



Fondamentaux Python : Variables et Types

De la simple calculatrice à la science des données reproductible.

Python est une excellente calculatrice... mais sans mémoire.

Vous pouvez effectuer des calculs complexes, mais une fois l'opération terminée, le résultat est perdu. Python ne "se souvient" pas des valeurs précédentes à moins que nous ne lui demandions explicitement.

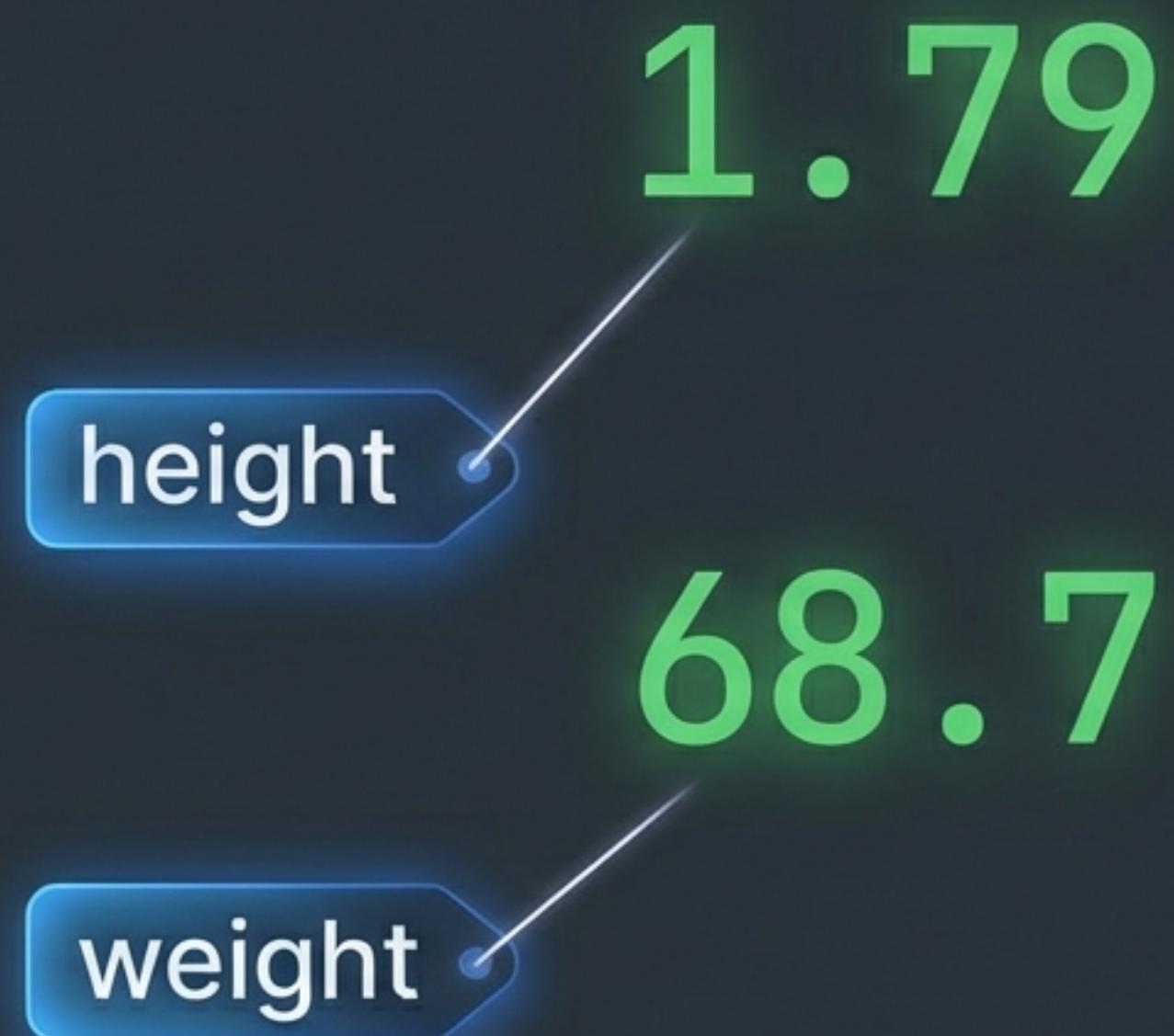
```
>>> 68.7 / 1.79 ** 2  
21.441278  
21.441278
```



La solution : L'assignation de variables

Pour sauvegarder une valeur pendant que vous codez, vous devez la définir avec un nom spécifique (sensible à la casse). On utilise le signe égal `=` pour assigner une valeur à ce nom.

```
height = 1.79  
weight = 68.7
```



L'appel de variable : Une référence immédiate

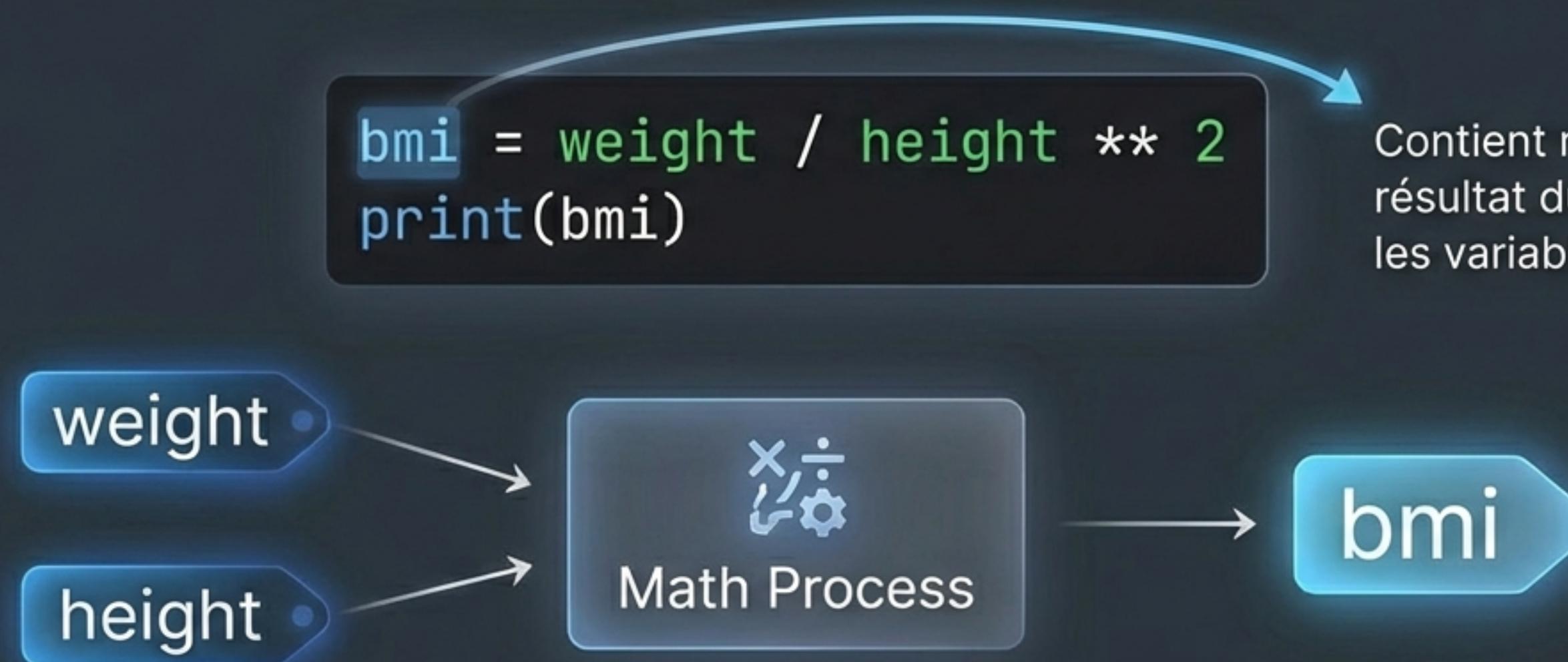
Une fois déclarée, vous pouvez appeler la valeur simplement en tapant le nom de la variable. Python recherche ce nom, récupère la valeur associée et l'utilise.



Chaque fois que vous tapez le nom, vous demandez à Python de référencer la valeur réelle.

Application concrète : Le calcul du BMI (IMC)

Nous pouvons utiliser ces variables directement dans une formule mathématique.
Ici, nous stockons le résultat du calcul dans une *nouvelle* variable.



La puissance du script : La reproductibilité

Les variables rendent votre code reproductible. Si vous changez la définition de la variable `weight` et relancez le script, le `bmi` se met à jour automatiquement.



“C'est l'essence de l'automatisation en Data Science.”

La nature de la donnée : Les Types

Jusqu'ici, nous avons manipulé des nombres. Mais en Python, chaque valeur possède un type spécifique. C'est sa "catégorie".



```
type(bmi)
```

Console ×

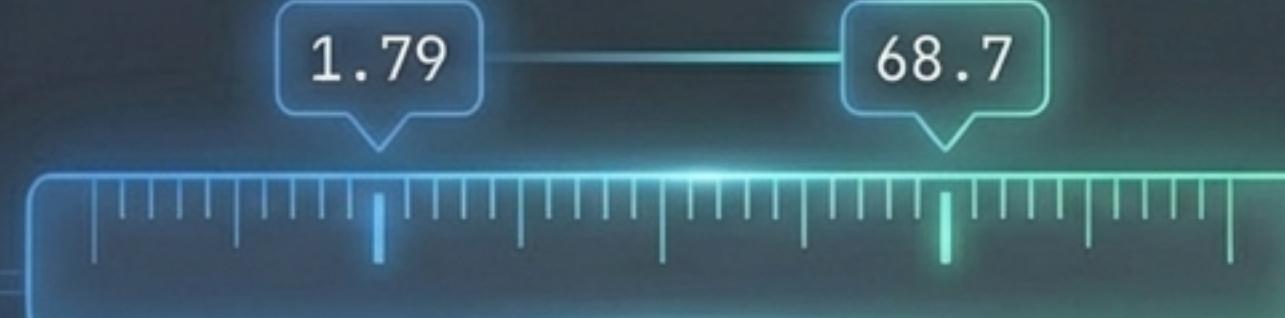
```
<class 'float'>
```

La fonction `type()` nous révèle comment Python interprète la donnée.

Les nombres : `float` vs `int`

float

Représente un nombre réel. Il possède une partie entière et une partie fractionnaire (décimale).



1.79 → 68.7

int

Représente un nombre entier, sans virgule.



42



100

Au-delà des nombres : Le texte (`str`)

Pour faire de la science des données, il faut plus que des chiffres. Le type `str` (string/chaîne de caractères) permet à Python de représenter du texte.

The diagram features a central code block with a glowing blue and green border, set against a dark background with faint circuit board patterns. Inside the box, two lines of Python code are shown:

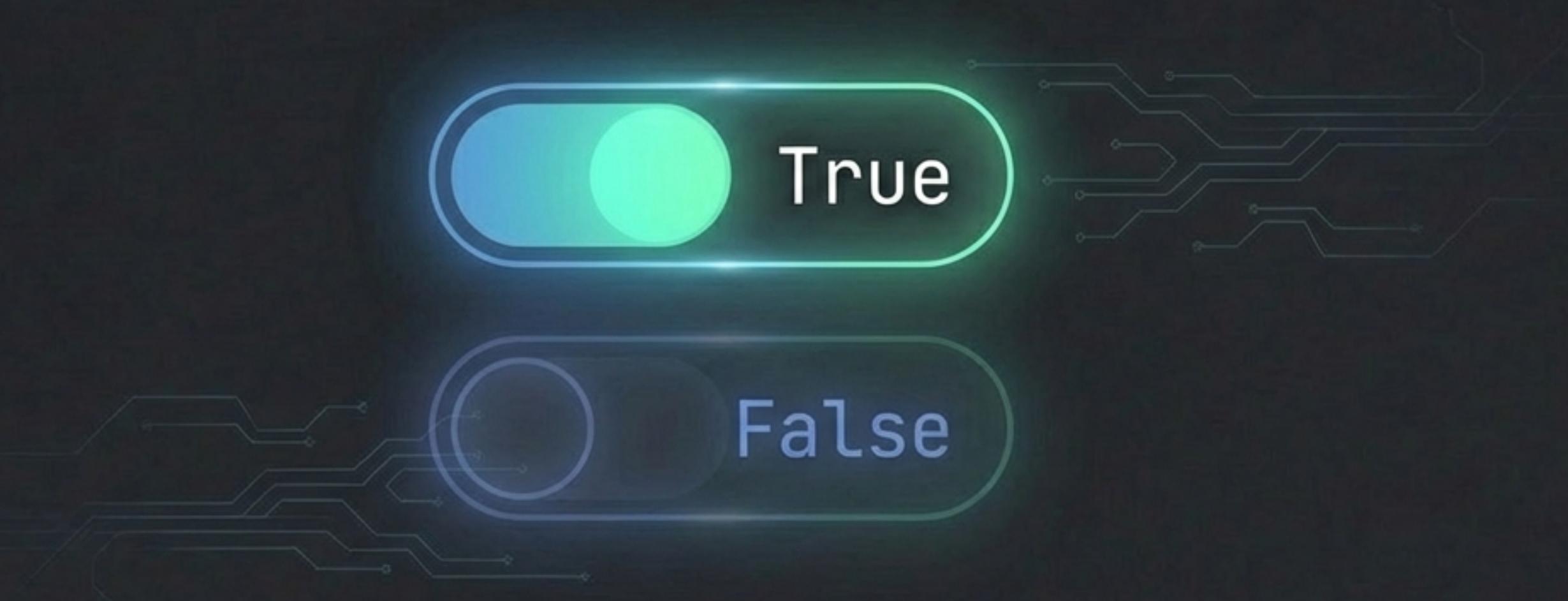
```
x = "body mass index"  
y = 'this is also a string'
```

Both the assignment operators (=) and the string literals ("body mass index" and 'this is also a string') are circled in red. A red arrow points from the text "Les deux syntaxes sont valides." to the right side of the code box.

Les deux syntaxes
sont valides.

La logique pure : Les Booléens (`bool`)

Le type `bool` ne peut être que `True` (Vrai) ou `False` (Faux). Considérez cela comme le "Oui" et "Non" du langage informatique.



Ces types seront essentiels plus tard pour effectuer des opérations de filtrage sur vos données.

Règle fondamentale : Le type dicte le comportement

Il y a quelque chose de spécial concernant les types de données en Python : les opérateurs (comme `+`) ne fonctionnent pas toujours de la même manière.



La façon dont le code se comporte dépend
du **type** des objets que vous manipulez.

Preuve visuelle : L'opérateur `+`

$2 + 3$



Somme mathématique

"ab" + "cd"



Concaténation (Collage)

Pour les entiers, les valeurs sont additionnées. Pour les chaînes, elles sont "collées" ensemble.

En Résumé

	Variables	Attribution avec `=` (JetBrains Mono), sensible à la casse (ex: `bmi` (JetBrains Mono)).
	float	Nombres réels (ex: `1.79` (JetBrains Mono)).
	int	Nombres entiers.
	str	Texte, utilise `\" \"` ou ' ' (JetBrains Mono).
	bool	Logique (True/False (JetBrains Mono)), utile pour filtrer.
	Comportement	Les opérateurs changent selon le type.

Prêt à coder ?

Vous maîtrisez maintenant les blocs de construction de base. Dans les prochains exercices, vous créerez vos premières variables et expérimenterez avec ces types.

Prochaine étape : Les Listes.

