

Introduction à Python

1.1. Buts et moyens

Comme son nom l'indique, ce livre a pour but d'utiliser le langage Python en maths appliquées. Le but n'est pas d'apprendre le langage Python de manière exhaustive : on verra juste les notions fondamentales dont on aura besoin. Les élèves qui voudront aller plus loin pourront suivre le cours d'informatique pour étudier plus à fond ce langage.

Ici, l'idée est de faire des maths autrement qu'avec un papier et un crayon, mais l'accent sera mis sur les maths. Certaines notions ne seront peut-être pas encore connues (les dérivées par exemple). Ce n'est pas grave. Le minimum vital suffira.

1.2. Programmer ?

La **programmation** consiste à « expliquer » en détails à un ordinateur ce qu'il doit faire, sachant qu'il ne comprend évidemment pas une langue humaine, mais seulement ce qu'on appelle un **langage de programmation** (par exemple C, Python, Java, etc.). En d'autres termes, il faudra traduire une idée simple (par exemple « trier des nombres dans l'ordre croissant »), en un vrai raisonnement parfaitement structuré et détaillé, que l'on appelle un **algorithme**.

Un langage de programmation a de nombreux points communs avec un langage humain : il a une syntaxe (l'orthographe des mots) et une grammaire (la façon d'agencer les mots). La différence la plus importante est qu'un langage informatique ne tolère **aucune** erreur, alors que l'on peut mal parler une langue tout en se faisant comprendre quand même.

1.2.1. Types d'erreurs

La programmation est une tâche complexe, et on y commet de nombreuses erreurs.

Erreurs de syntaxe

Un programme ne peut être exécuté que si sa syntaxe est parfaitement correcte. Dans le cas contraire, le processus s'arrête (on parle communément de « plantage ») et vous obtenez un message d'erreur. Le terme syntaxe se réfère aux règles que les auteurs du langage ont établies pour la structure du programme. **Tous** les détails ont de l'importance : le respect des majuscules et des minuscules, l'orthographe, la ponctuation, ...

Erreurs sémantiques

Le second type d'erreur est l'erreur sémantique ou erreur de logique. S'il existe une erreur de ce type dans un de vos programmes, il n'y aura aucun message d'erreur, mais le résultat ne sera pas celui que vous attendiez.

Erreurs à l'exécution

Le troisième type d'erreur est l'erreur en cours d'exécution (*Run-time error*), qui apparaît seulement lorsque votre programme fonctionne déjà, mais que des circonstances particulières se présentent (par exemple, votre programme essaie de lire un fichier qui n'existe plus).

Pour des raisons anecdotiques, les erreurs de programmation s'appellent des « bugs ». *bug* est à l'origine un terme anglais servant à désigner de petits insectes gênants, tels les punaises. Il est arrivé à plusieurs reprises que des cadavres de ces insectes provoquent des court-circuits et donc des pannes incompréhensibles.



Recherche des erreurs et expérimentation

Débuguer efficacement un programme demande beaucoup de perspicacité et ce travail ressemble à une enquête policière. Vous examinez les résultats, et vous devez émettre des hypothèses pour reconstituer les processus et les événements qui ont logiquement entraîné ces résultats.

1.2.2. Les 5 règles d'or pour bien programmer

Programmer ne suffit pas. Il faut **bien** programmer, de sorte que quelqu'un qui voudra réutiliser votre programme puisse le faire facilement. En un mot, le programme doit être **compréhensible**. La première chose consistera à **décomposer un problème compliqué en plusieurs sous-problèmes simples**, donc à découper le programme en plusieurs sous-programmes. Voici les règles d'or à respecter :



Règle numéro 1.

Ne pas écrire de longs sous-programmes (pas plus de 10-15 lignes de code).



Règle numéro 2.

Chaque sous-programme doit avoir un objectif clair.



Règle numéro 3.

Ne jamais utiliser les fonctionnalités du langage dont on n'est pas sûr du résultat ou du rôle.



Règle numéro 4.

Éviter à tout prix le copier/coller, source d'innombrables erreurs.



Règle numéro 5.

Écrire des commentaires pour expliquer les parties les plus subtiles du programme.



Guido von Rossum

1.3. Python

Python est un langage portable, extensible, gratuit, qui permet (sans l'imposer) une approche modulaire et orientée objet de la programmation. Python est développé depuis 1989 par Guido **van Rossum** et de nombreux contributeurs bénévoles.

Il est apprécié par les pédagogues qui y trouvent un langage où la syntaxe permet une initiation aisée aux concepts de base de la programmation.

Extrait de
<http://www.linux-center.org/articles/9812/python.html>

1.3.1. Principales caractéristiques du langage

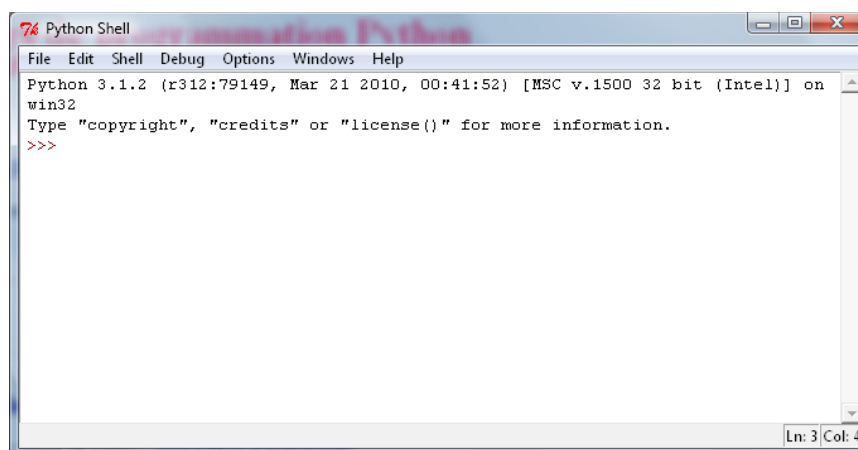
Détaillons quelques caractéristiques de Python :

- Python est **portable** : il fonctionne non seulement sur Linux, mais aussi sur MacOS, et les différentes variantes de Windows.
- Python est **gratuit**, mais on peut l'utiliser sans restriction dans des projets commerciaux.
- Python convient aussi bien à des **scripts** d'une dizaine de lignes qu'à des **projets complexes** de plusieurs dizaines de milliers de lignes.
- La **syntaxe** de Python est très simple et, combinée à des **types de données évolués** (listes, dictionnaires,...), conduit à des programmes à la fois très compacts et très lisibles. À fonctionnalités égales, un programme Python (abondamment commenté et présenté selon les canons standards) est souvent de 3 à 5 fois plus court qu'un programme C ou C++ (ou même Java) équivalent, ce qui représente en général un temps de développement de 5 à 10 fois plus court et une facilité de maintenance largement accrue.
- Python est **orienté objet**. Il supporte l'**héritage multiple** et la **surcharge des opérateurs**.
- Python est un langage qui **continue à évoluer**, soutenu par une communauté d'utilisateurs enthousiastes et responsables, dont la plupart sont des supporters du logiciel libre. Nous utiliserons dans ce cours la **version 3**. Il est à noter qu'un programme écrit en Python 2 n'est pas compatible avec la version 3, et nécessitera quelques modifications.

1.3.2. Installation de Python 3

Allez sur le site officiel de Python : www.python.org/download/

1. Choisissez la dernière version (non bêta) adaptée à votre système d'exploitation. Pour Windows, choisissez la version pour processeur x86.
2. Dans le programme d'installation, le plus simple est de cliquer chaque fois sur *Next* jusqu'au bouton *Finish*.
3. Python est maintenant installé. Vous le trouverez dans votre liste de programmes. Pour le démarrer, choisissez dans le répertoire Python le programme **IDLE (Python GUI)**. La fenêtre suivante devrait alors apparaître :



```
Python Shell
File Edit Shell Debug Options Windows Help
Python 3.1.2 (r312:79149, Mar 21 2010, 00:41:52) [MSC v.1500 32 bit (Intel)] on
win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

Un guide d'installation est disponible sur le site compagnon (voir § 1.4).

La version (ici 3.1.2) peut varier.

1.4. Site web compagnon

Vous trouverez sur le site associé à ce cours les programmes des jeux, afin d'éviter de perdre du temps à les recopier et afin de pouvoir les tester facilement. Vous trouverez aussi les corrigés des exercices et les ressources à utiliser.

L'adresse est www.apprendre-en-ligne.net/pymath/

1.5. Contenu du cours

La première chose à faire sera évidemment de se familiariser avec Python. Nous verrons les notions fondamentales dans le chapitre 2. D'autres notions seront abordées dans les chapitres suivants si nécessaire.

Le chapitre 3 présentera quelques algorithmes simples que nous programmerons (algorithme d'Euclide, calcul de la date de Pâques, etc.)

Nous nous intéresserons au chapitre 4 à des méthodes numériques pour trouver les zéros d'une fonction. Nous programmerons aussi un traceur de courbes.

Nous aborderons au chapitre 5 les suites numériques, ce qui nous conduira à la théorie du chaos.

Enfin, le dernier chapitre traitera des nombres premiers et de la manière de les trouver.

1.6. Ce que vous avez appris dans l'introduction

- Ce qu'est la programmation et quels sont les types d'erreurs que vous rencontrerez.
- Les 5 règles d'or pour bien programmer.
- Les principales caractéristiques de Python 3 et comment l'installer.