Types intégrés (built-in)

Entier `int`

```
>>> a = 5
>>> type( a )
<class 'int'>
>>> a = 3546879442324561021588971316549872316576543214687
>>> a
3546879442324561021588971316549872316576543214687
>>> type( a )
<class 'int'>
```

En Python 3, le type entier (`int`) n'a pas de limite. La grandeur d'un nombre entier, nombre de chiffres, n'a aucune limite, mises à part les performances et la mémoire disponible.

Réel `float`

En programmation d'origine anglophone, le point remplace notre virgule pour séparer la partie décimale de la partie entière.

```
>>> a = 5.
>>> type( a )
<class 'float'>
>>> print( a )
5.0
>>> a = .5
>>> print( a )
0.5
>>> a = 3.14159265358979323846
>>> a
3.141592653589793
```

N. B. que nous ne sommes pas obligé d'écrire l'éventuel zéro (« non significatif ») avant ou après le point.

N. B. que nous ne retrouvons donc pas toutes les décimales renseignées lors de l'affectation.
(16 chiffres significatifs)

Le type réel (float) est soumis à une limite dans le nombre de chiffres significatifs (16) et également une limite quant « au nombre de zéros avant ou après » (± 300 zéros)…

Un float peut également avoir les valeurs *inf*, *-inf* et *nan* pour respectivement infini, moins l'infini et "pas un nombre".

```
>>> a = 1e300 # ← écriture scientifique
>>> b = 1.*10**300 # ← 10 exposant 300
>>> a == b
True
>>> print( a, b )
1e+300 1e+300
>>> a *= 1_000_000_000
>>> a
inf
>>> a = float("nan") # conversion str → float
>>> a
nan
```

N. B. les blancs soulignés ("underscores") dans le nombre, ici en guise de séparateur de milliers.