NSI, TP N°1 : VARIABLES, OPÉRATIONS, BOUCLES.

1. TROIS PREMIERS TYPES DE VARIABLES

On distingue en programmation différents types de variables : les nombres entiers, les nombres réels, les chaînes de caractères .....
Python gère dynamiquement le typage des données contrairement à d'autres langages où il est nécessaire de préciser le type de la variable lorsque l'on la déclare.

A. Affecter une valeur à une variable, retourner la valeur d'une variable

Tapons les deux dernières commandes des commandes suivantes dans la console Python de rep.it :

```
> # le symbole #
> # permet d'écrire des commentaires dans les programmes en Python
> # ces commentaires ne sont pas éxécutés comme lignes de commandes
> a=3 # On affecte à a la valeur 3
> a # renvoie la valeur contenue par a
```

Pour savoir de quel type est la variable a, on utilise la commande type() :

```
> type(a) # renvoie le type de la variable a
```

a est une variable de class int puisqu'il s'agit d'un entier. On peut définir des variables contenant des caractères :

EXERCICE 1.

On considère les commandes suivantes :

```
> b = "Python est langage avec un typage dynamique." # b est une chaîne de
caractères
> c = 5/3 # la variable c est de type réel
```

Déterminer le type des variables b et c en utilisant la commande adéquate.

En résumé, on distingue les entiers (class int), des flottants (class float) des chaînes de caractères (class str pour string).

B. Opérations sur les variables.

On peut effectuer différentes opérations mathématiques ou logiques sur les variables.

```
> a=2
> b=5
> c=a+b # addition
> d=a*b # multiplication
> e=a**b # puissance
> f=a/b # division décimale
> g=b%a # reste de la division euclidienne
> h=b//a # quotient de la division euclidienne
```

EXERCICE 2.

1. Déterminer à l'aide python le contenu des variables c,d,e,f,g et i.
2. Quel est le type de ces variables ?

On peut également effectuer des opérations sur les chaînes de caractères :

```
> a="Python est un"
> b="langage de programmation"
> c=a+b
```

Afficher le contenu de la variable c.

Attention

Il faut faire attention aux opérations sur des variables de types différents :

```
> a=3
> b="azer"
> a+b
```

EXERCICE 3.

Quelques opérations plus spécifiques :

```
> a=6
> a+=3 # on incrémente a de 3
> b=5
> b-=2 # on décrémente b de 2
```

Déterminer les valeurs des variables A et B après ces lignes de commandes.

EXERCICE 4.

On peut également utilser des opérateurs logiques et découvrir une nouvelle classe de variables :

```
> a=2
> b=3
> c=5
> d=(a==b) # égalité de 2 variables
> e=not(a==b) # différence de 2 variables que l'on peut noter aussi e=(a!=b)
> f=(a+b==c)
```

1. Déterminer la valeur des variables d,e et f.
2. Déterminer le type de ces 3 variables.

C. Faire attention.

Python est sensible à la casse (c'est à dire qu'il distinge majuscule et minuscule).
Les commandes suivantes déclencheront des erreurs.

```
> Delta=3
> b=delta+2
```

Les nombres décimaux se notent bien avec un .

```
> a=3.2
> b=3,5
> a+b
```

Nous reviendrons plus tard sur la classe des tuples en Python.

2. BOUCLES.

A. Boucles for.

Ecriver les programmes suivants puis exécuter les.

```
for i in range(10):
    print(i)
```

```
for i in range(3,8):
    print(i)
```

```
for i in range(2,12,3):
    print(i)
```

```
for i in range(10,5,-1):
    print(i)
```

```
a="abcdez"
for i in a :
    print(i)
```

En python, les chaînes de caractères sont considérées comme un certain type de collections d'objets, en l'occurrence des caractères.
On peut donc accéder aux éléments de cette collection.

```
> a="azerty"
> a[0] # retournera le 1° caractère de la chaîne , d'indice 0, soit 'a'
> a[2] # retournera le 3° caractère, soit 'e'
> a[-1] # retournera le dernier caractère soit 'y'
> len(a) # retournera la longueur de la chaîne de caractères.
```

La facilité pour Python de parcourir une chaîne est assez propre à ce langage.
On a souvent recours aux passages par les indices pour d'autres langages.

```
a="abcdez"
for i in a :
    print(i)
```

donnera le même résultat que :

```
a="abcdez"
for i in range(len(a)) :
    print(a[i])
```

B. Boucles while.

Les boucles while forment ce que l'on appelle des boucles conditionnelles.
Le contenu de la boucle est effectué tant que la condition est vérifiée.

```
a=2
while a<10 : # tant que a<10, on effectue le contenu de la boucle
    a+=2 # on incrémente a de 2
    print(a) # on affiche a
print("cest fini",a) # on effectue cette instruction à la sortie de la boucle
```