Interagir avec l'utilisateur

Il est très courant dans un script d'avoir besoin à un moment ou un autre de demander des informations à l'utilisateur.

On pourrait par exemple demander la nouvelle taille d'une image qu'on souhaite modifier, l'URL d'un site web ou encore un chemin de dossier dans lequel on veut écrire des données.

Pour demander des informations à un utilisateur dans un script en ligne de commande, on utilise avec Python la fonction `input`.

La fonction input

La fonction input accepte un argument qui correspond au texte que l'on souhaite afficher à l'utilisateur.

Par exemple, si on souhaite demander l'âge de l'utilisateur :

```
>>> input("Quel âge avez-vous ? ")
Quel âge avez-vous ?
```

L'utilisateur peut ensuite écrire à la fin de la phrase affichée.

La saisie de l'utilisateur est ensuite retournée par la fonction input :

```
>>> input("Quel âge avez-vous ? ")
Quel âge avez-vous ? 25
'25'
```

En l'état actuel, cela ne sert pas à grand-chose car nous ne récupérons par cette valeur dans une variable.

Pour ce faire, il suffit de faire une affectation simple lors de l'utilisation de la fonction input.

Une fois les données renseignées par l'utilisateur, la valeur retournée sera affectée dans la variable indiquée :

```
>>> age = input("Quel âge avez-vous ? ")
Quel âge avez-vous ? 25
>>> print(age)
25
```

Depuis la version 3 de Python, la fonction input retourne **systématiquement** une chaîne de caractères.

Ainsi, la variable `age` dans l'exemple ci-dessus ne sera pas considérée comme un nombre entier mais comme une chaîne de caractères contenant le nombre 25, ce que l'on peut vérifier avec la fonction type :

```
>>> age = input("Quel âge avez-vous ? ")
Quel âge avez-vous ? 25
>>> type(age)
<class 'str'>
```

Si vous souhaitez convertir la variable `age` en nombre entier, il faudra donc utiliser la fonction de conversion `int`.

Une petite note sur Python 2

Avant l'apparition de la version 3 de Python, il existait deux fonctions pour récupérer des données de l'utilisateur : la fonction `input` et la fonction `raw_input`.

La fonction raw_input de Python 2 est l'équivalent de la fonction input de Python 3. Cette fonction retournait systématiquement une chaîne de caractères.

La fonction input avec Python 2 quant à elle interprétait le résultat donné par l'utilisateur.

Ainsi, quand l'utilisateur renseignait un nombre entier par exemple, la fonction input retournait directement un objet de type nombre entier.

Ce problème ne se pose plus avec Python 3 qui n'a conservé que la fonction `raw_input` d'origine et l'a renommé en '`input`'.

Ainsi, avec Python 3, seule la fonction input demeure, et elle vous retournera systématiquement une chaîne de caractères.

Cela peut sembler être un inconvénient (on aimerait bien recevoir directement un nombre quand l'utilisateur nous envoie un nombre).

Dans la pratique, cela permet de redonner le contrôle au développeur.

Les données retournées par la fonction input sont ainsi constantes (chaîne de caractères), et c'est au développeur de valider les données par la suite et, s'il le souhaite, de changer le type de la variable.

Python ne se charge donc plus de réaliser des opérations à notre place, opérations qui dans certains cas pouvaient être problématiques, par exemple si l'on envoyait du texte à la fonction input sans l'entourer de guillemets (la fonction input interprétant le résultat comprenait ainsi le texte saisi par l'utilisateur comme une variable et non pas une chaîne de caractères) :

```
>>> prenom = input("Quel est votre prénom ? ")
Quel est votre prénom ? Thibault
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'Thibault' is not defined
```

☝️ Ici, Python évalue `Thibault` comme une variable, variable qui n'existe pas dans notre script. On obtient donc une erreur de type `NameError`.