Variables et constantes

Table des matières

Introduction

Tous les langages de programmation utilisent la notion de variable. Une variable permet de stocker des valeurs d'un certain type. On peut affecter des valeurs à des variables et effectuer des opérations sur ces valeurs. Par exemple on peut additionner deux variables et stocker le résultat dans une troisième.

Ce module a pour objectif de présenter le concept de variable et de découvrir les principales opérations que l'on fait avec des variables.

Nous étudierons l'affectation, les opérations sur les nombres et les chaînes de caractères, ainsi que la notion de type. Nous définirons également les constantes, des variables dont la valeur n'est jamais modifiée.

I Qu'est ce qu'une variable ?

Objectifs

- Savoir ce qu'est une variable ;
- Connaître les bonnes pratiques de nommage.

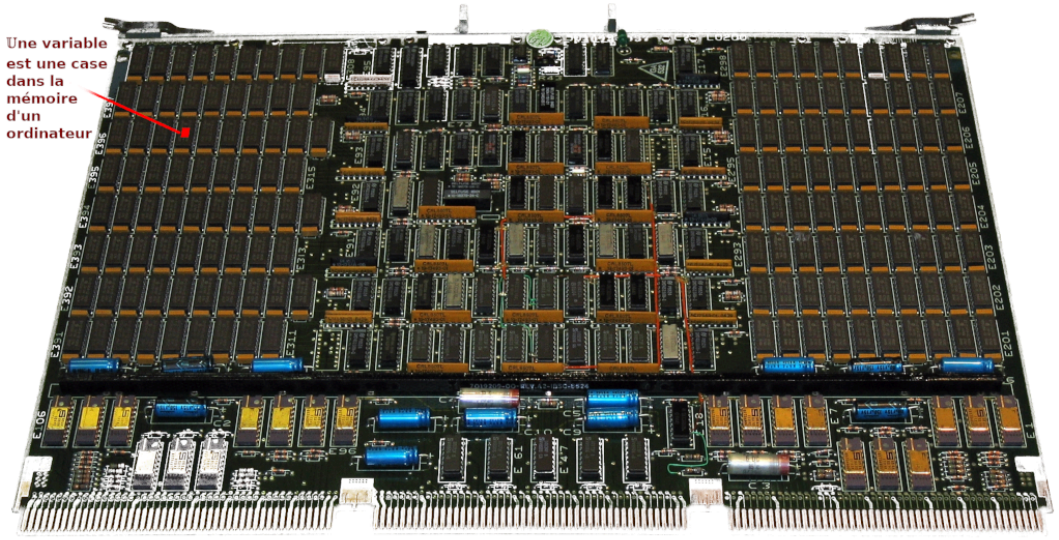
Mise en situation

Dans les programmes, les variables servent à stocker les données pour réaliser différentes opérations dessus. C'est un outil commun à tous les langages qui permet de faire l'interface entre le programme et la mémoire de l'ordinateur. Chaque variable a un nom ainsi qu'une valeur. Le nom d'une variable est défini par le développeur dans le code de son programme, mais de nombreuses règles existent pour contraindre les noms de variable à un format spécifique à chaque langage. La valeur est dynamique et peut évoluer en fonction de l'exécution du programme, d'où le terme de variable.

Az Définition

Un ordinateur est doté d'une mémoire, et cette mémoire est composée de nombreuses cases. Les programmes peuvent lire et écrire dans ces cases, grâce à des variables.

Une variable est une case mémoire avec un nom. On peut créer autant de variables que l'on veut dans un programme, pour mémoriser et retrouver des informations.



Une variable est une case dans la mémoire d'un ordinateur

Exemple

```
1 /** JavaScript : affiche un Hello World */
2 let aString = 'Hello World'
3 let aNumber = 1
4 console.log(aString)
5 console.log(aNumber)
```

```python
"""Python : affiche un Hello World."""
a_string = "Hello World"
a_number = 1
print (a_string)
print (a_number)
```

Déclarer une variable avec let (JavaScript)

Syntaxe

En JavaScript la première fois qu'une variable est utilisée, on la déclare avec le mot clé `let` :

`let aString = 'Hello World'.`

Nommer une variable : règles syntaxiques

Attention

Il existe des règles pour donner des noms aux variables :

- Un nom de variable ne peut pas commencer par un chiffre.
- Un nom de variable ne contient pas d'espace.

Nommer une variable : conventions

Méthode

Il existe aussi des conventions, qui permettent de rendre un programme plus lisible et plus compréhensible.

- Choisir des noms clairs et précis (en lisant le nom d'une variable, il doit être possible de savoir à quoi elle sert dans le programme).
- Écrire le nom des variables en anglais.
- Ne pas utiliser de caractères spéciaux (on se restreint aux 26 lettres en minuscules et majuscules, aux 10 chiffres et à l'*underscore* _).
- Pour être explicite le nom d'une variable peut contenir plusieurs mots, pour cela on peut :
  - remplacer les espaces par des *underscores* (méthode *snake_case* préférée en Python).
  - ou commencer chaque nouveau mot par une lettre majuscule (méthode *camelCase* préférée en JavaScript).

camelCase (variables en JavaScript)

Méthode

`multipleWordsVariable`

snake_case (variables en Python)

Méthode

`multiple_words_variable`

La différence entre majuscules et minuscules

Attention

Le langage Python, le JavaScript et la majorité des autres langages différencient les majuscules et les minuscules (ils respectent la casse).

Ainsi, `totalprice` et `totalPrice` correspondent à deux variables différentes.

À retenir

Syntaxe

```js
/** JavaScript */
let variableName = value
```

```python
"""Python."""
variable_name = value
```

Pourquoi let ?

Complément

Let signifie *« soit »* en anglais, au sens mathématique de *« soit i un entier compris entre... »*.

`let name = 'Alice'` peut donc être traduit par *« prenons une variable name qui a pour valeur initiale Alice... »*

Pourquoi pas var ?

Complément

En fait, `var` existe aussi et sert aussi à déclarer des variables. Le terme est antérieur à `let` et sert à déclarer des variables globales. Mais comme en programmation on n'aime pas beaucoup les variables globales, en 2015 lors d'une évolution de JavaScript `let` a été introduit pour déclarer des variables locales (ça on aime bien).

Depuis, il est globalement conseillé de ne plus utiliser `var` et d'utiliser `let` à la place.

- *Portée des variables* *(cf. p.36)*
- *Variables globales* *(cf. p.37)*

II Exercice

Question 1

[solution n°1 p. 28]

Écrire un programme qui déclare une variable qui a pour nom *superVariable* avec la valeur 42 et qui affiche le contenu de *superVariable*.

Indice :

En JavaScript, on déclare une variable avec le mot-clé `let`.

Question 2

[solution n°2 p. 28]

Créer trois variables initialisées à zéro qui doivent respectivement servir à calculer un nombre de pommes, une quantité de farine en kilogramme et un nombre de gâteaux vendus.

Indice :

Le nommage des variables doit permettre de comprendre le sens de ce qu'elles stockent.

III Affectation

Objectifs

- Savoir déclarer et affecter une variable ;
- Savoir affecter la valeur d'une variable à une autre variable.

Mise en situation

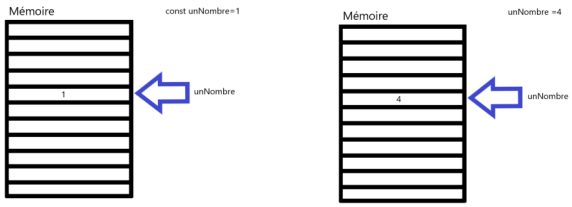
La syntaxe qui permet d'affecter une valeur à une variable dépend du langage de programmation.

Az Définition

On appelle affectation le fait de donner une certaine valeur à une variable. **Affecter une valeur à une variable** signifie écrire cette valeur dans la case mémoire représentée par la variable.

Pour réaliser une affectation en JavaScript ou en Python on utilise le signe =.

Exemple



Affectation d'une variable en mémoire

JavaScript

Exemple

```javascript
/** JavaScript : déclarations, affectations et affichages */
let x = 1
console.log(x)
x = 2
console.log(x)
let y = x
console.log(y)

// Affiche 1, 2 et 2
```

Python

Exemple

```python
"""Python : affectations et affichages."""
x = 1
print(x)
x = 2
print(x)
y = x
print(y)

// Affiche 1, 2 et 2
```

Remarque

Il est possible d'affecter la valeur stockée dans une variable à une autre variable.

À retenir

Syntaxe

```javascript
/** JavaScript */
let variableName = value
console.log(variableName)
```

```python
"""Python."""
variable_name = value
print (nom_variable)
```

IV Exercice

Question

[solution n°3 p. 28]

Écrire un programme JavaScript qui contient une variable qui a le nom *superVariable* avec la valeur 42 et qui affiche cette variable. Ensuite, affecter la nouvelle valeur 43 à *superVariable* et l'afficher à nouveau.

V Type

Objectif

- Comprendre l'utilité des types.

Mise en situation

L'affectation d'une valeur à une variable permet de stocker la valeur en question en mémoire, durant l'exécution du programme. Cependant l'espace mémoire nécessaire pour le stockage va fortement dépendre de cette valeur. En effet, stocker le chiffre 4 demandera beaucoup moins d'espace que de stocker une phrase complète. C'est pour cela que de nombreux langages associent un type à une variable, qui doit être définie lors de sa déclaration. De plus, le type d'une variable permet de définir la manière d'effectuer des opérations dessus. En effet il est possible de réaliser des opérations mathématiques sur des nombres, mais pas sur du texte.

Az Définition

Une variable est associée à un type, qui fixe la taille de la case mémoire et les opérations que l'on peut faire sur cette variable.

Méthode

Pour afficher le type d'une variable appelée `mot` :

- En JavaScript : `console.log(typeof mot)`
- En Python : `print(type(mot))`

Types courants en JavaScript

Az Définition

- Les nombres : `number`
- Les chaînes de caractères : `string`
- Les booléens (une variable qui peut uniquement être vraie (`true`) ou fausse (`false`)) : `boolean`

Types courants en Python

Az Définition

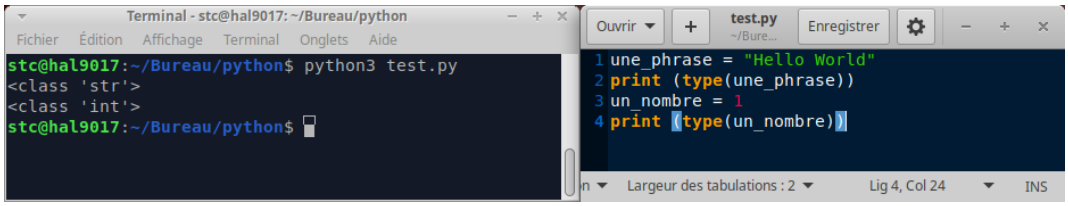
- Les nombres entiers : `int`
- Les nombres décimaux : `float`
- Les chaînes de caractères : `str`

Exemple

```javascript
/** JavaScript : affiche le type de variables */
let aString = 'Hello World'
console.log(typeof aString)
let aNumber = 1
console.log(typeof aNumber)

```

```
string
number
```




```python
"""Python : affiche le type de variables."""
a_string = "Hello World"
print(type(a_string))
a_number = 1
print(type(a_number))

```

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

À retenir

Syntaxe

```python
"""Python."""
type(variable_name)
```

```javascript
/** JavaScript */
typeof variableName
```

VI Exercice

Question

[solution n°4 p. 28]

En JavaScript :

- Créer une variable *superVariable* et lui affecter la valeur 42.
- Créer une autre variable *superVariable2* et lui affecter la valeur 42.1.
- Créer une autre variable *superVariable3* et lui affecter la valeur '42.1'.

Afficher le type de ces trois variables.

VII Qu'est ce qu'une constante ?

Objectifs

- Savoir quand utiliser une constante ;
- Connaître la syntaxe de déclaration de constante en JavaScript.

Mise en situation

Dans un programme, il arrive que certaines données ne changent pas. Ce sont des constantes dont la déclaration et l'initialisation est un peu différente de celles des variables.

Constance

Az Définition

Dans un programme, certaines valeurs sont fixées à l'avance et ne changent pas.

Le calcul du périmètre d'un cercle fait intervenir son diamètre (qui dépend du cercle) et le nombre Pi (qui ne change pas).

Pour s'assurer que dans le programme la valeur qu'on aura associée à Pi ne change pas, on déclare que Pi est une constante.

Constantes en JavaScript

Syntaxe

Pour indiquer qu'une variable est constante en JavaScript, on utilise le mot-clé `const` au lieu du mot-clé `let`. Tenter de modifier une variable déclarée `const` provoquera une erreur.

```
1 // diameter pourra être modifiée, pour un autre cercle, mais pi est une
  constante
2 let diameter = 10
3 const pi = 3.14159
```

Initialisation

Attention

Il est possible de séparer la déclaration d'une variable et son initialisation.

En revanche il est impossible de séparer déclaration et initialisation d'une constante : elle doit **toujours être initialisée lors de sa déclaration**.

```
1 // Ce code ne fonctionne pas : la constante n'est pas
2 // initialisée lors de sa déclaration.
3 const pi
4 pi = 3.14
```

Constantes en Python

Complément

Il n'est pas possible de définir des constantes en Python, il faut donc faire attention à ne jamais modifier une variable dont la valeur est sensée être fixe.

Exercice

[solution n°5 p. 29]

Exercice

Comment créer la constante Phi en JavaScript ?

A `let phi = 1.618`

B `const phi  phi = 1.618`

C `const phi = 1.618`

Exercice

Il est possible de déclarer des constantes en Python.

A Vrai

B Faux

Exercice

Quel programme calcule et affiche **correctement** l'aire d'un disque de rayon 10 cm ?

A

```
let ray = const pi = let area = ray * console.log(area + "
10      3.14159      ray * pi          cm²")
```

B

```
let ray = const pi      = let area = ray * console.log(area  +
10      pi    3.14159   ray * pi          " cm²")
```

C

```
let   const   pi   = let area = ray * ray console.log(area  +  "
ray   3.14159       * pi                  cm²")
```

IX Opérations sur les nombres

Objectif

- Savoir faire des opérations simples sur des variables de type numérique.

Mise en situation

Il est possible d'effectuer des opérations mathématiques sur les variables.

Les variables de type nombres entiers ou décimaux peuvent être manipulées avec les fonctions mathématiques classiques, comme :

- La somme : +
- La soustraction : -
- La multiplication : *
- La division : /

Exemple

```javascript
/** JavaScript : effectue la somme 42 + 1 */
const x = 42
const y = 1
const z = x + y
console.log(z)

```

```python
"""Python : effectue la somme 42 + 1."""
x = 42
y = 1
z = x + y
print (z)

```

Les deux programmes affichent 43.

À retenir

Syntaxe

```javascript
/** JavaScript */
const x = 1 + 2
const y = x - 1
let z = x / y
z = z * 2
```

```python
"""Python."""
x = 1 + 2
y = x - 1
z = x / y
z = z * 2
```

X Exercice

Question 1

[solution n°6 p. 30]

Écrire un programme JavaScript qui affiche le résultat du calcul 42 * 43.

Question 2

[solution n°7 p. 30]

En JavaScript :

- Créer les variables *x* et *y* de valeurs 42 et 43.
- Affecter la valeur *x* multipliée par *y* à une variable *z*.
- Afficher *z*.

Indice :

Les variables ne sont pas ré-affectées, penser à utiliser le mot-clé `const`.

Question 3

[solution n°8 p. 30]

En JavaScript :

- Créer la variable x avec la valeur initiale 42.
- Ajouter 1 à *x* et stocker le résultat dans *x* (on utilise ici une seule variable).
- Afficher *x*.

XI Opération sur les chaînes de caractères

Objectif

- Savoir faire des opérations simples sur des variables de type chaîne de caractères.

Mise en situation

Il existe de nombreuses opérations pour manipuler les chaînes de caractères, par exemple :

- Connaître le nombre de caractères de cette chaîne.
- Retourner le caractère qui est à la position *x*.
- Retourner une sous-chaîne de caractère qui commence à la position *x* et se termine à la position *y*.

JavaScript Syntaxe

- `myString.length` permet de connaître le nombre de caractères de `myString`,
- `myString[x]` permet de retourner le caractère qui est à la position *x*,
- `myString.substring(x, y)` permet de retourner une sous-chaîne de caractères qui commence à la position *x* et se termine à la position *y*.

Python

- `len(my_string)` permet de connaître le nombre de caractères de `my_string`,
- `my_string[x]` permet de retourner le caractère qui est à la position *x*,
- `my_string[x:y]` permet de retourner une sous-chaîne de caractères qui commence à la position *x* et se termine à la position *y*.

Attention

En Python, en JavaScript, comme dans la plupart des langages informatiques, on commence à compter à partir de 0 et non à partir de 1. Le premier caractère d'une chaîne est donc le caractère numéro 0.

Exemple

```
1 /** JavaScript : manipule la chaîne Hello World */
2 let s = 'Hello World'
3 console.log(s.length)
4 console.log(s[0])
5 console.log(s.substring(6, 11))
6
```

```
"""Python : manipule la chaine Hello World."""
s = "Hello World"
print(len(s))
print(s[0])
print(s[6:11])

```

Les deux programmes affichent :

```
11
H
World
```

XII Exercice

Question

[solution n°9 p. 30]

En JavaScript, créer la chaîne correspondant au vers « A stone Troll sat on his seat of stone », puis :

- Compter le nombre de caractères.
- Afficher le 9^{e} caractère (les espaces sont des caractères comme les autres).
- Et afficher la sous-chaîne correspondant à Troll.

Indice :

La numérotation des caractères commence à 0. Ainsi, le premier A est à l'indice 0 et non 1.

XIII Essentiel

XIV Quiz

Exercice 1 : Quiz - Méthode

[solution n°10 p. 30]

Exercice

Quels noms de variables sont autorisés ?

- **A** nbFruitsVendus
- **B** total_pommes_restantes
- **C** somme Pasteques
- **D** 1er_choix

Exercice

Quels noms de variables sont en *camel case*, la syntaxe conseillée en JavaScript ?

- **A** nbFruitsSold
- **B** total_bananas
- **C** SumPeaches
- **D** firstChoice

Exercice

Quelle opération peut réaliser le programme suivant ?

```
1 const a = 9.81
2 const b = 74.56
3 const c = a * b
4 console.log(c)
5
```

- **A** Il affiche le résultat de la multiplication de 9,81 par 74,56.
- **B** Il affiche le nombre moyen de grenouilles vertes qui a été multiplié par 9,81 dans les étangs d'une commune.
- **C** Il calcule le poids d'un objet dont la masse est de 74,56 kg.

Exercice 5 : Quiz - Code

[solution n°11 p. 31]

Exercice

Qu'affiche le programme JavaScript suivant ?

```
1 const book = "Les Misérables"
2 console.log(book)
3
```

Exercice

Qu'affiche le programme suivant ?

```
1 const word = 'ananas'
2 console.log(typeof word)
3 console.log(word[0], word[3], word[5])
4 console.log(word.substring(4, word.length))
5
```

- **A** string a a a nas
- **B** string n n s nas
- **C** string a n s as
- **D** string a a s nas

Exercice

Qu'affiche le programme suivant ?

```
1 const firstName = 'Marie'
2 const name = 'Curie'
3 console.log(firstName, name)
```

- **A** Marie
- **B** Curie
- **C** Marie Curie
- **D** MarieCurie

Exercice

Qu'affiche le programme suivant ?

```
1 const firstName = 'Marie'
2 const name = 'Curie'
3 console.log(name + firstName)
```

- **A** Marie

B Curie

C Marie Curie

D CurieMarie

Exercice

Soit le programme suivant. Par quoi doit-on remplacer les « **...** » pour obtenir les initiales d'Alan Turing ?

```
1 const name = 'Alan Turing'
2 console.log(name[0] + name[...])
```

Exercice

Que fait le programme suivant ?

```
1 const firstVehicle = 'voiture'
2 const secondVehicle = 'avion'
3
4 if (firstVehicle.length > secondVehicle.length) {
5   console.log(secondVehicle, firstVehicle)
6 } else {
7   console.log(firstVehicle, secondVehicle)
8 }
9
```

A Le programme compare les mots selon l'ordre alphanumérique et les affiche triés.

B Le programme ne fait rien, on ne peut pas comparer des constantes.

C Le programme compare la longueur des mots et affiche le mot le plus court en premier et le plus long en deuxième.

Exercice

Quelle opération permet de calculer le nombre de paires que je peux faire avec 51 chaussettes ?

A const numberOfPairs = 51 / 2

B const numberOfPairs = 51 - 1/2

C numberOfPairs = Math.floor(51 / 2)

XV Exercice : Défi

Alice et Bob font leurs courses au supermarché. Chaque fois qu'ils ajoutent un article dans le caddie, ils calculent le prix total des articles présents dans le caddie.

Liste des articles entrées au fur et à mesure :

- 3 pastèques à 3 € pièce,
- 2 kilos de sucre à 1 € par kilo,
- 4 litres d'huile de noix à 2 € le litre,
- 1 tarte aux pommes à 5 €,
- 2 paquets de riz à 2 € le paquet,

Question

[solution n°12 p. 34]

Écrire un programme qui commence par créer la variable total en l'initialisant à la valeur 0.

Puis, pour chaque article de la liste le programme doit :

- affecter le nom de l'article à une variable `article`,
- affecter le prix de l'article à une variable `price`,
- affecter la quantité à une variable `quantity`,
- calculer le nouveau total,
- afficher les trois premières lettres de l'article, puis le prix unitaire de l'article, puis la quantité, puis le prix total cumulé dans le caddie depuis le début des achats.

Indice :

On utilisera les variables :

- `total`,
- `article`,
- `price`,
- `quantity`.

Conclusion

Les variables sont essentielles au fonctionnement d'un programme. Sans elles, il ne serait pas possible de traiter des données ou de stocker des informations temporairement. Ces variables sont créées par le développeur dans un programme, et des valeurs sont ensuite affectées, en fonction de son déroulement. Les variables ont systématiquement un type, qui permet de définir à quoi ressemble la valeur qui sera stockée : un nombre, une chaîne de caractères, etc. De plus ce type permet de définir les opérations qu'il est possible de réaliser sur la variable, chaque type ayant des opérations qui lui sont propres.

Solutions des exercices

Solution n°1

[exercice p. 7]

```javascript
/** JavaScript : affecte et affiche superVariable */
let superVariable = 42
console.log(superVariable)
```

```python
"""Python : affecte et affiche superVariable."""
superVariable = 42
print (superVariable)
```

Solution n°2

[exercice p. 7]

```javascript
/** JavaScript : création des trois variables */
let apples = 0
let flour = 0
let cakes = 0
```

```python
"""Python : création des trois variables."""
apples = 0
flour = 0
cakes = 0
```

> 💬 Remarque
>
> On aurait aussi pu choisir un nommage du type `appleNumber`, `flourQuantity` et `cakeSold`. Le choix dépend de l'utilisation des variables dans le programme : est-ce-qu'il est toujours explicite que la variable `apples` référence un **nombre** de pomme et pas un **nom** de pommes ?

Solution n°3

[exercice p. 10]

```javascript
let superVariable = 42
console.log(superVariable)
superVariable = 43
console.log(superVariable)
```

Solution n°4

[exercice p. 13]

```javascript
let superVariable = 42
let superVariable2 = 42.1
let superVariable3 = '42.1'
console.log(typeof superVariable)
console.log(typeof superVariable2)
console.log(typeof superVariable3)

```

```
number
number
string
```

> **Remarque**
>
> Le langage Python différencie les nombres entiers des nombres décimaux alors que le langage JavaScript associe le même type à tous les nombres (le type `number`).

Solution n°5

[exercice p. 16]

Exercice

Comment créer la constante Phi en JavaScript ?

A `let phi = 1.618`

B `const phi  phi = 1.618`

C `const phi = 1.618`

Exercice

Il est possible de déclarer des constantes en Python.

A Vrai

B Faux

Python n'a pas de mécanisme pour forcer une variable à être constante. Une convention entre développeurs est d'indiquer le nom de la pseudo-constante en majuscules, mais rien n'empêchera techniquement sa modification.

Exercice

Quel programme calcule et affiche **correctement** l'aire d'un disque de rayon 10 cm ?

A

```
let ray = 10
```

```
const pi = 3.14159
```

```
let area = ray * ray * pi
```

```
console.log(area + " cm²")
```

Ici, `ray` et `area` pourront être modifié pour effectuer d'autres calculs. En revanche, pi ne changera jamais : sa valeur est fixe.

B

```
let ray = 10
```

```
const pi = 3.14159
```

```
let area = ray * ray * pi
```

```
console.log(area + " cm²")
```

Il faut initialiser une constante lors de sa déclaration.

C

```
let const  pi let   area   = console.log(area Il y a une erreur de syntaxe,
ray =         ray  *  ray  * + " cm²")       ray n'est associé à aucune
    3.14159   pi                             valeur.
```

Solution n°6

[exercice p. 18]

```
1 console.log(42 * 43)
2
```

Solution n°7

[exercice p. 18]

```
1 const x = 42
2 const y = 43
3 const z = x * y
4 console.log(z)
5
```

Solution n°8

[exercice p. 18]

```
1 let x = 42
2 x = x + 1
3 console.log(x)
4
```

Solution n°9

[exercice p. 21]

```
1 const verse = 'A stone Troll sat on his seat of stone'
2 console.log(verse.length)
3 console.log(verse[8])
4 console.log(verse.substring(8, 13))
```

Solution n°10

[exercice p. 23]

Exercice

Quels noms de variables sont autorisés ?

A nbFruitsVendus

B total_pommes_restantes

C somme Pasteques Un nom de variable ne peut pas contenir d'espaces.

D 1er_choix Un nom de variable ne peut pas commencer par un chiffre.

Exercice

Quels noms de variables sont en *camel case*, la syntaxe conseillée en JavaScript ?

- **A** nbFruitsSold
- **B** total_bananas
- **C** SumPeaches
- **D** firstChoice

La convention *camelCase* sépare les mots par une majuscule, mais le premier mot est **toujours** en minuscules.

Exercice

Quelle opération peut réaliser le programme suivant ?

```
1 const a = 9.81
2 const b = 74.56
3 const c = a * b
4 console.log(c)
5
```

- **A** Il affiche le résultat de la multiplication de 9,81 par 74,56.
- **B** Il affiche le nombre moyen de grenouilles vertes qui a été multiplié par 9,81 dans les étangs d'une commune.
- **C** Il calcule le poids d'un objet dont la masse est de 74,56 kg.

Ce programme est sensé calculer le poids d'un objet dont la masse est de 74,56 kg mais les noms des variables (***a*** pour pour la constante de gravitation, ***b*** pour la masse, ***c*** pour le poids) ne permettent pas de le comprendre facilement.

Par conséquent ce programme pourrait tout aussi bien servir à afficher le résultat de la multiplication de 9,8 par 74,56 ou à calculer et à afficher le nombre de grenouilles vertes.

C'est pour cela qu'il est très important de nommer de façon claire et précise toutes les variables et constantes d'un programme.

Solution n°11

[exercice p. 24]

Exercice

Qu'affiche le programme JavaScript suivant ?

```
1 const book = "Les Misérables"
2 console.log(book)
3
```

Les Misérables

Exercice

Qu'affiche le programme suivant ?

```
1 const word = 'ananas'
2 console.log(typeof word)
3 console.log(word[0], word[3], word[5])
4 console.log(word.substring(4, word.length))
5
```

A string a a a nas

B string n n s nas

C ☐

```
string a as
       n
       s
```

La variable est de type `string`, ses premier, quatrième et sixième caractères (indices 0, 3 et 5) sont `a`, `n` et `s`. Aussi, la chaîne partant du cinquième caractère (indice 4) et allant jusqu'à la fin du mot est `as`.

D string a a s nas

Exercice

Qu'affiche le programme suivant ?

```
1 const firstName = 'Marie'
2 const name = 'Curie'
3 console.log(firstName, name)
```

A Marie

B Curie

C Marie Curie

D MarieCurie

La fonction `console.log` ajoute automatiquement une espace entre les différentes valeurs qu'on lui demande d'afficher.

Exercice

Qu'affiche le programme suivant ?

```
1 const firstName = 'Marie'
2 const name = 'Curie'
3 console.log(name + firstName)
```

A Marie

B Curie

C Marie Curie

D CurieMarie

L'opérateur « **+** » permet de concaténer (coller) des chaînes de caractères qui sont donc affichées les unes à la suite des autres, sans espace.

Pour séparer les chaînes et rendre le texte plus lisible, on peut ajouter une espace.

```
1 const firstName = 'Marie'
2 const name = 'Curie'
3 console.log(name + ' ' + firstName)
```

Exercice

Soit le programme suivant. Par quoi doit-on remplacer les « **...** » pour obtenir les initiales d'Alan Turing ?

```
1 const name = 'Alan Turing'
2 console.log(name[0] + name[...])
```

5

La lettre T est la sixième lettre de la variable `name`. Comme les indices **commencent à 0**, il faut utiliser l'indice 6 - 1 = 5.

```
1 const name = 'Alan Turing'
2 console.log(name[0] + name[5])
```

Exercice

Que fait le programme suivant ?

```
1 const firstVehicle = 'voiture'
2 const secondVehicle = 'avion'
3
4 if (firstVehicle.length > secondVehicle.length) {
5   console.log(secondVehicle, firstVehicle)
6 } else {
7   console.log(firstVehicle, secondVehicle)
8 }
9
```

A Le programme compare les mots selon l'ordre alphanumérique et les affiche triés.

B Le programme ne fait rien, on ne peut pas comparer des constantes.

C Le programme compare la longueur des mots et affiche le mot le plus court en premier et le plus long en deuxième.

Exercice

Quelle opération permet de calculer le nombre de paires que je peux faire avec 51 chaussettes ?

A const numberOfPairs = 51 / 2

B const numberOfPairs = 51 - 1/2

C numberOfPairs = Math.floor(51 / 2)

L'opération `51 / 2` renvoie un nombre décimal : `25.5`. Or le nombre de paires doit être un nombre entier.

Il faut donc retirer la partie décimale, ce que permet de faire la fonction `Math.floor()`. *floor* signifie sol ou plancher en anglais.

Le programme suivant permet de calculer et d'afficher le nombre de paires faisables.

```
1 const numberOfSocks = 51
2 const division = numberOfSocks / 2
3 const numberOfPairs = Math.floor(division)
4 console.log(numberOfPairs)
```

Solution n°12

[exercice p. 26]

```
1  let total = 0
2  let article = 'Pastèque'
3  let price = 3
4  let quantity = 3
5  total = total + price * quantity
6  console.log(article.substring(0, 3), price, quantity, total)
7
8  article = 'Sucre'
9  price = 1
10 quantity = 2
11 total = total + price * quantity
12 console.log(article.substring(0, 3), price, quantity, total)
13
14 article = 'Huile de noix'
15 price = 4
16 quantity = 2
17 total = total + price * quantity
18 console.log(article.substring(0, 3), price, quantity, total)
19
20 article = 'Tarte aux pommes'
21 price = 5
22 quantity = 1
23 total = total + price * quantity
24 console.log(article.substring(0, 3), price, quantity, total)
25
26 article = 'Riz'
27 price = 2
28 quantity = 2
29 total = total + price * quantity
30 console.log(article.substring(0, 3), price, quantity, total)
31
```

Crédits des ressources

Une variable est une case dans la mémoire d'un ordinateur p. 4

Attribution - Partage dans les Mêmes Conditions - Photo originale de Poil, "4 Mb VAX 8600 memory board", https://commons.wikimedia.org[1] *(modifée par Stéphane Crozat)*

1. https://commons.wikimedia.org/wiki/File:4mbramvax.jpg

Contenus annexes

1. Portée des variables

Objectif

- Comprendre la notion de portée locale et de bloc de définition.

Mise en situation

Un programme est séparé en blocs afin d'être plus lisible. Par exemple si on veut écrire un programme qui affiche une suite de nombres puis qui calcule leur somme, on peut séparer le programme en deux parties : deux blocs, qui ont chacun leurs variables. En effet chaque variable ne sera utilisable que dans le bloc de code où elle a été définie. On appelle portée d'une variable la zone de code dans laquelle une variable sera définie et utilisable.

Portée d'une variable Az Définition

Lorsqu'on définit une variable elle est associée au bloc où elle se trouve et n'est visible que dans celui-ci. On parle de **portée locale**.

Cela signifie qu'il est **impossible** d'afficher ou d'utiliser cette variable dans un autre bloc : elle n'existe tout simplement pas.

Si une variable est déclarée dans le bloc principal (c'est à dire en dehors de toute boucle ou fonction) elle est visible dans toute la partie du programme située après sa déclaration.

Exemple

Dans le programme suivant la variable `numberApple` est visible dans tout le programme y compris dans le bloc `if`.

En revanche la variable `enoughApple` est déclarée dans le bloc `if`, donc visible uniquement dans celui-ci.

La dernière instruction génère donc une erreur.

```
1 /** JavaScript */
2 const numberApple = 30
3 console.log(numberApple)
4 if(numberApple === 30) {
5   let enoughApple = true
6   console.log(enoughApple)
7 }
8 console.log(enoughApple)
9
```

Remarque

La portée des variables fonctionne de la même manière avec d'autres boucles ou dans des fonctions.

Une variable définie dans une fonction n'est visible que dans celle-ci.

À retenir

En fonction du bloc où sont déclarées les variables, elles n'ont pas la même portée et ne sont donc pas visibles et utilisables dans la même partie du programme.

2. Variables globales

Objectifs

- Savoir ce que sont et à quoi servent les variables globales ;
- Adopter les bonnes pratiques.

Mise en situation

Les variables ont généralement une portée locale, c'est à dire qu'elles ne sont accessibles que dans le bloc de code où elles ont été définies. C'est la plupart du temps largement suffisant, mais il peut arriver d'avoir besoin d'une variable soit accessible dans la totalité du programme. Par exemple une variable qui contiendrait une information de configuration régulièrement utilisé à des endroits très différents du code. Pour cela, il est possible d'utiliser une variable de portée globale, qui ne sera donc pas limitée de la même manière qu'une variable locale.

Variable globale en JavaScript

Az Définition

En JavaScript, on définit habituellement une variable avec `let nameV = value`.

En remplaçant le mot-clé `let` par le mot-clé `var` on change la portée de la variable.

La variable aura une portée globale et sera donc visible dans toute la partie du programme située après sa définition.

En Python le mot-clé `global`, suivi du nom de la variable indique que la variable à utiliser a été préalablement définie dans un autre bloc du programme et que c'est cette variable qu'il faut modifier.

Exemple

```
1 /** JavaScript : programme erroné */
2 const numberApple = 30
3 console.log(numberApple)
4 if(numberApple === 30)
5 {
6   let enoughApple = true
7   console.log(enoughApple)
8 }
9 console.log(enoughApple)
10
```

L'exécution de ce programme renvoie une erreur car la dernière instruction fait référence à une variable qui n'existe pas : `enoughApple` est locale au bloc `if`.

```javascript
/** JavaScript  : programme corrigé */
const numberApple = 30
console.log(numberApple)
if(numberApple === 30)
{
  var enoughApple = true
  console.log(enoughApple)
}
console.log(enoughApple)

```

Si on remplace `let` par `var` dans la définition de `enoughApple`, il n'y a plus d'erreur car la variable est visible hors de la boucle `if`.

Ce code fonctionne, mais c'est une **mauvaise pratique** !

À consommer avec modération

⚠ Attention

Les variables globales sont source de nombreux problèmes.

En effet, une variable déclarée globale pourra être modifiée dans n'importe quelle autre partie du programme, ce qui peut provoquer des erreurs difficiles à détecter, comme l'écrasement inattendu de la valeur d'origine (on appelle cela un effet de bord).

Il est recommandé de ne jamais utiliser de variables globales par défaut, et de les utiliser avec parcimonie uniquement dans des cas bien identifiés.

⊕ Complément

Si on veut inverser la valeur de deux variables avec une fonction, on peut utiliser des variables globales (mais ce n'est pas une bonne façon de faire).

```python
"""Python."""
fruit_1 = ' pear '
fruit_2 = ' pineapple '

def inversion():
    global fruit_1
    global fruit_2
    temp = fruit_2
    fruit_2 = fruit_1
    fruit_1 = temp

print(fruit_1 + fruit_2)
inversion()
print(fruit_1 + fruit_2)
```

Le programme affiche « pear pineapple » puis « pineapple pear ».

À retenir

Il est possible de créer des variables dont la portée s'étend au delà de leur bloc de définition, avec le mot-clé `var` en JavaScript et le mot-clé `global` en Python, il s'agit des variables globales. Cependant il est recommandé de les utiliser avec prudence car cela peut provoquer des résultats inattendus.