# Notes de cours Python

Thibaut Marmey

September 18, 2018

## Contents

**1 Commandes générales** **1**
1.1 Environnement . . . . . 1
1.2 Le langage . . . . . 1
1.2.1 Infos . . . . . 1
1.2.2 Instructions utiles . . . . . 2
1.2.3 Les string . . . . . 2
1.2.4 Les listes . . . . . 2
1.2.5 Les boucles . . . . . 3
1.2.6 Boucle *for* . . . . . 3
1.2.7 Condition *if* . . . . . 3
1.2.8 Les fonctions . . . . . 3

# 1 Commandes générales

## 1.1 Environnement

- Tutorial 1 python
- Suite du tutorial : page où on s'est arrêté
- Programmes python
- Changer les permissions du programme .py : *chmod -x nom.py*
- Exécuter le programme : *python nom.py*

## 1.2 Le langage

### 1.2.1 Infos

- La division renvoie toujours un *double*
- Mettre un nombre au carré : utiliser ****
- Attendre une entrée clavier : *x = int(input("Please enter an integer: "))*
- Pour commenter en multi lignes, il faut utiliser trois guillements : *""" commentaires """*

### 1.2.2 Instructions utiles

- Ajouter élément à la fin de la liste : *liste.append[value]*
- Insérer élément dans une liste : *liste.insert(indice, value)*
- Itérer sur une suite de nombre : *range(nombre)* (utiliser dans boucle *for*)
  C'est un objet dit *itérable* c'est à dire qu'elle est utilisée par des *itérateurs* comme *for* ou *list()*. Pour obtenir une liste de nombre successif on peut ainsi combiner *range()* et *list()* : *list(rang(n))*
- L'instruction *pass* ne fait rien, elle permet de de fournir une syntaxe correcte comme : *while True: pass*

### 1.2.3 Les string

- Dans un *print()*, si l'on ne veut pas que les charactères qui précèdent le charactère \ on peut utiliser le paramètre *row : r*.
  *print(r"mot\nom")* : ainsi la combinaison $\backslash n$ n'est pas considérée comme un saut de ligne
- Pour modifier le comportement de saut de lige par défaut de *print*. Rajouter la valeur *end* : *print(a, end = ', ')*
- Opération sur les strings : possibilité des les additionner entre eux, de les multiplier par un entier : *3 * 'un' + 'deux'*
- Accéder aux élements d'un tableau ou string etc... : *tab[indice]*. L'indice commence à 0. On peut accéder au dernier élément par l'indice -1.
- Les strings sont immuables c'est à dire qu'on ne peut pas modifier d'élément. On peut seulement lire les éléments.
- Récupérer plusieurs éléments du tableau : *tableau[n:m]*. Permet de récupérer l'élément n juqu'a m-1.
- Récupérer taille d'un tableau : *len(tableau)*

### 1.2.4 Les listes

- Déclaration : *liste = [1, 2 ,3]*
- Concaténation : *liste1 + liste2*
- Les listes sont muables (contrairement aux strings) : *liste[1] = 20*
- Remplacer des valeurs : *liste[0:3]=[10,20,30]*
- Effacer la liste ou certains éléments : *liste[:] = []*
- Imbriquer des listres entre elles : *liste3 = [liste1, liste2]*. Les listes n'ont pas besoin de posséder le même type de valeur.
  Accéder au premier élément de liste2 :*liste3[1][0]*

- Affectation multiples de variables : *a, b = b, a+b*. Cepdant lors du calcul de *b*, *a* a toujours la même valeur et n'est pas encore égale à *b*.

### 1.2.5 Les boucles

- Les boucles *for, while* finissent par un *" : "* et ne nécessitent pas de parenthèse.
- A l'intérieur de ces fonctions, l'indentation est primordial puisque c'est la méthode utilisée par python pour regrouper les instructions.
- Pour signaler la fin d'une fonction un double saut de ligne est nécessaire.
- On peut combiner les boucles *while* et *for* avec des *break*, *continue* et *else*.
  En combinant un *for avec un break à la fin* suivi d'un *else*. Ainsi on va d'abord aller dans la boucle *for*, si une condition est bonne on break la boucle *for* et donc on ne va pas dans le *else*. Si au contraire à la fin de la boucle *for* on n'a pas eu de *break* on va alors passer dans la boucle *else*.
- *Continue* quant à lui fait passer directement la boucle à l'instruction suivante.

### 1.2.6 Boucle *for*

- L'instruction for itère sur les éléments d'une séquence : *for w in words:*
- Si l'on veut modifier la séquence sur laquelle on itère, il faut créer une copie de cette séquence dans l'instruction du *for*: *for w in ranges words**[:]***
- Itérer sur une suite de nombre : *for i in range(5)* : génère 5 valeurs de 0 à 5 exclu (le dernier élement n'est jamais pris en compte)

### 1.2.7 Condition *if*

- Finir l'instruction par un deux-points *:*
- Les autres instructions sont : *elseif cd1:* et *else:*
- Toujours sauter des lignes entre ces instructions et faire attention à l'indentation

### 1.2.8 Les fonctions

- Définir une fonction par *def*
- Faire attention à l'indentation
- Chaîne de documentation : explication succinte de la fonction dans les premières lignes de la fonction.