Unité 1 : Débuter la programmation en Python	Compétence 2 : Les types de données en Python
Dans cette deuxième leçon de l'unité 1, vous allez découvrir comment utiliser le type des données en Python.	**Objectifs :** • Connaitre les différents types de données en langage Python. • Mettre en forme le format d'une donnée numérique.

Connaitre le type des grandeurs utilisé.

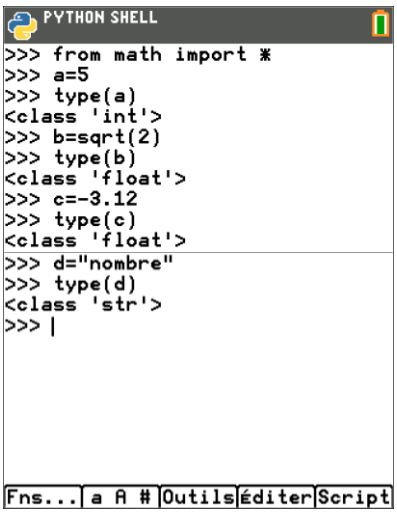
Lorsque vous utilisez un script en langage Python, il peut s'avérer nécessaire de connaitre le type de variable utilisé, ou bien de modifier ces variables en vue d'une utilisation ultérieure. Par exemple si la grandeur renvoyée par un script Python correspondant à une mesure en sciences physiques, il n'est pas nécessairement opportun de conserver un résultat à 6 décimales.

Vous allez créer un script dans l'application **Python** permettant de distinguer les types de grandeurs utilisées.

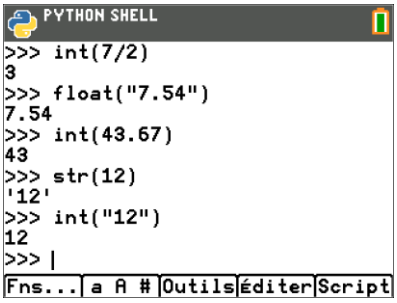
- Une chaine de caractère.
- Un nombre réel.
- Un nombre irrationnel, $\sqrt{2}$ par exemple.

Vous afficherez le nombre ainsi que son type à l'aide l'instruction **type**.

- Importer le module **maths (**Fns… puis ~ Modul puis 1 : math… et enfin **1 : from math import ***) puis appuyer sur la touche **entrer**.
- L'ensemble des commandes utilisé peut être obtenu en appuyant sur la touche **F1** Fns… puis en choisissant les sous menus **E/S** (entrée/sortie) et **Type**.
- La commande type qui permet de connaitre la nature d'une variable peut être tapée « à la main ».



Astuce : L'utilisation des touches de direction } permet de recopier une ligne lorsque l'utilisateur travaille dans la console.



Remarques :

- L'opérateur **int()** extrait lorsque c'est possible, un entier d'une chaîne de caractères et renvoie la partie entière d'un nombre comprise entre $\pm$ 2 147 483 648 (codage sur 32 bits, soit 4 octets).
- L'opérateur **str()** transforme un nombre en une chaine de caractères.
- L'opérateur **float()** extrait lorsque c'est possible, un flottant d'une chaîne de caractères.

Ce document est mis à disposition sous licence Creative Commons
http://creativecommons.org/licenses/by-nc-sa/2.0/fr/

Une autre astuce :

Pour incrémenter une variable dans un compteur, on dispose de deux possibilités :

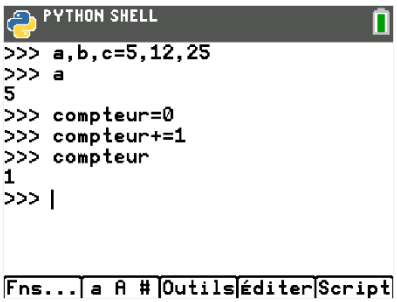
a) Écrire par exemple :

```
compteur = 0
compteur = compteur +1
compteur
```

demander l'affichage de la variable

b) Ou bien :

```
compteur = 0
compteur+=1
compteur
```



Conseils à l'enseignant : Penser à utiliser le clavier alphanumérique y f et f pour passer du mode majuscule à minuscule.
Le symbole « _ » est obtenu en utilisant la palette **a A #** accessible par la touche **F2.**
Utiliser les fonctions de « copier-coller » à partir du menu **F3** Outils.

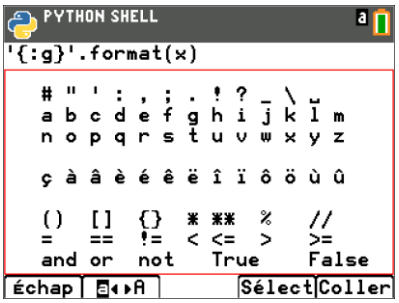
Lors de la rédaction, la touche { permet d'effacer un caractère entré par erreur.

A partir de la console (Shell), l'accès au catalogue y Ê permet d'accéder à toutes les fonctionnalités en Python disponibles dans la calculatrice.
Vous avez la possibilité de commenter vos scripts en ajoutant devant le commentaire, un (# commentaire). Le fait de mettre un # indique que la ligne ne sera pas interprétée. Pour l'obtenir, appuyer sur la touche **F2** (a A #) ou bien y Â

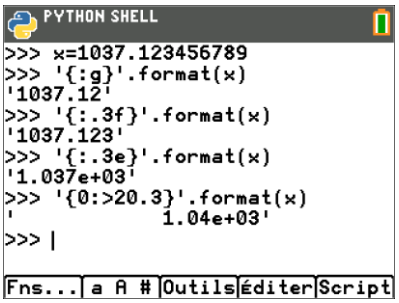
Ce document est mis à disposition sous licence Creative Commons
http://creativecommons.org/licenses/by-nc-sa/2.0/fr/

Pour aller plus loin :

Le format des nombres : La méthode format de l'objet string est un outil très puissant permettant de créer des chaines de caractères en remplaçant certains champs (entre accolades) par des valeurs (passées en argument de la fonction format) après conversion de celles-ci. On peut préciser à l'intérieur de chaque accolade un code de conversion, ainsi que le gabarit d'affichage. Donnons quelques exemples.




```
>>> x = 1037.123456789
>>> '{ :g}'.format(x) # choisit le format le plus approprié '1.04e+03'
>>> '{ :.3f}'.format(x) # fixe le nombre de décimales
'1037.123'
>>> '{ :.3e}'.format(x) # notation scientifique
'1.037e+03'
>>> '{0 :20.3f}'.format(x) # précise la longueur de la chaîne
' 1037.123'
>>> '{0 :>20.3f}'.format(x) # justifié à droite
' 1037.123'
>>> '{0 :<20.3f}'.format(x) # justifié à gauche
'1037.123 '
>>> '{0 :^20.3f}'.format(x) # centré
' 1037.123 '
>>> '{0 :+.3f} ; {1 :+.3f}'.format(x, -x) # affiche toujours le signe '+1037.123 ; -
1037.123'
>>> '{0 : .3f} ; {1 : .3f}'.format(x, -x) # affiche un espace si x>0
```



Appuyer sur la touche **F2** (**a A #)** afin d'accéder à la table de caractères

Ce document est mis à disposition sous licence Creative Commons
http://creativecommons.org/licenses/by-nc-sa/2.0/fr/