Nom :	Prénom :	Classe :

SNT — Type de donnée : entier, flottant, booléen, chaîne

Objectifs :
- Savoir identifier un type de donnée élémentaire, en Python.
- Connaître les avantages et limites des types de données élémentaires.
- Savoir affecter un type de donnée élémentaire à une variable.

Qu'est-ce qu'un type de donnée ?

Dans la vie de tous les jours, nous utilisons des informations de différents « types » (différentes *natures*) :

- Une information de type "texte" comme « Bonjour ! ».
- Une information de type "nombre entier" comme « 453 ».
- Une information de type "nombre décimal" comme « 453,12 »

En **Python** c'est la même chose, une information (ou **donnée** ou **valeur**) peut être de différents types.

Principaux types de données en Python

Voici une **liste des principaux types de données** en Python :

Nom	Type de donnée Python	Exemples Python
Nombre entier	`int`	`12` ou `492` ou `-15`
Nombre décimal	`float`	`12.3` ou `-2.2` ou `3.14159`
Texte	`str`	`'Salut'` ou `'oui'` ou `"18"`
Interrupteur	`bool`	`True` ou `False`

➤ **Exercice 1** — À vous de jouer !

À votre avis, quel est le **type de donnée Python** de chacune des **valeurs** suivantes :

Valeur	Type de donnée Python
`'Namaste!'`	...
`True`	...
`10.1`	...
`'64'`	...

Valeur	Type de donnée Python
`12 + 5`	...
`"Faux"`	...
`10 / 3`	...
`10 // 3`	...

int — le type dédié aux nombres entiers

En Python, le type `int` sert à représenter les **nombres entiers**, comme `-5`, `0`, `12` ou `2026`.

int signifie « entier » (de l'anglais *integer*) et représente tous les nombres sans virgule, positifs ou négatifs. Ce type de donnée permet de faire dessus toutes les opérations classiques : addition, soustraction, multiplication, division entière, puissance, etc[1].

➤ Exercice 2

Déterminez si les valeurs suivantes sont de type `int` ou non :

Valeur	int ? (marquez oui ou non)
`14864532154845`	...
`-5462114458542`	...
`12354 + 548914`	...
`488545 // 2621`	...

Valeur	int ? (marquez oui ou non)
`5488545 % 2621`	...
`6532 ** 425623`	...
`564215599445.4`	...
`'845652545521'`	...

float — le type dédié aux nombres décimaux

En Python, le type `float` sert à représenter les **nombres décimaux**, comme `3.14` ou `-0.5`, c'est-à-dire les nombres « à virgule » (**écrits avec un point** en informatique, par exemple `2.5`).

Le mot anglais *float* signifie « flottant » et représente tous les nombres avec virgule, positifs ou négatifs.

En Python, **tout nombre contenant un point** (`2.0`, `0.5`, `2.`) est automatiquement de type `float`.

Notons également qu'une division « simple », utilisant le symbole `/`, retourne toujours un nombre décimal, donc un `float`.

➤ Exercice 3

Déterminez si les valeurs suivantes sont de type `float` ou non :

Valeur	float ? (oui ou non)
`65424453115.1`	...
`10 / 2`	...
`2.1`	...
`2.0`	...

Valeur	float ? (oui ou non)
`12 + 10.1`	...
`10 // 2`	...
`10 % 2`	...
`'452.12'`	...

1 Voir le cours « Premiers pas en Python : calculs, commentaire, variable ».

str — le type dédié aux textes

Le type `str` en Python représente une **chaîne de caractères**, c'est-à-dire une séquence de lettres, chiffres ou symboles comme `"Bonjour"` ou `'Python'`.

Une chaîne de caractères (aussi appelée simplement une **chaîne**), est une suite de caractères **délimitée** par des **guillemets simples** `'`, **doubles** `"`.

Quelques exemples de chaînes de caractères

Voici quelques exemples de chaînes de caractères :

```
1  'Je suis une chaîne de caractères utilisant des guillemets simples'
2  "Je suis une chaîne de caractères utilisant des guillemets doubles"
3  "Une chaîne peut contenir toutes sortes de caractères : 😀 ?! Δ @"
```

On remarque qu'une chaîne doit **commencer** et **terminer** par la même sorte de guillemet : simple ou double.

Spécificités

Attention : un nombre (entier ou décimal) entouré de guillemets est de type `str` !

Exemples :

```
1  4562   # Ce nombre ne possède pas de guillemets, il n'est pas de type « str »
2  45.62  # Idem, il n'est pas de type « str »
3
4  '4562' # Ce nombre possède des guillemets, donc il est de type « str »
```

➤ Exercice 4

Déterminez si les valeurs suivantes sont de type `str` ou non :

Valeur	str ? (oui ou non)
`"a"`	...
`12.452`	...
`'3.141592654'`	...
`"Werenoi"`	...

Valeur	str ? (oui ou non)
`'नमस्ते'`	...
`"😍"`	...
`"Python = ❤️"`	...
`145 // 12`	...

bool — le type soit vrai, soit faux

Le type `bool` en Python représente un « booléen », une donnée qui ne peut prendre que deux valeurs : `True` (ce qui se traduit par "vrai" en français) ou `False` (faux, en français).

C'est un type de donnée qui sert à exprimer des états logiques « binaires », comme oui/non ou allumé/éteint.

Dans la page 1 de ce présent cours, nous avons volontairement dénommé ce type de donnée comme étant un type « interrupteur », car il ne peut avoir que deux états : True (allumé) ou False (éteint).

➤ Exercice 5

Déterminez si les valeurs suivantes sont de type `bool` ou non :

Valeur	bool ? (oui ou non)
`True`	...
`"True"`	...
`3.141592654`	...
`"NSI"`	...

Valeur	bool ? (oui ou non)
`False`	...
`"Faux"`	...
`Faux`	...
`145 // 12`	...

Affecter un type de donnée à une variable

En Python, une **variable**[2] prend automatiquement le **type de la donnée** qui lui est **affectée**.

Par exemple, lors de l'affectation `x = 5`, Python crée une variable `x` de type `int`.

```
1  x = 5       # Ici la variable x est de type « int »
2  x = x + 1   # La variable x est toujours de type « int »
3
4  x = x / 2   # La variable x devient de type « float »
5
6  x = 'Yeah'  # La variable x devient de type « str »
7
8  x = 3       # La variable x devient de type « int »
```

Dans le programme Python ci-dessus nous voyons que la variable x prend différents types de donnée (selon la ligne de code du programme).

- Si la donnée affectée est de type `int`, alors la variable est également de type `int` (lignes 1, 2 et 8).
- Si la donnée affectée est de type `float`, alors la variable est également de type `float` (ligne 4).
- Etc.

2 Voir le cours « Premiers pas en Python : calculs, commentaire, variable ».

➤ **Exercice 6** — Le grand final !

Déterminez le type de donnée Python de la variable `a`, **<u>à la fin</u>** de chacun des programmes suivants :

N°	Programme Python	Type de donnée de `a` (à compléter)
1	`a = True`	

2	`a = 10`
	`b = 2`
	`a = a / b`	

3	`a = 125`
	`b = 10`
	`a = a + 2`
	`a = '127'`	

4	`a = 8`
	`b = 12.3`
	`a = a + b`	

5	`a = 145 // 10` `a = a + 1`	
6	`a = False` `a = 'True'`	
7	`a = '☣'`	
8	`a = 145 ** 2 // 10`	
9	`a = 145 ** 2 / 1`	

10	`ahmed = 10`
	`celia = 1.1`
	`bob = 12`
	`alice = 14`
	`jade = 14`
	`jules = False`
	`camille = 18`
	`a = ahmed + jade + celia + bob + camille`	

Ce qu'il faut retenir

Vous retrouverez ici l'essentiel du cours.
Astuce : transformez ces points en questions/réponses pour vos flashcards (Anki).

Dans la vie de tous les jours, nous utilisons des informations de différents « types » (différentes natures). Dans un langage de programmation comme Python c'est pareil : une information (appelée également "donnée" ou "valeur") peut être de **différents types** :

- Le **type** `int` sert à représenter tous les **nombres entiers** (donc sans virgule) **positifs** ou **négatifs**.
- Le **type** `float` (flottant) sert à représenter tous les **nombres avec virgule**, **positifs** ou **négatifs**.
  - En Python, les nombres flottants n'utilisent **pas des virgules** mais des **points**.
  - Tout nombre contenant un point est automatiquement de type `float`.
  - Attention : Une **division simple** utilisant le symbole `/` retourne toujours un `float` !
- Le **type** `str` en Python représente une chaîne (ou "chaîne de caractères"), c'est-à-dire une séquence de lettres, chiffres ou symboles.
  - Une chaîne commence toujours par un **guillemet** simple (ou double), et se termine avec le **même guillemet** (simple ou double).
  - Attention : Un nombre entier ou un nombre flottant entouré de guillemets devient un `str` !
- Le **type** `bool` en Python représente un « booléen », une donnée qui ne peut prendre **que deux valeurs** : `True` (vrai) ou `False` (faux).
- Une **variable** prend automatiquement le type de la donnée qui lui est affectée.