ALGORITHMIQUE · CHAPITRE 14 · CONDITIONS ET BOUCLES

Comprendre ce qu'est une boucle

Découvrez le corps d'une boucle, son compteur et son état à chaque tour. Comparez trois répétitions en pseudo-code, Python et MATLAB.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Identifier le corps d'une boucle.
- Tracer l'état après chaque tour.
- Distinguer répétition et instruction placée après le bloc.

Prérequis. Savoir suivre des affectations et des conditions.

Répéter sans recopier

Un compteur commence à zéro et doit être augmenté trois fois. On pourrait écrire trois fois la même affectation. Une **boucle** exprime cette répétition dans un bloc unique et permet de faire varier le nombre de tours sans recopier les instructions.

Un tour peut contenir plusieurs instructions

Le **corps** de notre boucle contient deux actions : augmenter `count`, puis l'afficher. Un **tour** exécute tout ce corps une fois. Trois tours produiront donc trois augmentations et trois affichages. L'état obtenu à la fin d'un tour devient l'état de départ du suivant.

Distinguer compteur, tour et corps de boucle

Une **boucle** répète un groupe d'instructions appelé son **corps**. Une **itération** est un passage dans ce corps. Un **compteur** est une variable qui compte des événements ; l'**initialisation** fixe sa valeur avant de commencer. Ici, `count` vaut 0 avant les répétitions et augmente de 1 à chaque tour.

`REPEAT 3 TIMES` signifie "répéter trois fois" dans notre pseudo-code. Cette construction fixe le nombre de répétitions à l'avance : elle ne signifie pas "répéter jusqu'à obtenir une réponse". `END REPEAT` termine le bloc répété. La traduction Python utilise `range(3)`, qui fournit trois positions successives, et MATLAB utilise `1:3`, qui fournit 1, 2 et 3.

La variable parcourue par la boucle et le compteur de notre résultat ont des rôles distincts. Python peut parcourir 0, 1 et 2 alors que l'affichage de `count` produit 1, 2 et 3. Pour éviter la confusion, écrivez deux colonnes dans la trace : numéro du tour et valeur du compteur après la mise à jour.

Déplacer l'initialisation dans le corps remettrait le compteur à zéro à chaque tour. Déplacer l'affichage avant l'incrémentation montrerait 0, 1 et 2. Le nombre d'itérations resterait trois, mais les observations seraient différentes. Une boucle se comprend donc par les instructions répétées et par leur ordre, pas seulement par le mot-clé utilisé.

Un premier exemple de répétition

Pseudo-code

```
// example.pseudo
// count_repetitions
count <- 0
REPEAT 3 TIMES
    count <- count + 1
    PRINT count
END REPEAT
```

Python

```python
# count_repetitions.py
count = 0
for step in range(3):
    count = count + 1
    print(count)  # 1, then 2, then 3
```

MATLAB

```matlab
% count_repetitions.m
count = 0;
for step = 1:3
    count = count + 1;
    disp(count); % 1, then 2, then 3
end
```

`range(3)` fournit ici trois valeurs successives en Python ; `1:3` en fournit trois en MATLAB. Nous utilisons ces valeurs seulement pour déclencher trois tours : c'est `count`, et non la valeur de `step`, qui représente notre compteur. Les bornes seront étudiées dans le chapitre POUR.

Voir les états successifs

Trois tours d'une même boucle

Moment	count avant	count après	Affichage
Tour 1	0	1	1
Tour 2	1	2	2
Tour 3	2	3	3

L'initialisation `count ← 0` est avant la boucle. La placer dans le corps ferait repartir le compteur de zéro à chaque tour : on afficherait alors trois fois 1. Cette différence ne vient pas du nombre de répétitions, mais de la position de l'affectation.

Déplacer un affichage

Imaginez maintenant que l'affichage est placé après la boucle. Le compteur atteint toujours 3, mais on ne voit plus les états intermédiaires : une seule valeur est affichée. Prévoir ce qui est observable demande donc de distinguer les instructions répétées des instructions communes après la répétition.

Compter les exécutions

▼ Si le corps contient deux affichages et que la boucle fait quatre tours, combien d'affichages a-t-on ?

Huit : chacun des quatre tours exécute les deux affichages. Il faut compter les instructions du corps, pas seulement les tours.

▼ Que se passe-t-il si le nombre de tours est zéro ?

Le corps n'est jamais exécuté. Le compteur garde sa valeur initiale zéro ; les instructions situées après la boucle restent exécutées normalement.

Deux façons de décider de répéter

On peut répéter tant qu'une condition reste vraie, ou parcourir un ensemble de valeurs prévu à l'avance. Les chapitres TANT QUE et POUR expliquent ces deux façons de contrôler les tours. Dans les deux cas, une trace permet de comprendre ce qui change réellement.

À retenir - fiche mémo

Corps

Les instructions répétées à chaque tour.

Compteur

Une variable qui compte ici les répétitions déjà effectuées.

Trace

Distinguez numéro du tour et valeur du compteur après la mise à jour.

Pseudo-code

```
// example.pseudo
// count_repetitions
count <- 0
REPEAT 3 TIMES
    count <- count + 1
END REPEAT
```

Point de vigilance. Initialiser le compteur dans le corps le remettrait à zéro à chaque tour.

Consulter la cheatsheet : Conditions et boucles · **Télécharger la fiche du thème en PDF**

Version web : https://www.clearcode.fr/courses/programming/algorithmic/control-flow/loops