Premiers programmes avec Python

Après avoir fait vos premiers pas sur Python en pratiquant directement dans le shell, vous allez construire vos premiers programmes qui pourront être sauvegardés si nécessaires.

Consignes :
→ Ouvrez votre session sur le réseau du lycée. Dans votre dossier personnel, créez un dossier (par exemple "Informatique").
→ Ouvrez ensuite l'environnement PYZO. La partie gauche ou inférieure de la fenêtre contient un éditeur de texte. Vous y écrirez vos programmes. Si nécessaire, créer un nouveau fichier.
Commencez par sauvegarder ce fichier dans le dossier qui vient d'être créé (par exemple sous le nom "TP01_ex01.py"). À chaque exercice, vous enregistrerez un nouveau fichier.
→ Les programmes seront écrits en python 3.
→ Enfin, vous devrez sélectionner les instructions utiles que vous exécuterez (sélectionner les lignes puis bouton de droite de la souris - menu contextuel – puis exécuter la sélection ou bien aller dans le menu Exécuter et choisissez une des commandes).

1) Instructions d'entrée et de sortie avec python
L'instruction a=input('Entrer une donnée : ') permet d'affecter dans la variable *a* la réponse de l'utilisateur à la phrase « Entrer une donnée ». Par défaut, python considèrera que la variable *a* entrée est une chaîne de caractères. Si vous souhaitez que l'utilisateur entre une valeur numérique, vous devrez donc la convertir.

Exemples :
→ a=int(input('Entrer un nombre : ')), si a est un nombre entier,
→ a=float(input('Entrer un nombre : ')), si a est un flottant.
L'instruction print('Du texte ou ',autre) permet d'afficher, lors de l'exécution d'un programme, des « blocs » de données de tout type : ici, la chaîne de caractère « Du texte ou » sera écrite, et suivie de la valeur de la variable autre.

2) Instructions conditionnelles

Si . . . alors . . .

```
if condition :
    instruction(s)
```

Si . . . alors. . ., sinon. . .

```
if condition :
    instruction(s)
else :
    autre(s) instruction(s)
```

Si . . . alors. . ., sinon, si . . .

```
if condition :
    instruction(s)
elif autre condition :
    autre(s) instruction(s)
else :
    encore d'autres instructions
```

Attention : L'*indentation* est *signifiante* et remarquez les :

Exercice 1 Valeur absolue
Écrire un programme qui, après avoir demandé à l'utilisateur un nombre *x*, calcule et affiche sa valeur absolue.

3) Boucles du type for

Si on a l'intention de répéter *n* fois un certain bloc d'instructions, la syntaxe dans Python est :

```
for compteur in range(n,m,pas) :
    instruction(s)
```

La variable compteur parcourt l'ensemble des entiers compris entre *n* et *m* – 1 suivant un pas donné.

→ Si le *pas* est omis, alors *pas* = 1.

→ Si *n* est omis, alors *n* = 0. En particulier, range(m) donne la séquence 0, 1, 2, . . . , *m* – 1.

Exercice 2 L'instruction *range*

Tapez les commandes suivantes dans le shell pour en comprendre le fonctionnement :

list(range(8)) puis list(range(4,9)) puis list(range(4,9,3)) puis list(range(5,0,-1))

Exercice 3 Liste de carrés

Faire afficher la liste de tous les carrés des entiers compris entre 5 et 15.

4) Boucles du type while

Lorsqu'on veut qu'une (ou plusieurs) instruction(s) se répète(nt) tant qu'une condition est vérifiée, on écrit :

```
while condition :
    instruction(s)
```

Attention : La condition doit évoluer à chaque itération pour éviter que la boucle se répète indéfiniment.

Exercice 4 Algorithme de seuil

On considère la suite (u_n) définie, pour tout entier naturel n, par $u_n = \dfrac{1}{n^3 + n + 3}$.

Celle-ci est décroissante et converge vers 0.

Écrire un programme qui permette d'obtenir le premier rang n à partir duquel $u_n \leqslant 0{,}001$.