SNT Python 1 - Types de données

1 – Thonny

HTML et **CSS** sont **des langages de description**.

Scratch est un **langage de programmation par bloc**.

Python est un **“ ” par lignes de code**.

Thonny permet de programmer en Python et d’interagir en direct avec l’utilisateur (dans la zone console avec les >>>).

Les textes en violet ne doivent pas être recopiés !

2 – Le type int : integer / entier

Type de données correspondant aux entiers.
Voici les opérateurs classiques : + - * / et exposant avec **

```
>>> 7 + 2
9
```

```
>>> 7 * 2
14
```

```
>>> 7 / 2
3.5
```

```
>>> 10 ** 3
1000
```
$= 10^3 = 10*10*10$

Les opérateurs de la division euclidienne :

→ Quotient

```
>>> 13 // 5
2
```

```
>>> 13 // 6
2
```

→ Reste

```
>>> 13 % 5
3
```

```
>>> 13 % 6
1
```

13	5
-10	2
3	

13	6
-12	2
1	

Priorités

D’abord : ()
Ensuite **
Ensuite * / % //
Puis + -

Exemple :

```
   4 + 2**3 * 2
= 4  +   8   * 2
= 4   + 16
= 20
```

3 – Le type str : string, chaîne de caractères

Le type texte se nomme string (str). Il est délimité par des guillemets.

COLLER et COMPLETER LE DOCUMENT n°1

Concaténation de str avec +

```
>>> "Bon" + "jour"
'Bonjour'

>>> "5" + "2" + "3"
'523'
```

Répétition d'un str avec *

```
>>> "Bon" * 2
'BonBon'

>>> "Bon" + "jour " * 2
'Bonjour jour '
```

Suppléments

```
>>> "Bo" + "um " * 3
'Boum um um '
```

```
>>> ("Bo" + "um") * 3
'BoumBoumBoum'
```

Attention aux types des données :

```
>>> 5 * 2
10


>>> "5" * 2
'55'
```

COLLER et COMPLETER LE DOCUMENT n°2

Fonction **chr()** : renvoie le caractère correspondant au code fourni .
Fonction **ord()** : renvoie le code du caractère fourni entre parenthèses.
Fonction **len()** : renvoie le nombre d'éléments dans la donnée fournie.
Fonction **print()** : ne renvoie rien mais affiche et interprète l'entrée.

```
>>> chr(65)
'A'
>>> chr(66)
'B'

>>> chr(10)
'\n'
```

```
>>> ord('A')
65
>>> ord('B')
66

>>> ord('\n')
10
```

```
>>> len("Bob")
3
>>> len("B o b")
5

>>> len("Bo\nb")
4
```

Notez bien que \n représente 1 caractère : le passage à la ligne.

```
>>> print('Bob')
Bob

>>> print('Bo\nb')
Bo
b

>>> print('abc\nABC\nZ')
abc
ABC
Z
```

4 – Le type bool : True ou False

Le type booléen ne possède que deux valeurs possibles :

→ True pour dire vrai

→ False pour dire faux.

Les opérateurs booléens de cette partie ont une priorité plus petite que les opérateurs arithmétiques (* + ...)

COLLER et COMPLETER LE DOCUMENT n°3

```
>>> 8 > 10
False
```

```
>>> 8 >= 5+3
True
```

```
>>> 8 == 5 + 3
True
```

```
>>> 8 != 5 + 3
False
```

```
>>> 'on' in 'Bonbon'
True
```

```
>>> 'ON' in 'Bonbon'
False
```

```
>>> len('on') > len('Bonbon')
False
```

Explications (inutiles de les noter une fois compris)

>>> 8 > 10 : 8 est il strictement supérieur à 10 ? Non donc False.

>>> 8 >= 5+3 : 8 est-il supérieur ou égal à (5+3) ? Oui car 8 est égal à (5+3)

>>> 8 == 5 + 3 : 8 est-il égal à (5+3) ? Oui, 8 est égal à (5+3)

>>> 8 != 5 + 3 : 8 est-il différent de (5+3) ? Non, 8 n'est pas différent de (5+3)

>>> 'on' in 'Bonbon' : **'on'** est-il présent dans 'B**on**b**on**' ? Oui : True

>>> 'ON' in 'Bonbon' : **'ON'** est-il présent dans 'B**on**b**on**' ? Non : False

>>> len('on') > len('Bonbon')

La longueur de **'on'** est-elle strictement supérieure à la longueur de **'Bonbon'** ? Non donc 2 n'est pas supérieur à 6. Donc False.

COLLER et COMPLETER LE DOCUMENT n°4

Bilan des opérateurs à connaître

+ est l'opérateur addition ou concaténation

*** est l'opérateur multiplication ou répétition**

/ est l'opérateur division

// est l'opérateur quotient de la division euclidienne

% est l'opérateur reste de la division euclidienne

in est l'opérateur appartenance

== est l'opérateur d'égalité

!= est l'opérateur de différence

>, >=, < et <= sont les opérateurs de comparaison

5 – Exemples de gestion de données [PARTIE OPTIONNELLE]

Fonction **str()** : renvoie une copie des données sous forme d'un string.

```
>>> str(5)
'5'
```

```
>>> str(5)*4
'5555'
```

Fonction **int()** : renvoie une copie des données sous forme d'un entier.

```
>>> int("5")
5
```

```
>>> int("5")*4
20
```

La fonction **split()** renvoie un tableau contenant les différents morceaux **du string placé devant le point** en utilisant le caractère de séparation fourni.

COLLER et COMPLETER LE DOCUMENT n°5

```
>>> "Bonjour à tous".split(" ")
['Bonjour', 'à', 'tous']
    0         1      2

>>> "Bonjour à tous".split(" ") [0]
'Bonjour'

>>> "Bonjour à tous".split(" ") [2]
'tous'

>>> "80.100.15.3".split(".")[1]
'100'

>>> int( "80.100.15.3".split(".")[3] )
3
```

6 – Séquentialité dans la console [PARTIE OPTIONNELLE]

L'interpréteur exécute les lignes une par une.

Tant qu'il n'a pas fini une ligne, il ne passe pas à la suivante.

Console interactive

L'invite de commande est notée >>>

Impossible de modifier une ligne déjà exécutée.

En plaçant votre souris sur la console, on peut récupérer les dernières instructions avec la flèche UP.