Mes débuts avec python

Votre premier programme en Python

A partir de l'invite de commande.

à taper si cela n'est pas déjà fait

```
>>> print("Bonjour le monde !")
Bonjour le monde
```

A partir d'un fichier (*appelle le prof*)

ne pas saisir

```
print("Bonjour le monde !")
```

1.Les calculs en python

à taper

```
>>> 8*3.57
28.559999999999999
```

Voici maintenant la table des opérations

Opérateur	Opération
+	Addition
-	Soustraction
*	Multiplication
/	Division

TABLE 2.1 – Opérateurs.

Maintenant fais des calculs avec priorités opératoires (tu te rappelles ?)
Voici un petit calcul sympa

à taper

```
>>> (5 + 30) * 52
1820
>>> 35 * 52
1820
```

voici un deuxième

à taper

```
>>> 5+30*20/10
65.0
```

Et hop un troisième. On a cette fois-ci rajouter **print** pour afficher le résultat . Mais comme le résultat sera de toute façon affiché , ce **print** ne sert à rien ici….Mais je vais vous expliquer pourquoi je l'ai introduit .

à taper

```
>>> print(((5+30)*20)/10)
70.0
```

Tu peux inventer et faire des calculs à souhait !

2.Les variables

On avait déjà abordé la notion de variable avec scratch, eh bien c'est la même chose avec python. *(Tu peux appeler le prof pour plus d'explication)*

Voici un exemple bien clair !

à taper

```
>>> fred = 100
>>> print(fred)
100
```

Essaye de deviner le résultat avant de l'afficher (avant de le « **print** »)

à taper

```
>>> fred = 200
>>> jean = fred
>>> print(jean)
```

L'orthographe de nos variables précédents n'est pas compliqué, mais imaginons que l'on veuille créer une variable qui s'appelle « Jean Claude » ou « Nombre d'eleves» … c'est simple : on écrit la même chose dans python, sauf que l'on doit mettre des « _ » à la place des espaces, donc nos variables deviennent « Jean_Claude » et « Nombre_d_eleves» .

Voici un exemple avec une variable que je veux appeler « Nombre d'étudiants »

à taper

```
>>> nombre_d_étudiants=200
```

à taper

```
>>> print(nombre_d_étudiants)
200
```

Vous voyez ça marche !

```
ceci_est_aussi_un_nom_de_variable_valide_mais_peut_être_pas_très_utile
```

Utilisation des variables

Maintenant nous savons comment créer une variable, mais comment l'utilisons-nous ?

Exercice

combien d'argent tus as à la fin de l'année, si tu gagne 5€ par semaine à faire le ménage, 30€ par semaine à livrer les journaux et dépense 10€ par semaine ? *(il y a 52 semaines dans l'année)*

Si je te demande de faire le calcul avec python, tu feras peut-être ça :

à taper

```
>>> print((5 + 30 - 10) * 52)
1300
```

Mais moi je veux que tu fasses le calcul avec des variables ! c'est-à-dire, tu dois donner un nom à la somme d'argent gagnée ,un nom à la somme d'argent perdue et un nom au nombre de semaine. Et puis faire le calcul avec ces noms. *(Appelle le prof si tu as besoin d'aide)*

Je vais un peu t'aider :

a taper

```
>>> ménage=5
>>> livraison_journal=30
>>> dépenses=10
```

A toi de faire le reste

3.Les chaînes de caractère

les variables peuvent être utilisées pour toutes sortes de choses et pas seulement des nombres. En programmation, la plupart du temps, nous appelons les textes des « chaînes ». Cela peut sembler un peu étrange mais vous pouvez imaginer que les textes sont des tas de lettres « enchaînées » (ou jointes). Chaque caractère est un peu un maillon de la chaîne. Peut-être cela fait-il plus sens ?

Mais peut-être que non, finalement.

Dans ce cas tout ce que vous avez besoin de savoir est que les chaînes sont juste un tas de lettres, de nombres et des symboles. Votre nom peut être une chaîne. Comme votre adresse. Le premier programme Python que nous avions créé à la première partie utilisait une chaîne : « Bonjour le monde ! ».

En python nous créons une chaîne en plaçant des guillemets « " » [8] autour du texte. Ainsi nous pouvons reprendre notre variable **fred** inutile et la faire pointer vers une chaîne comme cela :

à taper
```
>>> fred = "Ceci est une chaîne."
```

à taper
```
>>> print(fred)
Ceci est une chaîne.
```

à taper
```
>>> fred = 'Cela est une autre chaîne.'
>>> print(fred)
Cela est une autre chaîne.
```

Tours de chaînes

Il y a une question intéressante : que vaut dix fois cinq, « 10*5 » ? La réponse est, bien sûr, cinquante.

Bon d'accord, ce n'est pas intéressant du tout.

Mais que vaut dix fois « a » (10*'a') ? Cela peut paraître une question sans queue ni tête, mais il y a une réponse dans le monde de Python :

à taper
```
>>> print(10 * "a")
aaaaaaaaaa
```

Amusez-vous en essayant le même chose avec des chaînes de plusieurs caractères.

Une autre astuce avec une chaîne consiste à utiliser des *valeurs embarquées*. Vous pouvez faire cela avec « **%s** » qui est comme une marque (ou un paramètre substituable) pour une valeur que vous voulez inclure dans une chaîne. C'est plus simple à expliquer avec un exemple :

à taper
```
>>> montexte = "J'ai %s ans."
>>> print(montexte % 12)
J'ai 12 ans.
```

Dans la première ligne, la variable **« montexte »** est créée pointant sur une chaîne qui contient des mots et un caractère assez bizarre qui est : « %s » . Ce caractère est en fait une **balise** , son rôle est de dire à la console Python : « remplace moi avec quelque chose ».

Ainsi au niveau de la ligne suivant, quand nous appelons **« print(montexte % 12) »** nous utilisons le symbole « % » pour dire à Python de remplacer la **balise** « %s » par le nombre 12. Nous pouvons recycler cette chaîne et lui passer différentes valeurs :

à taper

```
>>> montexte = "Bonjour %s, comment ça va aujourd'hui ?"
>>> nom1 = "Guillaume"
>>> nom2 = "Johan"
>>> print(montexte % nom1)
Bonjour Guillaume, comment ça va aujourd'hui ?
>>> print(montexte % nom2)
Bonjour Johan, comment ça va aujourd'hui ?
```

Un autre exemple pour y voir plus clair

à taper

```
>>> montexte = "Bonjour %s et %s, comment ça va ?"
>>> nom1 = "Guillaume"
>>> nom2 = "Johan"
>>> print(montexte % (nom2,nom1))
Bonjour Johan et Guillaume, comment ça va aujourd'hui ?
```

4. Les listes

Du lait, du fromage, du céleri, de la confiture et du sirop : ce n'est pas vraiment une liste de courses, mais c'est suffisant pour notre propos. Si vous conservez ces informations dans une variable vous pouvez créer une chaîne :

à taper

```
>>> courses = "lait, fromage, céleri, confiture, sirop"
>>> print(courses)
lait, fromage, céleri, confiture, sirop
```

Une autre manière de faire est de créer une « liste » qui est un type particulier d'objet en Python. Notre liste sera crée en utilisant des crochets [] .

à taper

```
>>> liste_de_courses = ["lait", "fromage", "céleri", "confiture", "sirop"]
>>> print(liste_de_courses)
['lait', 'fromage', 'céleri', 'confiture', 'sirop']
```

Pourquoi des crochets ? eh bien, tu vas voir que cela va nous permettre de pouvoir afficher (ou accéder) chaque élément de la liste en utilisant sa position (qu'on appelle index).
Tu y verras plus clair avec cet exemple :

à taper

```
>>> print(liste_de_courses[2])
céleri
```

Tu comprends maintenant ? ….. Maintenant affiche le sirop en utilisant les crochets.

> Je ne te l'ai pas dit mais tu as vu que les listes commencent à la position 0 ; ainsi le premier élément est le numéro 0, le deuxième est 1, le troisième est 2 etc..

Si je suis seulement intéressé par les éléments numéros 2,3,4 et 5 de ma liste . Je vais juste faire :

à taper

```
>>> print(courses[2:5])
['céleri', 'confiture', 'sirop']
```

Il affiche les éléments depuis l'index 2 jusqu'à l'index 5.

Remplacer des éléments de ma liste

Je voudrais remplacer dans ma liste de tout à l'heure le « fromage » par de du « laitue », eh bien je vais juste faire ça :

à taper

```
>>> liste_de_courses[2] = "laitue"
>>> print(liste_de_courses)
['lait', 'fromage', 'laitue', 'confiture', 'sirop']
```

Tu as compris ? Tu peux maintenant faire les remplacements que tu veux.

Ajouter plus d'objets…

Maintenant je vais ajouter du chocolat sur ma liste :

à taper

```
>>> liste_de_courses.append('chocolat')
>>> print(liste_de_courses)
['lait', 'fromage', 'laitue', 'confiture', 'sirop', 'chocolat']
```