

# Notions avancées

En construction. Les liens sont juste donnés. Une introduction et un exemple devrait être proposé pour chaque rubrique, et le nombre de ces rubriques augmenté.

## Itérateurs

### Générateurs et "yield"

- <http://fr.openclassrooms.com/informatique/cours/pratiques-avancees-et-meconnues-en-python/es-generateurs-2>
- <http://feldboris.alwaysdata.net/blog/python-les-iterateurs-et-les-generateurs-fr.html>
- <https://wiki.python.org/moin/Generators>
- <http://sahandsaba.com/combinatorial-generation-using-coroutines-in-python.html>
- <http://code.activestate.com/recipes/580628-pluggable-python-generators/>

### Liste en compréhension

- [http://fr.wikipedia.org/wiki/Liste\\_en\\_compr%C3%A9hension](http://fr.wikipedia.org/wiki/Liste_en_compr%C3%A9hension)
- <http://www.pythonforbeginners.com/basics/list-comprehensions-in-python>
- <http://fgallaire.flext.net/comprehension-de-liste-en-python-map-filter/>, remplacement de map() et filter()
- [http://www.python-course.eu/list\\_comprehension.php](http://www.python-course.eu/list_comprehension.php), yc suppression de lambda et reduce()

### Expressions rationnelles (régulières)

- `import re`
- <http://howchoo.com/g/zdvmogrIngz/python-regexes-findall-search-and-match>
- <https://docs.python.org/2/howto/regex.html>
- <http://linuxfr.org/news/travailler-avec-des-expressions-rationnelles>
- [https://fr.wikipedia.org/wiki/Expression\\_rationnelle](https://fr.wikipedia.org/wiki/Expression_rationnelle)
- ...

## Décorateurs

- [http://www.python-course.eu/python3\\_memoization.php](http://www.python-course.eu/python3_memoization.php)

## Context managers

## Programmation orienté objet

- Exemples simples :
  - <http://nbviewer.ipython.org/url/bender.astro.sunysb.edu/classes/python-science/lectures/python-classes.ipynb>
  - <http://jeffknupp.com/blog/2014/06/18/improve-your-python-python-classes-and-object-oriented-programming/>

## Closures

- <http://stackoverflow.com/questions/36636/what-is-a-closure>
- <http://programmers.stackexchange.com/questions/40454/what-is-a-closure>

## Programmation fonctionnelle

- [Functional Programming in Python](#) (  )

## Tests unitaires

From:  
<https://dvillers.umons.ac.be/wiki/> - Didier Villers, UMONS - wiki

Permanent link:  
[https://dvillers.umons.ac.be/wiki/teaching:progappchim:notions\\_avancees?rev=1458898933](https://dvillers.umons.ac.be/wiki/teaching:progappchim:notions_avancees?rev=1458898933)

Last update: **2016/03/25 10:42**

