ALGORITHMIQUE · CHAPITRE 4 · FONDATIONS

Déterminer l’ordre des opérations

Décomposez une expression : parenthèses, priorités, associativité et signe. Comparez les puissances en Python et MATLAB.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Décomposer une expression en résultats intermédiaires.
- Distinguer priorité et associativité.
- Rendre le regroupement explicite quand les langages diffèrent.

Prérequis. Connaître les opérateurs arithmétiques scalaires.

Pourquoi les mêmes nombres donnent-ils 14 ou 20 ?

Dans `2 + 3 * 4`, multiplier avant d’additionner produit 14. Dans `(2 + 3) * 4`, additionner avant de multiplier produit 20. Les nombres et les opérations sont les mêmes ; leur regroupement change le résultat.

Définir la priorité

La **priorité** décide quels opérateurs se regroupent d’abord. Les parenthèses imposent un regroupement explicite. Une expression imbriquée s’évalue en calculant les sous-expressions nécessaires.

Priorités arithmétiques usuelles, du plus fort au plus faible

Niveau	Python	MATLAB (scalaires)

Niveau	Python	MATLAB (scalaires)
Regroupement	(expression)	(expression)
Puissance	**	^
Signe unaire	+x, -x	+x, -x
Produits et divisions	*, /, //, %	*, /
Sommes et différences	+, -	+, -

Cette table couvre les exemples arithmétiques du cours. `mod` et `floor` sont des appels de fonctions MATLAB. Pour les puissances signées, explicitez aussi l'exposant : `2 ** (-2)` ou `2 ^ (-2)` vaut 0,25. `-2 ** 2` en Python et `-2 ^ 2` en MATLAB valent -4 ; `(-2) ** 2` et `(-2) ^ 2` valent 4.

Définir l'associativité

L'**associativité** fixe le regroupement d'opérateurs de même priorité. `20 / 5 * 2` se lit de gauche à droite : `(20 / 5) * 2 = 8`. Lire `20 / (5 * 2)` donnerait 2. De même, `10 - 3 - 2` vaut 5.

Attention aux puissances : Python regroupe `2 ** 3 ** 2` à droite, donc `2 ** (3 ** 2) = 512`. MATLAB regroupe `2 ^ 3 ^ 2` à gauche, donc `(2 ^ 3) ^ 2 = 64`. Le code ci-dessous impose 512 dans les deux langages avec des parenthèses. Notre pseudo-code développe ici cette puissance en neuf facteurs égaux à 2.

Décomposer et traduire

Pseudo-code

```
// example.pseudo
// group_operations
first <- 2 + 3 * 4
second <- (2 + 3) * 4
third <- (20 / 5) * 2
fourth <- 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2
PRINT first, second, third, fourth
```

Python

```
# group_operations.py
first = 2 + 3 * 4
second = (2 + 3) * 4
third = 20 / 5 * 2
fourth = 2 ** (3 ** 2)
print(first, second, third, fourth)
```

MATLAB

```
% group_operations.m
first = 2 + 3 * 4;
second = (2 + 3) * 4;
third = 20 / 5 * 2;
fourth = 2 ^ (3 ^ 2);
disp([first, second, third, fourth]);
```

Trace des regroupements

Expression	Étape intermédiaire	Résultat
2 + 3 * 4	2 + 12	14

Expression	Étape intermédiaire	Résultat
(2 + 3) * 4	5 * 4	20
20 / 5 * 2	4 * 2	8
2 puissance (3 puissance 2)	2 puissance 9	512

L'affectation intervient après le calcul à droite. L'ordre des lignes et le regroupement des opérateurs sont deux questions distinctes. Les exemples sont sans appels qui modifient des données ; le regroupement ne suffit pas à décrire tous les effets d'une expression plus générale.

Vérifier une traduction

Prédisez `18 / 3 / 2` : 3. Ajoutez les parenthèses `18 / (3 / 2)` : 12. Retirez ensuite celles des puissances dans les versions Python et MATLAB : observez 512 et 64. Cette différence explique pourquoi traduire un symbole ne suffit pas.

Rendre une intention visible

▼ **Comment écrire la moyenne de 8 et 12 ?**

`(8 + 12) / 2` vaut 10. Sans les parenthèses, `8 + 12 / 2` vaut 14.

Les comparaisons et les connecteurs logiques complètent ces priorités au chapitre suivant. Gardez des parenthèses autour des comparaisons quand vous les combinez.

Références : **priorités Python** et **priorités MATLAB**.

À retenir - fiche mémo

Priorité

La multiplication se regroupe avant l'addition ; les parenthèses imposent le regroupement.

Associativité

À priorité égale, elle détermine le regroupement des opérateurs.

Traduction

Les puissances se regroupent à droite en Python, à gauche en MATLAB.

Pseudo-code

```
// example.pseudo
// explicit_grouping
first <- 2 + 3 * 4
second <- (2 + 3) * 4
```

Point de vigilance. Ne traduisez pas une chaîne de puissances sans expliciter ses parenthèses.

Consulter la cheatsheet : Fondations · **Télécharger la fiche du thème en PDF**

Version web : https://www.clearcode.fr/courses/programming/algorithmic/foundations/operator-precedence