Informatique 1 : Introduction au langage Python

I Architecture d'ensemble d'un ordinateur

Cette partie, assez brève, présente de façon culturelle le fonctionnement global d'un ordinateur.
Nous commençons par décrire le matériel informatique constituant un ordinateur et permettant son fonctionnement. Notre but est de donner une idée rapide, sans entrer dans le détail logique du fonctionnement du processeur et encore moins dans le détail électronique caché derrière ce fonctionnement logique (les différents composants).
Le terme **informatique** est une contraction des mots information et automatique.

Définition 1 (*Informatique*)

Il s'agit donc d'appliquer à un ensemble de données initiales des règles de transformation ou de calcul déterminées (c'est le caractère automatique), ne nécessitant donc pas de réflexion ni de prise d'initiative.
Un ordinateur est une concrétisation de cette notion.

Définition 2 (*Ordinateur*)

Il est donc nécessaire que l'ordinateur puisse communiquer avec l'utilisateur pour permettre l'entrée des données initiales, la sortie du résultat du traitement, et l'entrée des règles d'automatisation, sous la forme d'un programme.

Un ordinateur est un terme générique qui regroupe énormément d'appareils de la vie quotidienne : smartphones tablettes, la plupart des appareils électroménagers, la plupart des véhicules et les machines sur lesquelles nous travaillerons en TP de Python.

Le modèle le plus couramment adopté pour décrire de façon très schématique le fonctionnement d'un ordinateur est celui décrit dans la figure suivante :

Les schéma se compose de différents éléments :

1. **Entrées - sorties** :

2. **Mémoire** :

3. **Transfert de données :**

4. **Processeur :**

Remarque 3

II Le langage Python

1 Présentation

Définition 4 (*Langage de programmation*)

Python est un langage de programmation qui possède trois caractéristiques principales :

1. **C'est un langage de haut niveau** :

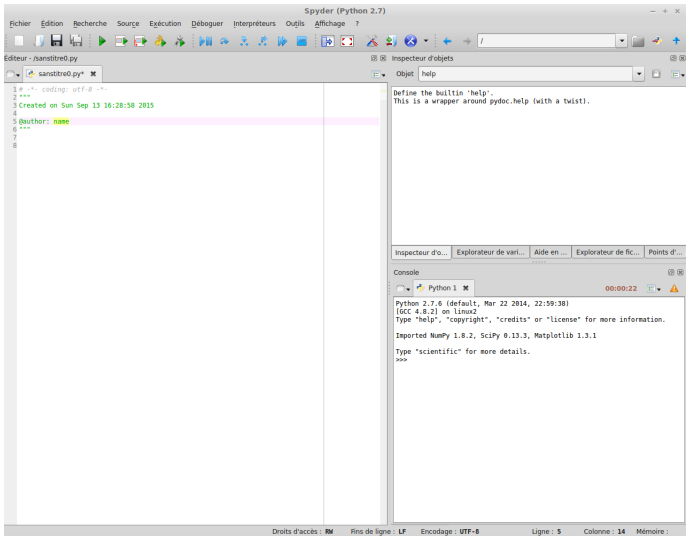
2. **C'est un langage semi-compilé** :

3. **C'est un langage où la présentation fait partie de la syntaxe** :

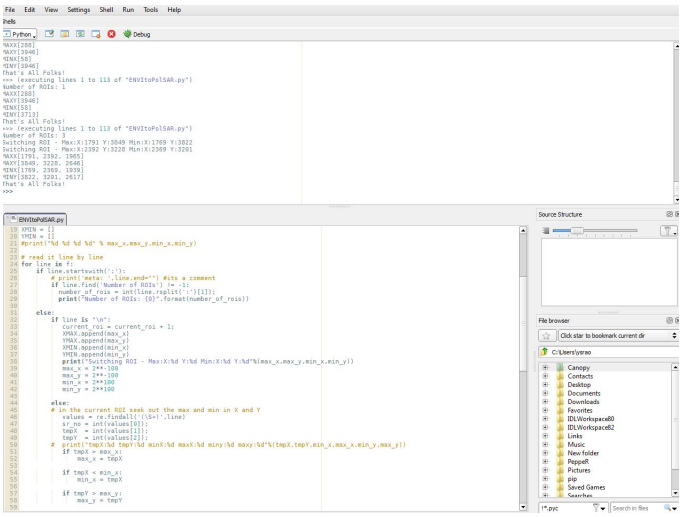
2 Environnement de travail avec Python

Pour pouvoir écrire en langage Python et exécuter les fonctions ou les programmes qu'on écrit, on utilise un logiciel spécifique qu'on appelle un environnement de travail. Un tel environnement permettra notamment de sauvegardera le travail fait en Python. On travaillera sous deux environnements différents :

1. Spyder :



2. Pyzo :



On explique l'intérêt de chaque fenêtre :

1. **La console** :

2. **L'éditeur de texte** :

3. **Informations :**

Remarque 5

III Premières utilisations de Python

1 Fonction d'aide

On commencera toujours par créer un nouveau fichier dans le shell et on sauvegardera régulièrement son travail. La fonction la plus importante est la fonction : `help()`.

Lorsqu'on tape ceci dans la console, on a l'apparition de `help>` et on peut taper une fonction pour savoir à quoi elle sert. Par exemple, si on tape dans la console

```
help()
```

puis, lorsque `help>` est affiché, on tape dans la console
`print`

on obtient toutes les explications sur l'utilisation de la fonction `print` de Python en anglais. Celle-ci est une fonction d'affichage de valeur dont nous verrons l'utilité en TP.
Si, lorsque `help>` est affiché, on tape sur entrée, on revient au mode de console initial.

Exemple 6

2 Utilisation de Python comme une calculatrice

On peut directement dans la console taper des instructions comme pour une calculatrice.

Exemple 7

3 Importation d'un module

Un certain nombre de fonctions sont chargés dans Python dès le démarrage de l'environnement de travail. Parfois, pour des fonctions plus spécifiques, il sera nécessaire de charger en plus un **module** complémentaire.
Pour ce faire, il y a plusieurs façons (on donne l'exemple avec le module `math` qui comporte des fonctions mathématiques) :

1. `import math`
2. `from math import *`
3. `import math as m`

Commentaire :

Exemple 8

Remarque 9

4 Notion de fonction

On donne ici une approche très brève de la notion de fonction en Python. Ceci sera vu en détail dans le prochain chapitre.
Une fonction en Python est un analogue d'une fonction mathématique : celle-ci prend en argument un certain nombre d'objets pour en retourner d'autres après un calcul. Par exemple, pour définir la fonction $g : x \mapsto x^2 + x + 1$, en Python on écrira :

```
def g(x):
        return x**2 + x + 1
```

dans le shell puis on compilera en mettant en surbrillance ce code tapé dans le shell et on sélectionnera « exécuter la sélection » ou en faisant F5. On pourra ensuite utiliser cette fonction **dans la console** en tapant

```
>>> g(3)
```

pour obtenir la valeur de g en 3.
Commentaire :

Exemple 10

Remarque 11

IV Codage

1 Mémoires, notion de codage

La mémoire d'un ordinateur est caractérisée par :

1.

2.

3.

La mémoire est un dispositif électronique qui sert à stocker des informations, qui, physiquement sont des impulsions électriques : un système binaire à deux états. On peut donc voir la mémoire comme un dispositif qui stocke un ensemble de 0 et de 1.

Les informations sont donc **codées** dans la mémoire sous forme binaire. Nous allons voir dans la suite comment on peut stocker des entiers puis des réels dans la mémoire de l'ordinateur.

2 Changement de base de numération

Dans cette partie b est un entier supérieur ou égal à 2.

Théorème 12

Pour tout $N \in \mathbb{N}$, il existe une suite d'entiers naturels $x_0, x_1, x_2, \ldots$ inférieur ou égaux à $b-1$ tel que

$$N = x_0 b^0 + x_1 b^1 + x_2 b^2 + \cdots .$$

Pour préciser dans quelle base on écrit un entier, on écrira :

$$N = \overline{x_0 x_1 x_2 \cdots}^b$$

Exemple 13

Remarque 14

Ainsi, lorsqu'on veut coder un entier écrit dans la base $b = 10$ en binaire (base $b = 2$), il faut changer de base de numération. On ne donnera pas de théorème précis mais seulement trois exemples qui donneront une méthode générale.

Exemple 15 (*Passage de la base 2 à la base 10*)

Écrivons le nombre $\overline{1101}^2$ en base 10.

Exemple 16 (*Passage de la base 10 à la base 2*)

Écrivons le nombre $\overline{242}^{10}$ en base 2.

Exemple 17 (*Passage de la base 2 à la base 6*)

Écrivons le nombre $\overline{11011}^2$ en base 6.

Remarque 18

3 Codage en mémoire : limitations

La mémoire est découpée en différentes parties pour stocker les entiers. Chaque partie a une taille limitée qu'on nomme **nombre de bits** où chaque bit peut prendre la valeur 0 ou 1.

Exemple 19

Néanmoins, ce type de situation ne permet a priori pas de coder d'autre nombres que des entiers alors qu'un ordinateur doit être capable de manipuler des entiers négatifs et des nombres décimaux ou plus précisément des nombres en **virgule flottante**. C'est la façon de mettre en mémoire ce type de nombre qu'on va voir dans la suite.

Théorème 20 (*Notation scientifique en base b*)

Soit $b \in \mathbb{N}$, $b \geq 2$. Soit $x \in \mathbb{R}^+$. Il existe un unique entier $z \in \mathbb{Z}$ et un unique nombre réel $y \in [1, b[$ tel que

$$x = y \times b^k.$$

Exemple 21

Remarque 22

Corollaire 23

On a désormais un moyen de coder en mémoire des nombres décimaux. On a le schéma suivant :

1. **Signe** :

2. **Exposant :**

3. **Mantisse :**

Remarque 24 (*Limitations du stockage en mémoire des nombres décimaux*)