ALGORITHMIQUE · CHAPITRE 3 · FONDATIONS

Choisir une opération arithmétique

Addition, soustraction, multiplication, division, quotient entier, modulo et puissance : choisissez une opération et vérifiez son domaine.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Distinguer opérateur, opérande, opération unaire et binaire.
- Relier quotient et reste à une division euclidienne.
- Traduire les opérations scalaires en Python et MATLAB.

Prérequis. Savoir lire une valeur et une affectation.

Combien de boîtes complètes ?

Répartir 17 objets dans des boîtes de capacité 5 pose deux questions : combien de boîtes peut-on remplir et combien d'objets restent-ils ? La division ordinaire donne 3,4 ; elle ne donne pas directement ces deux réponses entières.

Opérateurs, opérandes et domaine

Un **opérateur** indique une opération ; ses **opérandes** sont les valeurs utilisées. Dans `17 - 5`, la soustraction est **binaire** : deux opérandes. Dans `-5`, le signe moins est **unaire** : un seul opérande. Binaire ne désigne pas ici une base de numération.

Le **domaine** indique les valeurs autorisées. Nous travaillons ici sur des nombres scalaires (une seule valeur), avec un dividende entier non négatif et un diviseur entier strictement positif. Diviser par zéro est exclu.

Opérations sur des nombres scalaires

Opération	Python	MATLAB	Résultat
Addition	17 + 5	17 + 5	22
Soustraction	17 - 5	17 - 5	12
Multiplication	17 * 5	17 * 5	85
Division ordinaire	17 / 5	17 / 5	3,4
Quotient entier	17 // 5	floor(17 / 5)	3
Reste (modulo)	17 % 5	mod(17, 5)	2
Puissance	2 ** 3	2 ^ 3	8

Une puissance `2³` multiplie trois facteurs égaux à 2. En Python, `^` ne calcule pas une puissance : il réalise un OU exclusif sur les bits d'entiers.

Définir le quotient et le reste

Pour des entiers `n ≥ 0` et `d > 0`, la division euclidienne détermine les entiers q et r tels que `n = q × d + r` et `0 ≤ r < d`. Ici, `17 = 3 × 5 + 2`. Dans notre pseudo-code, `floor_divide` donne q et `MOD` donne r.

Pour des nombres négatifs, les conventions comptent : Python arrondit `//` vers moins l'infini. Ainsi `−7 // 3` vaut -3 et `−7 % 3` vaut 2. `floor(−7 / 3)` et `mod(−7, 3)` donnent ici les mêmes résultats en MATLAB. Tronquer vers zéro donnerait un autre quotient.

Calculer puis reconstituer

Pseudo-code

```
// example.pseudo
// divide_items
items <- 17
capacity <- 5
quotient <- floor_divide(items, capacity)
remainder <- items MOD capacity
PRINT quotient, remainder
PRINT 2 * 2 * 2
```

Python

```python
# divide_items.py
items = 17
capacity = 5
quotient = items // capacity
remainder = items % capacity
print(quotient, remainder)
print(2 ** 3)
```

MATLAB

```
% divide_items.m
items = 17;
capacity = 5;
quotient = floor(items / capacity);
remainder = mod(items, capacity);
disp([quotient, remainder]);
disp(2 ^ 3);
```

Le résultat attendu est 3 boîtes complètes, 2 objets restants, puis 8 pour la puissance. En MATLAB, `floor` arrondit vers le bas ; `mod` calcule le reste. Les crochets rassemblent les deux nombres pour les afficher sur une ligne.

Prédire les cas frontières

Remplacez 17 successivement par 0, 4, 5 et 6. Prévoyez les couples (q, r) : (0, 0), (0, 4), (1, 0), (1, 1). Exécutez puis vérifiez la reconstruction et les bornes du reste. À 5 exactement, le quotient augmente et le reste revient à zéro.

Choisir et expliquer une opération

▼ **Quel calcul teste si un entier non négatif est pair ?**

Son reste modulo 2 vaut zéro. Un entier est pair lorsqu'il peut être écrit comme deux fois un entier. Pour 18, le reste est 0 ; pour 17, il est 1.

▼ **Pourquoi 17 / 5 ne suffit-il pas pour compter les boîtes pleines ?**

Il produit 3,4, alors que le contrat demande un nombre entier de boîtes complètes. Il faut le quotient entier 3 et, si nécessaire, le reste 2.

À retenir - fiche mémo

Opérande

Une valeur sur laquelle agit un opérateur ; un opérateur unaire en utilise une, un binaire deux.

Division euclidienne

Pour $n \geq 0$ et $d > 0$, $n = q \times d + r$ avec $0 \leq r < d$.

Puissance

Python écrit ** ; MATLAB écrit ^ pour les scalaires.

Pseudo-code

```
// example.pseudo
// divide_items
quotient <- floor_divide(17, 5)
remainder <- 17 MOD 5
```

Point de vigilance. Python // arrondit vers moins l'infini, pas vers zéro.

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

Version web : https://www.clearcode.fr/courses/programming/algorithmic/foundations/arithmetic-operators