ALGORITHMIQUE · CHAPITRE 26 · REPRÉSENTATIONS

Dessiner un organigramme lisible

Lisez un organigramme avec une légende explicite. Suivez les tests, actions, branches et retours de boucle avec des animations et le code associé.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Lire début, fin, traitement, décision et flèche.
- Suivre un seul chemin pour une entrée choisie.
- Traduire un choix et une boucle entre texte, schéma et pseudo-code.

Prérequis. Avoir suivi des conditions, des boucles et des appels simples.

Que doit montrer un schéma ?

Pour un nombre réel fini, nous voulons afficher s'il est négatif ou non négatif. Le schéma doit permettre à une personne de suivre les tests et les actions sans deviner l'ordre des instructions. Un dessin décoratif qui omet les conditions de ses flèches ne suffit pas à expliquer la méthode.

Données du problème et flèches de contrôle

Un **organigramme algorithmique** représente l'ordre des étapes à l'aide de formes et de flèches. Une flèche de **contrôle** indique quelle étape peut être exécutée ensuite. Elle ne dessine pas nécessairement le transfert d'une valeur. L'entrée `number` est supposée fournie avant le début ; la sortie est ici un message affiché.

Un nombre **fini** est ici un nombre ordinaire, sans valeur infinie ni donnée manquante. Le test `number < 0` est vrai pour −3, faux pour 0 et faux pour 4. Le message anglais `"Non-negative"` signifie "non négatif" et comprend zéro ; `"Positive"` serait incorrect pour zéro.

Dans les dessins de ce cours, l'ovale marque le début ou la fin, le losange porte un test et le rectangle porte une instruction. Les libellés `true` et `false` indiquent la réponse du test. Il faut choisir une seule flèche de sortie du losange pour un essai donné. La légende définit notre convention avant toute animation.

Une animation de parcours aide à suivre le sens des flèches ; elle n'est pas une preuve du programme pour toutes les entrées. Faites varier l'entrée proposée, puis expliquez pourquoi le chemin change. Pour une boucle, distinguez le retour au test et une nouvelle lecture de l'entrée : dans notre décompte, la valeur diminue, elle n'est pas réinitialisée.

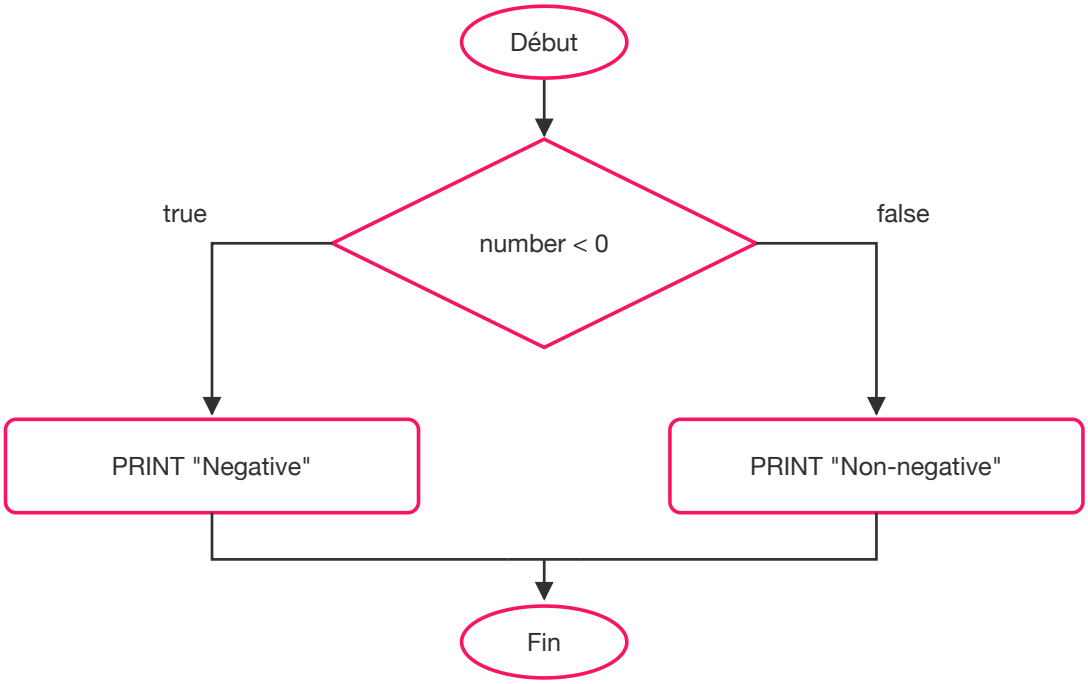
Annoncer la légende

- Une forme de début ou de fin borne le déroulement présenté.
- Un rectangle représente un traitement ; un parallélogramme peut représenter une entrée ou une sortie dans une notation d'organigramme.
- Un losange représente une décision et ses sorties doivent être distinguées.
- Les flèches donnent le sens de lecture, y compris lorsqu'une boucle revient vers un test.

Une légende fixe les conventions choisies. Le schéma ci-dessous écrit les affichages dans des rectangles de traitement pour garder un vocabulaire graphique réduit. Les formes et les libellés doivent rester cohérents à l'intérieur d'un même diagramme.

Suivre une décision

Pour zéro, le test est faux : on suit la branche "non négatif".



▼ **Lire les parcours sans animation**

number = -3

1. Début du parcours avec number = -3. Les données sont déjà disponibles.
2. Test 1 : number < 0 donne true (vrai).
3. Action choisie : PRINT "Negative".
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 : Negative.

number = 0

1. Début du parcours avec number = 0. Les données sont déjà disponibles.
2. Test 1 : number < 0 donne false (faux).
3. Action choisie : PRINT "Non-negative".
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 : Non-negative.

number = 4

1. Début du parcours avec number = 4. Les données sont déjà disponibles.
2. Test 1 : number < 0 donne false (faux).
3. Action choisie : PRINT "Non-negative".
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 : Non-negative.

Choisissez d'abord une entrée, par exemple −3. Évaluez le losange, suivez la flèche vrai, effectuez l'affichage, puis rejoignez la fin. Lors de cette exécution, la branche faux n'est pas visitée.

Retrouver les mêmes étapes dans le code

Pseudo-code

```
// example.pseudo
// classify_sign
number <- -3
IF number < 0 THEN
    PRINT "Negative"
ELSE
    PRINT "Non-negative"
END IF
```

Python

```python
# classify_sign.py
number = -3
if number < 0:
    message = "Negative"
else:
    message = "Non-negative"
print(message)
```

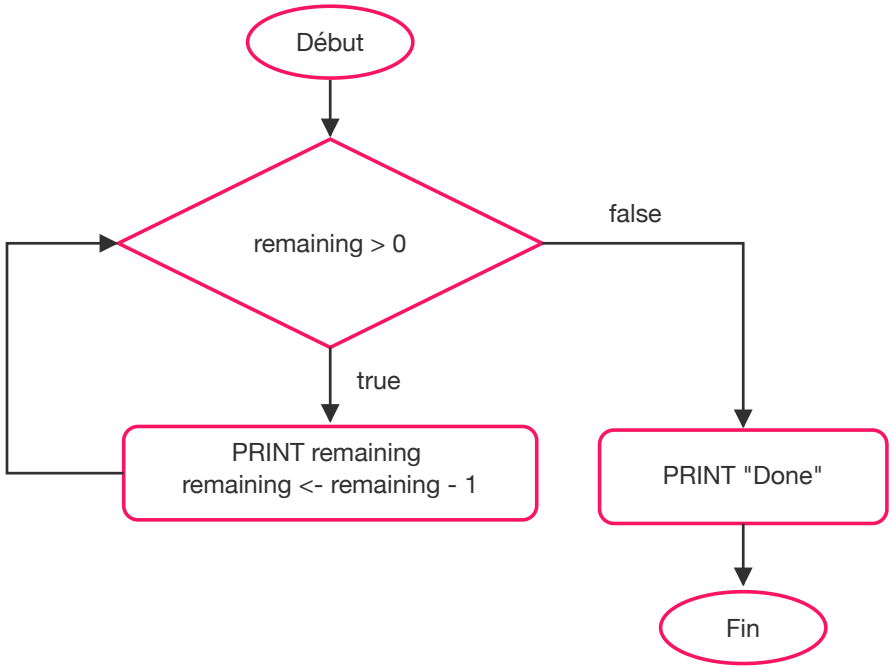
MATLAB

```matlab
% classify_sign.m
number = -3;
if number < 0
    message = 'Negative';
else
    message = 'Non-negative';
end
disp(message);
```

Le schéma ne crée pas une troisième règle : il représente la même condition que le pseudo-code et les deux langages. Changer "non négatif" en "positif" serait incorrect pour zéro, même si le dessin restait visuellement bien organisé.

Lire une flèche de retour

Une boucle revient au test ; elle ne retourne pas nécessairement au début de tout le programme.



▼ **Lire les parcours sans animation**

remaining = 3

1. Début du parcours avec remaining = 3. Les données sont déjà disponibles.
2. Test 1 avec remaining = 3 : remaining > 0 donne true (vrai).
3. Action choisie : PRINT remaining; remaining <- remaining - 1.
4. Test 2 avec remaining = 2 : remaining > 0 donne true (vrai).
5. Action choisie : PRINT remaining; remaining <- remaining - 1.
6. Test 3 avec remaining = 1 : remaining > 0 donne true (vrai).
7. Action choisie : PRINT remaining; remaining <- remaining - 1.
8. Test 4 avec remaining = 0 : remaining > 0 donne false (faux).
9. Action choisie : PRINT "Done".
10. Fin du parcours. Résultat observable : 3, 2, 1, Done.

remaining = 0

1. Début du parcours avec remaining = 0. Les données sont déjà disponibles.
2. Test 1 avec remaining = 0 : remaining > 0 donne false (faux).
3. Action choisie : PRINT "Done".
4. Fin du parcours. Résultat observable : Done.

Avant cette boucle, remaining doit être initialisé à un entier non négatif. Le corps réduit cette valeur de 1. Pour expliquer le schéma, dites ce qui a changé lorsque la flèche revient au losange. Revenir au test sans changer son résultat peut conduire à une boucle infinie.

Contrôler le schéma autant que le code

▼ Quelles informations manquent si les deux flèches d'un losange ne portent aucun libellé ?

On ne sait pas quel chemin correspond à une condition vraie et lequel correspond à une condition fausse. La position gauche/droite n'est pas une convention suffisante : les branches doivent être explicitement identifiées.

▼ Une flèche qui revient vers l'initialisation de remaining représente-t-elle la même boucle ?

Pas dans cet exemple. Réinitialiser remaining à chaque retour annulerait la diminution et pourrait empêcher l'arrêt. La destination exacte de la flèche fait partie de la méthode.

Comparer des représentations

Une bonne vérification consiste à choisir la même entrée et à parcourir successivement le texte, le schéma et le code. Les tests évalués, les actions et le résultat doivent correspondre. Le chapitre suivant introduit une autre notation graphique, UML, en conservant un périmètre très limité.

À retenir - fiche mémo

Ovale

Le début ou la fin dans les organigrammes de ce cours.

Losange

Un test et des sorties distinguées par leur condition.

Rectangle / flèche

Un traitement et le sens du parcours ; une boucle revient au test.

Pseudo-code

```
// example.pseudo
// classify_sign
IF number < 0 THEN
    PRINT "Negative"
ELSE
    PRINT "Non-negative"
END IF
```

Point de vigilance. Zéro n’est pas négatif, mais il n’est pas strictement positif non plus.

Consulter la cheatsheet : Organigrammes et UML · **Télécharger la fiche du thème en PDF**

Version web : https://www.clearcode.fr/courses/programming/algorithmic/representations/flowcharts