Introduction à l'algorithmique et à la programmation avec Python

Laurent Signac
https://deptinfo-ensip.univ-poitiers.fr

27 septembre 2017

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 !

```
>>> import this
```

Table des matières

Notations

Dans ce support de cours, les programmes sont généralement typographiés en police machine à écrire et encadrés, alors que les algorithmes n'ont pas de cadre et utilisent une police sans empattements.
Sauf si une indication contraire est donnée, le langage utilisé est Python. On différencie les «sessions shell» par la présence du prompt `>>>`. En l'absence de prompt, il s'agit généralement d'un programme à entrer dans un éditeur de texte et à exécuter ensuite (l'interface IDLE convient très bien pour faire tout ceci).

Introduction

L'informatique, telle que nous la connaissons aujourd'hui résulte à la fois des progrès théoriques, amorcés dans les années 30 par les travaux de Turing, et des progrès technologiques, liés à l'électronique, avec en particulier l'invention du transistor dans les années 40.
Depuis, ces deux facettes, science théorique et défis technologiques progressent de conserve.
L'objet de ce cours est de présenter l'ère numérique sous l'angle du codage des informations, puis celle des ordinateurs sous celui des algorithmes et de la programmation.

Ressources

La dernière version de ce fichier est disponible ici : https://deptinfo-ensip.univ-poitiers.fr/FILES/PDF/python1a.pdf
Les ressources Web associées à ce cours sont :

- Sur Updago : Algorithmique et Programmation Python
- https://deptinfo-ensip.univ-poitiers.fr/ENS/doku

Les introduction à Python sont nombreuses sur le Web, et certaines sont de très bonne qualité :

- Le cours de Bob Cordeau :
  http://perso.limsi.fr/pointal/python:courspython3
- Le cours de Pierre Puiseux :
  http://web.univ-pau.fr/~puiseux/enseignement/python/python.pdf

Voici ceux qui ont été utilisés lors de l'écriture de ce cours [1] :

- LE GOFF, Vincent, *Apprenez à Programmer en Python*, Le livre du Zéro
- PUISEUX, Pierre, *Le Langage Python*, Ellipses TechnoSup
- CHAZALLET, Sébastien, *Python 3 – Les fondamentaux du langage*, Eni
- ZELLE, John, *Python Programming*, Franklin, Beedle, and Associates
- LEE, Kent D., *Python Programming Fundamentals*, Springer
- SUMMERFIELD, Mark, *Programming in Python 3*, Addison Wesley, 2e ed.

Les parties du cours qui ne concernent pas précisément Python sont issues de nombreux ouvrages impossibles à tous mentionner ici et de quelques années de pratique.

Licence

1. Depuis, l'offre de livres sur Python 3 s'est diversifiée.

Chapitre I

Ordinateur, codage numérique

Il y a 10 sortes de gens, ceux qui savent compter en binaire et les autres.

1 Ordinateur ?

Les machines électroniques nous entourent désormais : de la simple calculatrice aux super-ordinateurs, en passant par les *smartphones*, les tablettes et les micro-ordinateurs. À partir de quel degré de complexité avons-nous affaire à un *ordinateur* ?
Comme nous allons le voir, les ordinateurs actuels sont à peu près calqués sur un modèle datant des années 50, le modèle de Von Neumann. Si nous devions retenir deux caractéristiques des ordinateurs, ce serait que :

- un ordinateur doit être programmable
- son programme doit être enregistré dans sa mémoire.

Par voie de conséquence, une simple calculatrice de poche n'est pas vraiment un ordinateur, alors que les modèles programmables plus perfectionnés en sont. De même que les tablettes et les *smartphones*...

1.1 Ordinateurs domestiques

Dans nos micro-ordinateurs, les différents composants sont : la carte mère, le processeur, la mémoire, la carte graphique, les disques...
Ces composants électroniques communiquent entre eux par l'intermédiaire de *bus* et les données qui transitent sont représentées sous forme *binaire*.
Dans un ordinateur, absolument tout (données et programmes, en mémoire, sur les disques durs ou sur CD...) est stocké sous forme binaire.
Dans le suite, nous verrons comment utiliser le système binaire, et comment des objets complexes, comme des images ou de la musique, sont codées en binaire.

Obtenir plus d'informations

Le film suivant :
http://www-sop.inria.fr/science-participative/film/
a été produit par l'INRIA et retrace l'histoire de l'informatique du 9e siècle à aujourd'hui en 24 minutes...

2 Codage numérique de l'information

L'information est la matière première de l'informatique[1]. Les algorithmes, qui sont constitués d'informations, stockent, manipulent et transforment d'autres informations, à l'aide de machines.
Tout, en informatique, est représenté *comme une séquence de 0 et de 1* : les algorithmes, ou plutôt les programmes, le contenu d'un livre, une photo, une vidéo...

1. Le mot vient d'ailleurs de là : **informa**tion auto**matique**, et a été adopté en 1967.

Pourquoi le binaire ?

Il y a une raison théorique et une raison technique à l'utilisation du binaire :

- on ne peut pas faire plus simple que 2 symboles. Avec un seul symbole, plus rien ne fonctionne (on a un seul codage de longueur fixe, la taille de l'écriture d'un nombre, écrit en unaire est proportionnelle au nombre, et non pas à son logarithme).
- les deux symboles 0 et 1 sont transposables électroniquement par : le courant passe ou ne passe pas (système assez robuste)

On représente donc toutes les informations sous forme de *bits* (binary digit = chiffre binaire). La quantité d'information est justement le nombre de bits nécessaires pour représenter cette information.

Unités

- le *bit* (*binary digit*) est l'unité d'information : 0 ou 1
- l'*octet* est un groupe de 8 bits (attention, octet se dit *byte* en anglais)

Tout le reste.... dépend du contexte. En particulier, le préfixe «kilo» correspond dans le système international au multiplicateur 10^3, mais conventionnellement, il était utilisé en informatique pour le multiplicateur 2^{10}. La vérité [2] est que, maintenant, nous devrions utiliser les préfixes du SI pour le multiplicateur 10^3 (symboles k,M,G...) et d'autres préfixes (kibi, mébi, gibi,... symboles ki,Mi,Gi...) pour les multiplicateurs 2^{10}, 2^{20}, 2^{30}...Il en est de même pour les débits (bits/seconde, puis kbits par seconde etc...). Dans les symboles, le «o» de octet est parfois remplacé par le «B» de byte, à ne pas confondre avec le «b» de bit...

Enfin, l'explorateur Windows utilise le symbole Ko pour 2^{10} octets (alors que ce devrait être 10^3 octets) et l'utilitaire GNU `du -k` utilise aussi des kilos de 1024 octets (encore qu'il a une option supplémentaire permettant d'utiliser les kilos du SI).

Dans le doute, partez du principe que si on vous vend de la mémoire (disque dur, mémoire flash...) sa taille est indiqué en utilisant le SI...

2.1 Changements de base

L'écriture d'un nombre en base b est formée à partir de b symboles, appelés chiffres. Les 10 chiffres de la base 10 sont par exemple : $0, 1, 2, 3, 4, 5, 6, 7, 8$ et 9.
Si l'écriture d'un nombre en base b est : $c_n c_{n-1} ... c_1 c_0$ (les c_i sont les *chiffres*), sa valeur sera :

$$v(c_n) \times b^n + v(c_{n-1}) \times b^{n-1} + ... + v(c_1) \times b^1 + v(c_0) \times b^0$$

où $v(c_i)$ est la *valeur* associée au chiffre (symbole) c_i. La valeur du symbole 9 en base 10 est par exemple neuf. Généralement, pour une base b inférieure à 10, on reprend les même chiffres qu'en base 10, avec les mêmes valeurs (mais on n'utilise que les b premiers chiffres). Pour une base b supérieure à 10, on ajoute aux chiffres ordinaires d'autres symboles, comme des lettres. La base 16 utilise par exemple les 16 chiffres suivantes : $0, 1, 2, ..., 9, A, B, C, D, E, F$.
Voici quelques exemple, la base est indiquée en indice à la fin du nombre. En l'absence de cet indication, c'est la base 10 qui est utilisée.

$$1011010|_2 = 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 90$$

$$5A|_{16} = 5 \times 16^1 + 10 \times 16^0 = 90$$

Pour passer de la base 10 à la base 2, on peut procéder par divisions successives (par 2). Voici comment trouver l'écriture binaire de 90 :

$$\begin{array}{rll} 90 & = 2 \times 45 & +0 \\ 45 & = 2 \times 22 & +1 \\ 22 & = 2 \times 11 & +0 \\ 11 & = 2 \times 5 & +1 \\ 5 & = 2 \times 2 & +1 \\ 2 & = 2 \times 1 & +0 \\ 1 & = 2 \times 0 & +1 \end{array}$$

La séquence des restes successifs, lue du dernier au premier donne : 1011010 qui est l'écriture en base 2 de 90.
On peut procéder de même en base 16 (en faisant des divisions par 16). Les restes sont alors entre 0 et 15, ce qui correspond bien aux valeurs des chiffres de la base 16.
Les nombres non-entiers sont codés de la même manière. Chaque chiffre placé après la virgule est associé à une puissance négative de la base :

$$101,011|_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} = 5 + 0,25 + 0,125 = 5,375$$