Type booléen en Python

1. Valeurs et Type

En Python, 2 valeurs sont définies : True et False, de type bool.

Illustration :

```
>>> True
True
>>> print(type(True))
<class 'bool'>
>>> False
False
>>> print(type(False))
<class 'bool'>
```

2. Opérateurs de comparaison

Les opérateurs de comparaison courants sont identiques à ceux de l'arithmétique.

Le résultat d'une comparaison donne une valeur booléenne.

ATTENTION, il ne faut pas confondre l'égalité et l'affectation

```
>>> variable = 5    # une affectation
>>> 5 == 8          # une égalité (qui est fausse)
False
```

2.1. Liste des opérateurs de comparaison

Type d'opérateur	Opérateur	Exemple	Résultat
Égalité	==	1 + 2 == 3	True
Différence	!=	1 + 2 != 3	False
Supérieur	>	4 > 3	True
Inférieur	<	2.2 < 2 * 3	True
Supérieur ou égal	>=	5 >= 6	False
Inférieur ou égal	<=	8 <= 3	False

2.2. Appartenance à une structure

Il est possible de tester l'appartenance d'un élément à une structure avec le mot clé in.

Illustration :

```
>>> "a" in "bonjour"       # False
False
>>> "bon" in "bonjour"     # True
True
>>> 1 in [2, 3, 4]         # False
False
```

3. Opérations sur les booléens

Les opérateurs sur les booléens sont de deux types :

- opérateur **unaire** : prend en paramètre un booléen, ou une expression booléenne et renvoie un booléen.
- opérateur **binaire** : prend en paramètre deux booléens, ou deux expressions booléennes et renvoie un booléen.

Type d'opérateur	Opérateur	Exemples

Négation	not	>>> not True
		False
		>>> not False
		True

OU	or	>>> False or False
		False
		>>> False or True
		True
		>>> True or False
		True
		>>> True or True
		True

ET	and	>>> False and False
		False
		>>> False and True
		False
		>>> True and False
		False
		>>> True and True
		True

OU EXCLUSIF	^	>>> False ^ False
		False
		>>> False ^ True
		True
		>>> True ^ False
		True
		>>> True ^ True
		False

4. Notion de cast : conversion explicite

Python permet de convertir différents types Python en booléen.

Illustration :

```
>>> bool(0)
False
>>> bool(1)
True
>>> bool(2)
True
>>> bool("")
False
>>> bool("abc")
True
>>> bool([])
False
>>> bool([1, 2])
True
```

Bilan :

- 0 est Faux, tout autre entier est Vrai,
- Une structure vide est Faux, toute autre valeur de structure est Vrai.

5. Complément : None et l'opérateur d'identité is

Python propose la valeur None (rien) qui est fréquemment utilisé pour représenter l'absence d'une valeur, qui est différent d'une valeur booléenne.

Étant le seul objet du type NoneType, on peut tester son identité avec is :

```
>>> True is None
False
>>> False is None
False
>>> 1 is None
False
>>> "abc" is None
False
>>> None is None
True
>>> a = 5
>>> a is None
False
```

Une fonction python sans return... renvoie néanmoins la valeur None.