'l', 'z':'m', ' ':' ' }
10
11 def encode(txt):
12     txt_code = ''
13     # pour chaque lettre de l'alphabet, trouver son correspondant
14     for i in range(len(txt)):
15         txt_code = txt_code + key[txt[i]]
16     # renvoyer le résultat
17     return(txt_code)
18
19 # fonction principale
20 def main():
21     txt = raw_input('texte à encoder : ')
22     txt_coded = encode(txt)
23     print "résultat: " + txt_coded
24
25 # utile pour le futur ;)
26 if __name__ == '__main__':
27     main()
```

Des particularités

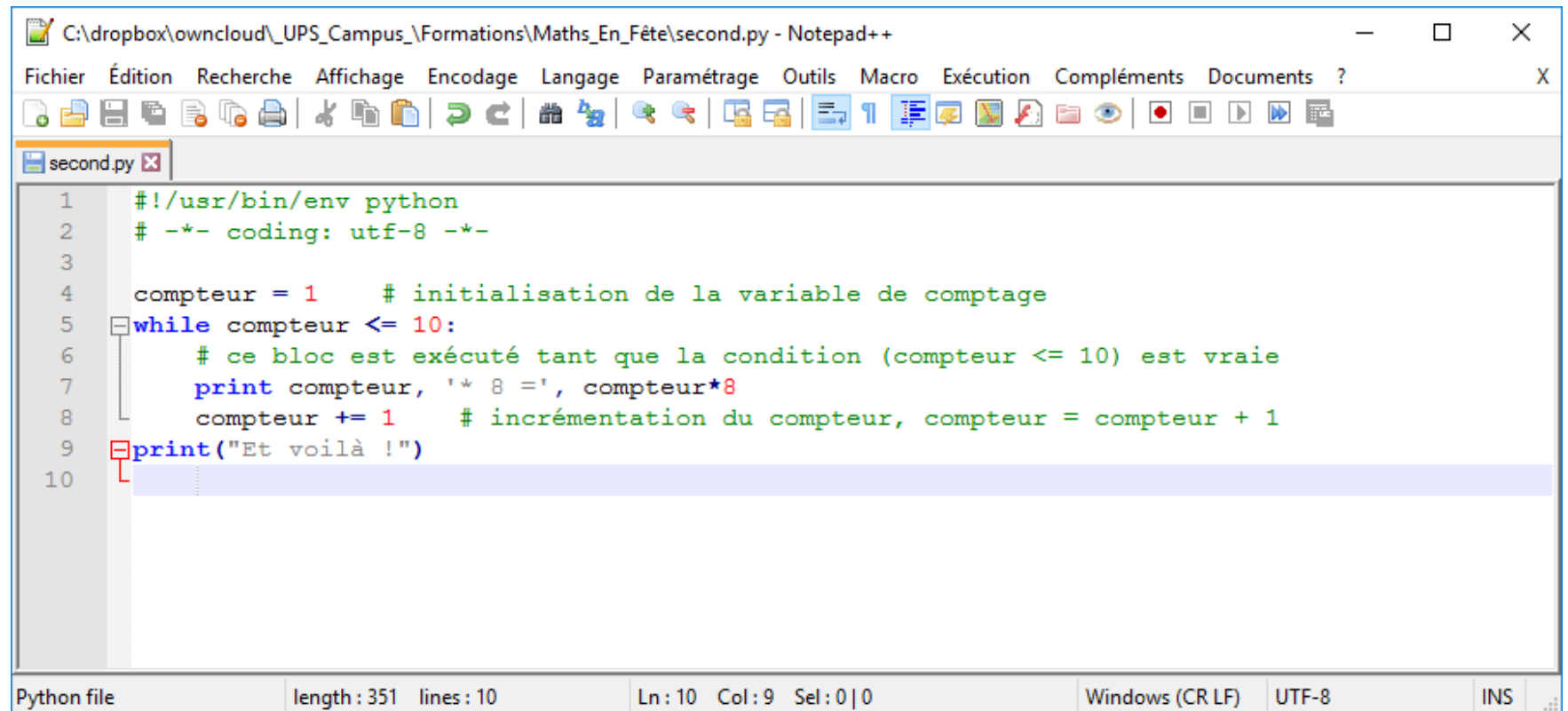


CHAPTER 2: PROGRAMMING

RULE 1: INDENT YOUR CODE
RULE 2: MAKE SURE IT HAS THE RIGHT COMPLEXITY

UN DEUXIÈME EXERCICE

Écrire un programme qui permette d'afficher la table de multiplication de 8



```
C:\dropbox\owncloud\_UPS_Campus_\Formations\Maths_En_Fête\second.py - Notepad++
Fichier  Édition  Recherche  Affichage  Encodage  Langage  Paramétrage  Outils  Macro  Exécution  Compléments  Documents  ?
second.py x
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  compteur = 1    # initialisation de la variable de comptage
5  while compteur <= 10:
6      # ce bloc est exécuté tant que la condition (compteur <= 10) est vraie
7      print compteur, '* 8 =', compteur*8
8      compteur += 1    # incrémentation du compteur, compteur = compteur + 1
9  print("Et voilà !")
10
```

Python file length : 351 lines : 10 Ln : 10 Col : 9 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

ALLER PLUS LOIN ...

Beaucoup de ressources sur le web

De nombreux aspects du langage sont intéressants à étudier ... et les résultats sont spectaculaires 😊