

Chapitre 12 – Cours

Algorithmique et raisonnement par étapes

Dans ce chapitre, tu vas apprendre à :

- Tracer une exécution
- Concevoir et contrôler une boucle
- Interpréter des tests et leurs frontières

Lire les affectations dans l'ordre

Définition – Variable et affectation

Une variable mémorise une valeur. L'instruction $x \leftarrow x + 1$ signifie : calculer l'ancienne valeur de x plus 1, puis enregistrer le résultat dans x . Ce n'est pas une équation à résoudre. Après une affectation, l'ancienne valeur est remplacée, sauf si elle a été sauvegardée dans une autre variable.

Exemple résolu – Échanger deux valeurs

Au départ, $a = 3$ et $b = 7$. Les instructions $a \leftarrow b$ puis $b \leftarrow a$ donneraient $a = b = 7$, car la valeur 3 serait perdue. Pour échanger les valeurs, on utilise une variable temporaire :

$$t \leftarrow a, \quad a \leftarrow b, \quad b \leftarrow t.$$

Après la première instruction, $t = 3$; après la deuxième, $a = 7$; après la troisième, $b = 3$. La variable temporaire a conservé l'ancienne valeur de a .

Choisir un test précis

Définition – Condition et frontières

Une condition est vraie ou fausse. Les tests $x < 5$ et $x \leq 5$ diffèrent pour $x = 5$. Pour vérifier un programme, essayer une valeur inférieure au seuil, une valeur égale et une valeur supérieure aide à contrôler le choix des branches. Un test d'égalité compare des valeurs ; une affectation modifie une valeur.

Construire une répétition

Définition – Compteur et accumulateur

Un compteur mémorise le nombre d'étapes effectuées ; on l'augmente souvent de 1 à chaque tour. Un accumulateur mémorise une somme ou un produit construit progressivement. Les valeurs initiales comptent : une somme commence souvent à 0, tandis qu'un produit commence souvent à 1.

Exemple résolu – Additionner les entiers de 1 à 4

On initialise $n = 0$ et $s = 0$. Tant que $n < 4$, on remplace d'abord n par $n + 1$, puis s par $s + n$. Les couples $(n; s)$ après chaque tour sont

$$(1; 1), \quad (2; 3), \quad (3; 6), \quad (4; 10).$$

Le programme affiche donc 10, qui est bien $1 + 2 + 3 + 4$. Si l'on ajoutait n avant de l'augmenter, on additionnerait $0 + 1 + 2 + 3$ et l'on obtiendrait 6 : l'ordre des instructions change le résultat.

Propriété – Contrôler l'arrêt

Dans cet exemple, n commence à 0 et augmente de 1 à chaque tour. Il atteint donc 4 après quatre tours, et le test $n < 4$ devient faux. Si l'instruction qui augmente n était supprimée, la condition resterait vraie et la boucle ne s'arrêterait pas. Pour une boucle à condition, expliquer pourquoi une variable finit par atteindre ou dépasser le seuil.

La trace d'exécution

tour	0	1	2	3	4
n	0	1	2	3	4
s	0	1	3	6	10

La colonne 0 décrit l'état avant la boucle ; les suivantes décrivent l'état après un tour complet. Le test suivant la dernière colonne est faux.

À retenir – Bilan

Tracer les valeurs après chaque instruction sensible. Dans une boucle, distinguer compteur, résultat accumulé et condition d'arrêt. Contrôler les valeurs au seuil et l'ordre des affectations permet d'expliquer une erreur, puis de la corriger.
