ALGORITHMIQUE · CHAPITRE 12 · CONDITIONS ET BOUCLES

Gérer plusieurs possibilités

Ordonnez plusieurs conditions sans oublier les frontières. Comparez if/elif/else en Python et if/elseif/else en MATLAB avec des scores de référence.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Lire une chaîne SI / SINON SI / SINON.
- Expliquer pourquoi la première condition vraie gagne.
- Vérifier la couverture des intervalles.

Prérequis. Savoir suivre un bloc SI / SINON.

Trois résultats possibles

Nous conservons des scores entiers entre 0 et 20. À partir de 16 : "Excellent". Entre 10 inclus et 16 exclu : "Passed". En dessous de 10 : "Try again". Chaque score doit recevoir exactement un message.

Tester du cas le plus restrictif au cas général

Un score de 18 vérifie à la fois `score ≥ 16` et `score ≥ 10`. Une chaîne **SI / SINON SI** s'arrête à la première condition vraie. En testant d'abord 16, on préserve le cas le plus précis. Le dernier SINON reçoit les scores qui n'ont satisfait aucun test précédent.

Comprendre une chaîne de décisions exclusives

Une **chaîne de conditions** examine des tests dans un ordre donné et exécute la première branche dont le test est vrai. Le pseudo-code écrit `ELSE IF`, Python `elif` et MATLAB `elseif`. Dans cette construction, les branches suivantes sont ignorées dès qu'une branche a été choisie.

Les messages du programme sont en anglais : `"Excellent"` correspond à un score de 16 à 20, `"Passed"` à un score de 10 à 15, et `"Try again"` à un score de 0 à 9. Les valeurs hors de 0 à 20 ne font pas partie du contrat de cette démonstration ; la dernière branche ne constitue pas une validation universelle des entrées.

Pour 17, les comparaisons `score >= 16` et `score >= 10` sont toutes deux vraies. C'est l'ordre décroissant des seuils qui conserve le message le plus précis. Deux instructions `if` indépendantes n'ont pas ce comportement : elles peuvent toutes deux être exécutées et la seconde peut remplacer le message de la première.

Avant de coder, découpez le domaine en intervalles sans trou ni recouvrement pour la décision attendue : [0, 10[, [10, 16[et [16, 20]. La notation [a, b[signifie que a est inclus et b exclu. Comparez ensuite les frontières 9, 10, 15 et 16 pour vérifier l'implémentation.

Comparer trois branches

Pseudo-code

```
// example.pseudo
// classify_score
score <- 16
IF score >= 16 THEN
    message <- "Excellent"
ELSE IF score >= 10 THEN
    message <- "Passed"
ELSE
    message <- "Try again"
END IF
PRINT message
```

Python

```python
# classify_score.py
score = 16
if score >= 16:
    message = "Excellent"
elif score >= 10:
    message = "Passed"
else:
    message = "Try again"
print(message)
```

MATLAB

```matlab
% classify_score.m
score = 16;
if score >= 16
    message = 'Excellent';
elseif score >= 10
    message = 'Passed';
else
    message = 'Try again';
end
disp(message);
```

Le test `score ≥ 10` est atteint seulement si le premier était faux. On sait donc déjà que le score est inférieur à 16 : il est inutile de répéter cette borne dans cette branche. Python écrit `elif`, MATLAB `elseif`.

Suivre les tests réellement évalués

Ordre d'évaluation d'une chaîne de conditions

score	Test ≥ 16	Test ≥ 10	Résultat
9	faux	faux	Try again

score	Test ≥ 16	Test ≥ 10	Résultat
10	faux	vrai	Passed
15	faux	vrai	Passed
16	vrai	Non évalué	Excellent

Un test non évalué n'est pas un test faux : le programme n'a pas besoin de connaître son résultat pour cette exécution. Cette distinction permet de comprendre un chemin sans inventer des étapes supplémentaires.

Comparer deux écritures

▼ Pourquoi commencer par score ≥ 10 masquerait-il la mention "Excellent" ?

Les scores 16 à 20 satisferaient déjà ce premier test. Dans une chaîne avec SINON SI, on n'atteindrait jamais le test plus restrictif placé ensuite.

▼ Deux blocs SI indépendants auraient-ils le même effet ?

Pas nécessairement. Avec deux SI indépendants testant score ≥ 16 puis score ≥ 10, un score de 18 peut déclencher les deux affichages. La chaîne utilisée ici exprime un choix exclusif.

Énoncer la règle avant les tests

Un tableau des intervalles permet de repérer les cas manquants et les chevauchements avant le code. Lorsque plusieurs conditions sont vraies, l'ordre des tests devient une partie de la règle. Il doit donc être expliqué, pas seulement observé sur un exemple.

À retenir - fiche mémo

Priorité

La chaîne exécute la première branche dont la condition est vraie.

Traduction

ELSE IF correspond à elif en Python et elseif en MATLAB.

Seuils

Pour les scores, vérifier 16 avant 10 conserve le cas le plus précis.

Pseudo-code

```
// example.pseudo
// classify_score
IF score >= 16 THEN
    PRINT "Excellent"
ELSE IF score >= 10 THEN
    PRINT "Passed"
ELSE
    PRINT "Try again"
END IF
```

Point de vigilance. Deux if indépendants peuvent s'exécuter tous les deux ; ce n'est pas une chaîne exclusive.

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

Version web : https://www.clearcode.fr/courses/programming/algorithmic/control-flow/conditions/multiple-branches