Les variables

Programmation en Sciences de la nature

Des contenants étiquetés pour ranger les valeurs

Plan

Des bols et des verres

Une variable, c'est un contenant étiqueté

Le signe = se lit « affecter » : on verse une valeur

- Changer le contenu
- Bien nommer ses contenants
- Chaque valeur a un type

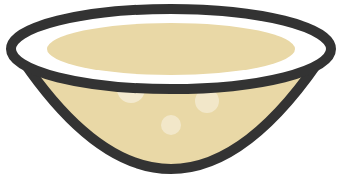
La mémoire, pas à pas

Le défi : échanger deux verres

Les contenants

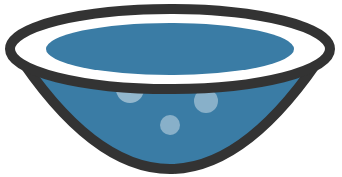
DES BOLS, DES VERRES ET DES ÉTIQUETTES

D'abord, la cuisine et le bricolage



`farine`

`sucre`



`eau`

Pour cuisiner ou bricoler, on prépare des **bols étiquetés** : un nom collé dessus, un contenu à l'intérieur.

On dit « passe-moi le bol de *sucre* » sans regarder dedans : **le nom suffit** pour retrouver le contenu.

Une variable, c'est un contenant étiqueté

`temperatureCelsius`

Bol ou verre, même idée.
On dessinera des verres : on voit le niveau.

Une **variable**, c'est une **étiquette** collée sur un contenant qui garde une **valeur** en mémoire.

- un **nom** : l'étiquette
- un **contenu** : la valeur
- un contenant garde **une seule valeur** à la fois

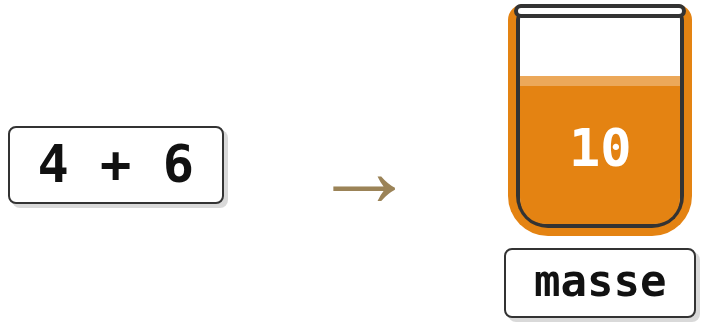
```
# on verse 25 dans le verre étiqueté
temperatureCelsius = 25
```

Le programme dit ensuite « *temperatureCelsius* » et l'ordinateur va chercher le contenu, comme on va chercher le bol de sucre.

Le signe = se lit « affecter », pas « égale »

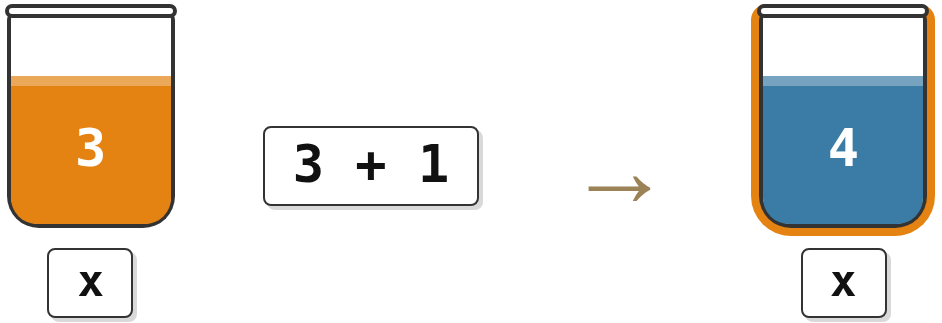
```
masse = 4 + 6
```

1. On calcule d'abord la **droite** : 4 + 6 donne 10
2. On **verse** le résultat dans le verre de **gauche**

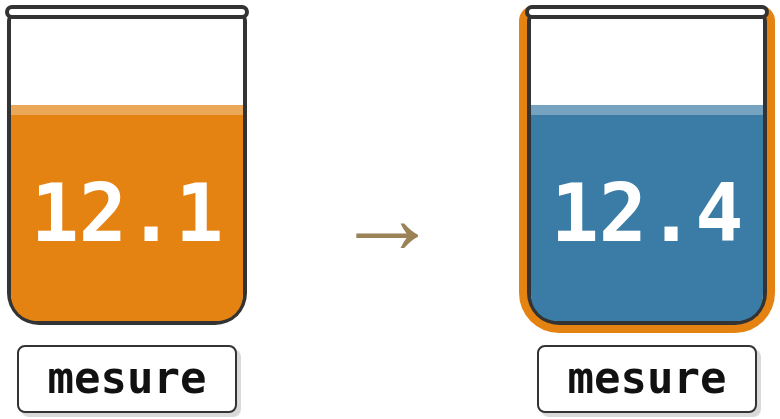



```
x = x + 1
```

Ce n'est pas une équation impossible : on prend le contenu de x, on ajoute 1, puis on **reverse** le résultat dans x.



On peut changer le contenu




```
mesure = 12.1
# même étiquette, autre contenu
mesure = 12.4
```

Réaffecter, c'est **vider le verre et y verser autre chose**. L'ancienne valeur est jetée : il n'en reste rien.

Retenez ce détail : il explique tout le défi de la fin.

Bien nommer ses contenants

Oui : un nom qui dit ce qu'il y a dedans

- `masseEchantillon`
- `volumeEchantillon`
- `temperatureCelsius`
- `vitesseInitiale`

Non : vague ou abrégé

- `m, v, t`
- `x1, truc, machin`
- `masseEch`
- `tmp, val`

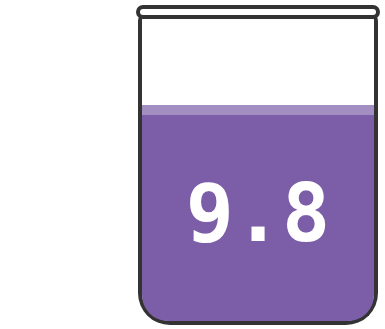
Un bon nom **dit ce qu'il y a dedans**, comme l'étiquette sur le bol. On écrit en **mixedCase** (premier mot en minuscules, majuscule à chaque mot suivant) et on évite les abréviations : le code se lit comme une phrase.

Exception assumée : x et y dans les exemples courts de ce diaporama, pour garder les verres lisibles.

Chaque valeur a un type

Type	Description	Exemples
`int`	nombre entier	`3, -7, 1024`
`float`	nombre à virgule	`9.8, -0.5, 3.14`
`str`	chaîne de caractères (texte)	`'eau', "H2O"`
`bool`	booléen (vrai ou faux)	`True, False`

Le type, c'est la **nature de l'ingrédient** dans le verre : de l'eau, du sable, du jus. La fonction `type( )` révèle le type d'une valeur.




```
type(9.8) → float
```

La mémoire, pas à pas

```
masse = 12.5
volume = 5.0
masse = 13.0
```

Chaque rangée est une photo de la mémoire après l'instruction. Le verre entouré d'orange vient de changer.

*La couleur suit la **valeur**, pas le verre : le verre garde son étiquette, le liquide change.*

#	masse	volume	INSTRUCTION	
1			`masse = 12.5`	masse est créée
2			`volume = 5.0`	volume est créée
3			`masse = 13.0`	masse change, 12.5 est jeté

Le défi de l'échange

X = 3, Y = 5 … ET SI ON LES ÉCHANGEAIT ?

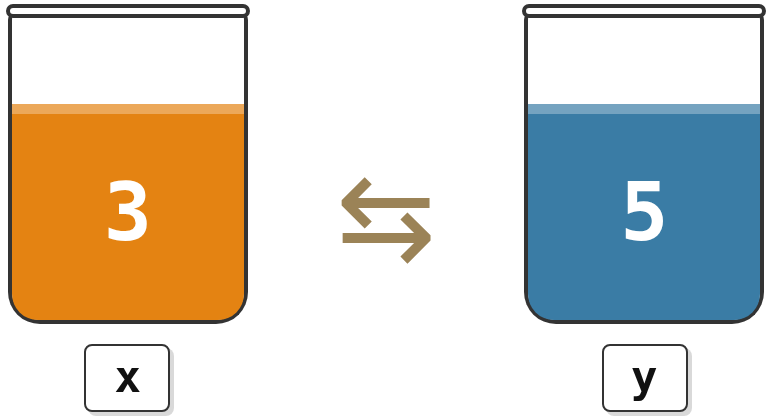
Le défi : échanger deux verres

```
x = 3
y = 5
```

Mission : écrire les instructions pour qu'à la fin, **x contienne 5** et **y contienne 3**.

- une instruction = un versement : un seul = par ligne
- interdit d'écrire les chiffres 3 et 5 : dans un vrai programme, ce sont des mesures qu'on ne connaît pas d'avance

Indice : que devient l'ancien contenu quand on verse dans un verre déjà plein ?



Objectif : x = 5 et y = 3

Tentative 1 : x = y, puis y = x

```
x = 3
y = 5
x = y
y = x
```

Raté. Dès la première ligne, on vide x pour y verser une copie de y : **le 3 est jeté**. À la fin, les deux verres contiennent 5.

#	x	y	INSTRUCTION
0			*état de départ*
1			`x = y` x reçoit une copie de y : le 3 est jeté
2			`y = x` y reçoit une copie de x... qui vaut déjà 5

Tentative 2 : l'inverse, y = x puis x = y

```
x = 3
y = 5
y = x
x = y
```

Raté encore. Cette fois c'est **le 5 qui est jeté** : les deux verres contiennent 3.

Peu importe l'ordre : le premier versement **détruit** une des deux valeurs. Il nous manque un endroit où la mettre en sécurité.

#	x	y	INSTRUCTION
0			*état de départ*
1			`y = x` **y reçoit une copie de x : le 5 est jeté**
2			`x = y` **x reçoit une copie de y... qui vaut déjà 3**

La solution : un troisième verre

```
x = 3
y = 5
temporaire = x
x = y
y = temporaire
```

Réussi ! x contient 5, y contient 3. Le troisième verre a gardé le 3 **en sécurité** pendant qu'on vidait x.

#	x	y	temporaire	INSTRUCTION
0				*état de départ*
1				`temporaire = x` le 3 est mis en sécurité
2				`x = y` x reçoit 5, sans regret
3				`y = temporaire` y récupère le 3 sauvegardé

L'algorithme d'échange

```
temporaire = x      # 1. sauvegarder le contenu de x
x = y               # 2. écraser x avec une copie de y
y = temporaire      # 3. restaurer la sauvegarde dans y
```

Comme deux verres de jus : pour échanger leur contenu, il faut un **troisième verre vide**.

- Le nom **temporaire** dit sa mission : il ne sert que le temps de l'échange.
- Trois versements, toujours dans cet ordre : sauvegarder, écraser, restaurer.
- Cet algorithme existe dans **tous les langages** (C, Java, JavaScript...) : on le retrouvera dans les tris.

Bonus, en Python seulement :

```
x, y = y, x
```

Python prépare les deux valeurs de droite avant de verser : le troisième verre est caché à l'intérieur.

À toi de jouer

Dans Google Colab (ou le compilateur en ligne de programiz), teste les trois versions et observe ce que `print` affiche :

```
x = 3
y = 5
temporaire = x
x = y
y = temporaire
print(x, y)
```

- Remplace les trois lignes du milieu par `x = y` puis `y = x` : qu'affiche print ?
- Puis par l'inverse. Et enfin par `x, y = y, x`.
- Demande à l'usager les deux valeurs avec `input()` au lieu d'écrire 3 et 5.

Pour rejouer chaque tentative verre par verre, avec ton propre code :

code.experimentations.xyz/visualisateur/variables/visualisateur-pas-a-pas.html