ALGORITHMIQUE · CHAPITRE 11 · CONDITIONS ET BOUCLES

Décider avec SI / SINON

Comprenez if et else avec un score, un diagramme animé et des exemples Python et MATLAB. Distinguez condition, branche et suite du programme.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Évaluer une condition avant de choisir une branche.
- Lire un organigramme SI / SINON.
- Identifier les instructions communes aux deux chemins.

Prérequis. Savoir lire une comparaison et une affectation.

Une seule décision, deux actions

Dans cet exercice, un score entier compris entre 0 et 20 est validé à partir de 10 inclus. Pour un score de 12, doit-on afficher les deux messages possibles ? Non : on évalue d'abord une condition, puis on suit une seule branche. La valeur 10 appartient explicitement au cas validé.

Nommer le test, les branches et leur point de rencontre

Une **condition** est une expression dont le résultat est vrai ou faux. Une **branche** regroupe les instructions choisies selon ce résultat. `IF` signifie "si", `THEN` introduit l'action et `ELSE` signifie "sinon". `END IF` ferme le choix dans notre pseudo-code. Python utilise `if` et `else` ; MATLAB ajoute un `end` final.

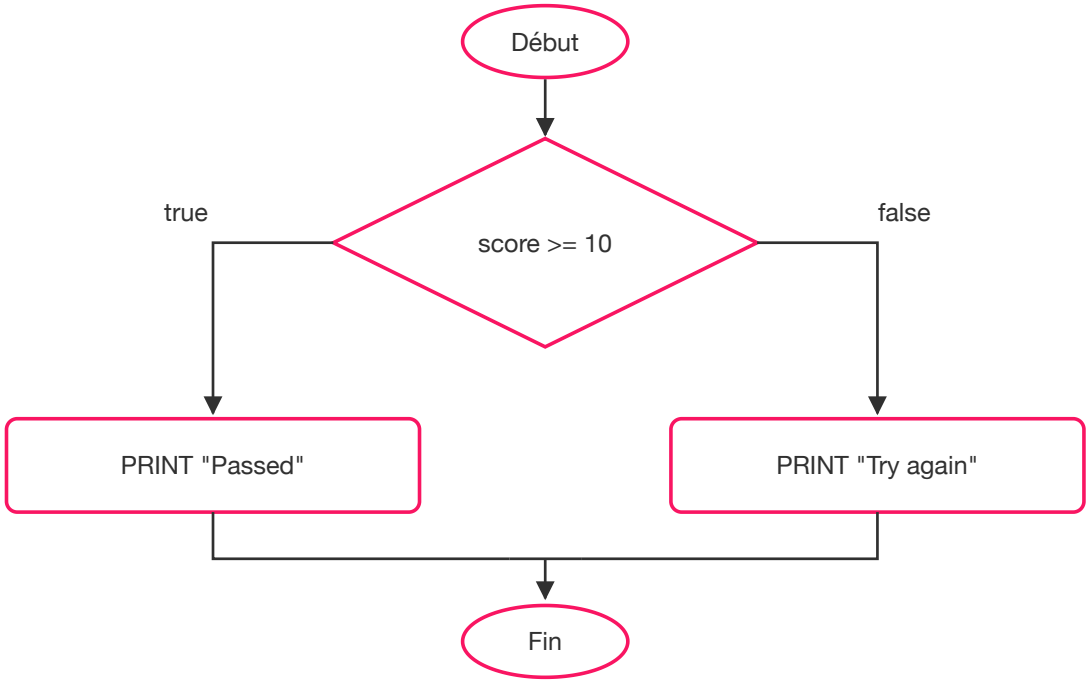
Dans cet exemple, un score entier de 0 à 20 est l'entrée. Le test `score >= 10` décide entre deux messages anglais : `"Passed"` pour un score validé et `"Try again"` sinon. `"Processing complete"` est affiché après le choix dans les deux cas. Cette troisième instruction n'appartient à aucune des deux branches.

Un **bloc** rassemble plusieurs instructions. En Python, l'**indentation**, c'est-à-dire les espaces placés au début des lignes, délimite le bloc. En MATLAB, les mots-clés et `end` le délimitent. Les espaces aident toujours à lire la structure, même lorsqu'ils ne jouent pas exactement le même rôle syntaxique.

Le losange du schéma porte le test ; chaque flèche de sortie indique la réponse qui autorise le chemin. Les rectangles décrivent des actions. Suivez seulement une branche par essai : 9 choisit le second message, 10 choisit le premier. Le passage à la suite du programme ne refait pas automatiquement le test.

La condition choisit le chemin

Le test est effectué une fois ; une seule des deux branches est suivie.



▼ **Lire les parcours sans animation**

score = 12

1. Début du parcours avec score = 12. Les données sont déjà disponibles.
2. Test 1 : score >= 10 donne true (vrai).
3. Action choisie : PRINT "Passed".
4. Les chemins alternatifs se rejoignent. Une seule branche a été exécutée ; il n’y a pas deux tâches à attendre.
5. Fin du parcours. Résultat observable : Passed.

score = 9

1. Début du parcours avec score = 9. Les données sont déjà disponibles.
2. Test 1 : score >= 10 donne false (faux).
3. Action choisie : PRINT "Try again".
4. Les chemins alternatifs se rejoignent. Une seule branche a été exécutée ; il n’y a pas deux tâches à attendre.
5. Fin du parcours. Résultat observable : Try again.

score = 10

1. Début du parcours avec score = 10. Les données sont déjà disponibles.
2. Test 1 : score >= 10 donne true (vrai).
3. Action choisie : PRINT "Passed".
4. Les chemins alternatifs se rejoignent. Une seule branche a été exécutée ; il n’y a pas deux tâches à attendre.
5. Fin du parcours. Résultat observable : Passed.

Le losange pose une question dont la réponse est vrai ou faux. Les flèches orientent l'exécution. Après la branche choisie, les chemins se rejoignent : l'instruction qui suit le bloc SI / SINON sera exécutée dans les deux cas.

Relier indentation et bloc

Pseudo-code

```
// example.pseudo
// check_score
score <- 12
IF score >= 10 THEN
    PRINT "Passed"
ELSE
    PRINT "Try again"
END IF
PRINT "Processing complete"
```

Python

```python
# check_score.py
score = 12
if score >= 10:
    print("Passed")
else:
    print("Try again")
print("Processing complete")
```

MATLAB

```
% check_score.m
score = 12;
if score >= 10
    disp('Passed');
else
    disp('Try again');
end
disp('Processing complete');
```

En Python, les lignes indentées appartiennent à une branche ; `else` est aligné avec `if`. En MATLAB, `end` ferme le bloc. Le dernier affichage est après le bloc dans les deux versions : il ne dépend pas de la branche choisie.

Observer les deux parcours

Trois entrées, deux chemins possibles

score	score ≥ 10	Message choisi
9	faux	À reprendre
10	vrai	Validé
12	vrai	Validé

Modifiez seulement le score et prédisez d'abord le message. Observez ensuite que "Traitement terminé" apparaît toujours. La variable score ne change pas : une condition lit des valeurs mais ne les modifie pas automatiquement.

Repérer une mauvaise traduction

▼ Que changerait un test score > 10 ?

Le score 10 suivrait la branche "À reprendre", ce qui contredit la règle "à partir de 10 inclus". La différence porte sur la frontière, pas sur la syntaxe de l'affichage.

▼ Peut-on enlever SINON si aucune action n'est nécessaire dans le cas faux ?

Oui. Si la condition est fausse, on passe directement aux instructions placées après le bloc. Il faut toutefois vérifier que les variables utilisées ensuite ont été définies sur tous les chemins utiles.

Condition et action ont deux rôles

`score ≥ 10` décrit un test. "Afficher Validé" décrit une action. En expliquant un algorithme, séparez toujours la question posée, la réponse obtenue et les instructions réellement exécutées.

À retenir - fiche mémo

IF

Exécute la première branche lorsque la condition est vraie.

ELSE

Sélectionne l'autre branche lorsque la condition est fausse.

Suite

Après END IF, les instructions communes continuent dans les deux cas.

Pseudo-code

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

Point de vigilance. Un test et une affectation n'ont pas le même rôle.

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