
Algorithmique

CHAPITRE 6 : LES BOUCLES

Sommaire

1. Pourquoi les boucles ?
2. Définition
3. Les types de boucles
4. Notation des boucles conditionnelles en pseudo-code
5. Exemples avec les boucles conditionnelles
6. Notation des boucles inconditionnelles en pseudo-code
7. Exemples avec les boucles inconditionnelles
8. Erreurs à éviter
9. Exercices
10. Corrections

1. Pourquoi les boucles ?

- Imaginons qu'on veuille écrire les algorithmes suivants :
 - **Sujet 1** : Demander à l'utilisateur, un nombre positif et calculer le double de ce nombre. Tant que le nombre saisi n'est pas correct afficher « Vous devez saisir un nombre positif », et lui demander de saisir un nombre. On calculera le double lorsque l'utilisateur aura saisi un nombre positif.
 - **Sujet 2** : Soit un nombre magique 24, demander à l'utilisateur de deviner le nombre magique. Tant que l'utilisateur ne trouve pas le nombre magique afficher « Veuillez entrer le nombre magique: ». Si l'utilisateur entre un nombre inférieur au nombre magique afficher « Le nombre magique est plus grand. ». Si l'utilisateur entre un nombre supérieur au nombre magique afficher « Le nombre magique est plus petit. ». Sinon si l'utilisateur trouve le nombre magique afficher « Bingo, vous avez trouvé le nombre magique. ».
 - **Sujet 3** : Demander un nombre à l'utilisateur et afficher la somme de tous les nombres compris entre 0 et ce nombre. Par exemple si l'utilisateur saisit 4, l'algorithme doit faire le calcul $1 + 2 + 3$ et afficher : « La somme des nombres entre 0 et 4 est 6. »
- La résolution des sujets précédents fait appel à la notion d'actions qui devront se répéter une ou plusieurs fois par exemple dans le sujet 1, l'on va afficher le message « Veuillez entrer un nombre positif : » tant que l'utilisateur n'aura pas saisi un nombre positif. Pour le sujet 2, l'algorithme va demander le nombre magique à l'utilisateur tant que celui-ci n'aura pas fourni le nombre magique 24.
- Ce type d'actions se nomme des **boucles** ou **répétitions** ou **itérations**.

2. Définition

- Les **boucles** ou **structures répétitives** ou **structures itératives** sont des types d'instructions algorithmique qui permettent d'exécuter à plusieurs reprises une suite d'instructions.
- Grâce aux boucles on va pouvoir répéter une suite d'instructions jusqu'à ce qu'une condition spécifique soit atteinte.
- Contrairement aux structures de test qui exécutent une suite d'instructions uniquement une fois, la boucle va quant à elle exécuter la suite d'instructions plusieurs fois.
- Comme avec les tests, on rencontre régulièrement les boucles, par exemple lors de la saisie de notre mot de passe, l'ordinateur nous demandera de saisir le mot de passe tant que le mot de passe entré ne correspond pas au mot de passe enregistré.
- En appliquant le cas des mots de passe aux boucles, on peut avoir comme raisonnement :
 - Tant que le mot de passe entré par l'utilisateur n'est pas conforme au mot de passe enregistré demander le mot de passe.
 - Si le mot de passe est conforme connecter l'utilisateur.
 - Sinon interdire la connexion et redemander le mot de passe.

3. Les types de boucles

- On distingue deux catégories de boucles dans la majorité des langages de programmation :
 - **Les répétitions conditionnelles (ou indéfinies)** : ici on répète une suite d'instructions selon une certaine condition. Ce type de répétition est utilisé pour les mots de passe, on ressaisit le mot de passe tant que celui-ci n'est pas correct. Avec ce type de boucle on ne peut savoir par avance le nombre de fois que l'utilisateur va saisir son mot de passe, c'est pourquoi on parle de **boucle indéfinie**.
 - **Les répétitions inconditionnelles (ou avec compteur ou définies)** : Contrairement au précédent, ici les répétitions concernées sont exécutées un nombre de fois donné. On peut les retrouver lors de la saisie de certains mots de passe avec un nombre limite de tentatives sous peine de voir son compte bloqué après X tentatives. Par exemple sur les applications de comptes bancaire, il arrive qu'après trois essais erronés du mot de passe, le compte soit bloqué. Dans ce cas de figure, la répétition inconditionnelle a un compteur max de 3 essais.
- A partir de ces deux types de catégories, on peut donc assigner un type de boucles à chacun de nos sujets précédents :
 - Sujet 1 : Répétition conditionnelle
 - Sujet 2 : Répétition conditionnelle
 - Sujet 3 : Répétition inconditionnelle



On peut bien résoudre chacun des sujets avec chaque type de répétition, si on le souhaite. Mais l'idéal est que lorsqu'on connaît le nombre de fois dont on devra faire la répétition, il faut préférer les boucles à compteur aux boucles indéfinies.

4. Notation des boucles conditionnelles en pseudo-code

- On dispose de deux notations :

```
TANTQUE condition FAIRE
...
FINTANTQUE
```

OU

```
TantQue condition Faire
.....
FinTantQue
```

```
REPETER
...
TANTQUE condition
```

Ou

```
Répéter
...
TantQue condition
```

- Il existe une différence entre les deux types de répétitions:
 - Avec la répétition **TantQue ... Faire**, la condition de répétition est évaluée avant répétitions. La boucle ici est exécutée si et seulement si la condition d'exécution est vérifiée.
 - Avec la répétition **Répéter ... TantQue**, la condition est évaluée après la première exécution de la répétition. Ici la boucle est exécutée au moins une fois.



Cette notation est juste un choix conventionnel, pour les langages de programmation il faudra se référer à la documentation du langage.

5. Exemples avec les boucles conditionnelles

Sujet 1 : Demander à l'utilisateur, un nombre positif et calculer le double de ce nombre. Tant que le nombre saisi n'est pas correct afficher « Vous devez saisir un nombre positif », et lui demander de saisir un nombre. On calculera le double lorsque l'utilisateur aura saisi un nombre positif.

Début

Variables nombre, double en **Entiers**

Ecrire "Entrer un nombre:"

Lire nombre

TantQue nombre < 0 **Faire**

 Ecrire "Vous devez saisir un nombre positif"

 Ecrire "Entrer un nombre:"

 Lire nombre

FinTantQue

double <- nombre * 2

Ecrire "Le double de " + nombre + " est :" + double

Fin

5. Exemples avec les boucles conditionnelles

Sujet 2 : Soit un nombre magique 24, demander à l'utilisateur de deviner le nombre magique. Tant que l'utilisateur ne trouve pas le nombre magique afficher « Veuillez entrer le nombre magique: ». Si l'utilisateur entre un nombre inférieur au nombre magique afficher « Le nombre magique est plus grand. ». Si l'utilisateur entre un nombre supérieur au nombre magique afficher « Le nombre magique est plus petit. ». Sinon si l'utilisateur trouve le nombre magique afficher « Bingo, vous avez trouvé le nombre magique. ».

Début

Variables nombreMagique, nombreSaisi en **Entiers**

nombreMagique \leftarrow 24

Répéter

Ecrire "Entrer le nombre magique:"

Lire nombreSaisi

Si nombreSaisi < nombreMagique **Alors**

Ecrire "Le nombre magique est plus grand"

SinonSi nombreSaisi > nombreMagique **Alors**

Ecrire "Le nombre magique est plus petit"

Sinon

Ecrire "Bingo, vous avez trouvé le nombre magique"

FinSi

TantQue nombreMagique \neq nombreSaisi

Fin

6. Notation des boucles inconditionnelles en pseudo-code

Pour utiliser une boucle inconditionnelle, on doit fournir quatre éléments :

- **La variable** qui va contenir la valeur du compteur.
- **La valeur de départ** qui est la valeur à laquelle on va commencer à compter.
- **La valeur de fin** qui est la valeur à laquelle on va arrêter de compter.
- **Le pas** qui décrit comment l'on va compter.

En pseudo code on utilisera la notation :

```
Pour nomVariable <- nombreDepart à nombreFin Pas nombreDePas Faire
...
Suivant

Ou

POUR nomVariable <- nombreDepart A nombreFin PAS nombreDePas FAIRE
...
Suivant
```

6. Notation des boucles inconditionnelles en pseudo-code

Soit les boucles suivantes :

```
Boucle 1
Pour a <- 0 à 10 Pas 1 Faire
    Ecrire a + " "
Suivant

Boucle 2
Pour a <- 0 à 10 Pas 2 Faire
    Ecrire a + " "
Suivant
```

- Dans la boucle 1, la variable « a » va prendre les valeurs de 0 inclus à 10 exclu ce qui donnera l'affichage « 0 1 2 3 4 5 6 7 8 9 ».
- Dans la boucle 2, la variable « a » va prendre les valeurs de 0 inclus à 10 exclu en prenant chaque deuxième nombre ce qui donnera l'affichage « 0 2 4 6 8 ».

Question 1: Quelle boucle permet d'avoir « 1 4 7 10 » ?

Question 2: Quelle boucle permet d'avoir « 1, 2, 3, ... 100 » ?

7. Exemples avec les boucles inconditionnelles

Sujet 3 : Demander un nombre à l'utilisateur et afficher la somme de tous les nombres compris entre 0 et ce nombre. Par exemple si l'utilisateur saisit 4, l'algorithme doit faire le calcul $1 + 2 + 3$ et afficher : « La somme des nombres entre 0 et 4 est 6. »

Début

Variables nombre, somme, compteur en Entiers

Ecrire "Entrer un nombre:"

Lire nombre

somme \leftarrow 0

Pour compteur \leftarrow 0 à nombre Pas 1 Faire

 somme \leftarrow compteur + somme

Suivant

Ecrire "La somme des nombres entre 0 et " + nombre + " est " + somme

Fin

8. Erreurs à éviter

- Lorsqu'on utilise une boucle conditionnelle, il faut s'assurer que celle-ci dispose bien d'une condition valide et qui peut évoluer c'est-à-dire qui peut passer de **Vrai** à **Faux** et vice versa. Il faut éviter les algorithmes comme :

```
Début
Variables a, b en Entiers
b <- 0
a <- 1
Répéter
    b <- b + 1
TantQue a = 1
Ecrire b
Fin
```

- Dans l'exemple précédent, nous utilisons la valeur de la variable « a » comme condition de fin de boucle, sauf que la valeur de la variable n'évolue pas, on se retrouve donc dans un cas de boucle infinie. Ce programme s'exécutera sans arrêt. La ligne « Ecrire b » ne s'exécutera jamais.



Les boucles sont utiles quand elles sont utilisées correctement. Mais peuvent être sources d'ennuis quand elles sont mal utilisées. Avant d'utiliser une boucle il faut surtout bien réfléchir à trouver la condition de démarrage mais aussi la condition d'arrêt.

9. Exercices

Exercice 1

Reprendre le sujet 3 mais en utilisant les boucles TantQue ... Faire et Répéter ... TantQue. Faire un programme pour chacune de ces boucles.

Exercice 2

Ecrire un algorithme qui demande à l'utilisateur un nombre compris entre 1 et 3 jusqu'à ce que la réponse convienne.

Exercice 3

Ecrire un algorithme qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : « Plus petit ! », et inversement, « Plus grand ! » si le nombre est inférieur à 10.

9. Exercices

Exercice 4

Ecrire un algorithme qui demande un nombre de départ, et qui affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Exercice 5

Ecrire un algorithme qui demande un nombre de départ, et qui calcule la somme des entiers jusqu'à ce nombre (inclus). Par exemple, si l'on entre 5, le programme doit calculer $1 + 2 + 3 + 4 + 5 = 15$.

Exercice 6

Ecrire un algorithme qui demande un nombre à l'utilisateur et qui ensuite écrit la table de multiplication de ce nombre de 0 à 10.

Exercice 7

Ecrire un algorithme qui demande un nombre à l'utilisateur et qui calcule le factoriel de ce nombre. Par exemple si l'utilisateur saisit 8, le factoriel de 8 noté $8!$ vaut : $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 40320$. Pour 3 on aura $3! = 3 \times 2 \times 1 = 6$.

9. Exercices

Exercice 8

Ecrire un programme qui demande successivement 20 nombres à l'utilisateur, et qui lui dise ensuite quel était le plus grand parmi ces 20 nombres. Exemple :

Entrer le nombre numéro 1 : 16

.....

Entrer le nombre numéro 20: 12

Le plus grand de ces nombres est 16.

Exercice 9

Reprendre l'algorithme précédent mais il faut afficher la position à laquelle le nombre a été saisi.

Par exemple : le plus grand de ces nombre est 16 et c'est le nombre 2.

10. Corrections

Exercice 1

Reprendre le sujet 3 mais en utilisant les boucles TantQue ... Faire et Répéter ... TantQue. Faire un programme pour chacune de ces boucles.

Début

```
Variables nombre, somme, compteur en Entiers
Ecrire "Entrer un nombre:"
Lire nombre
somme <- 0
compteur <- 0
TantQue compteur < nombre Faire
    somme <- somme + compteur
    compteur <- compteur + 1
FinTantQue
Ecrire "La somme des nombres entre 0 et " + nombre + " est :" + somme
```

Fin

10. Corrections

Exercice 1

Reprendre le sujet 3 mais cette fois en utilisant les boucles TantQue ... Faire et Répéter ... TantQue. Faire un programme pour chacune de ces boucles.

Début

Variables nombre, somme, compteur en Entiers

somme ← 0

compteur ← 0

Ecrire "Entrer un nombre:"

Lire nombre

Répéter

 somme ← somme + compteur

 compteur ← compteur + 1

TantQue compteur < nombre

Ecrire "La somme des nombres entre 0 et " + nombre + " est :" + somme

Fin

10. Corrections

Exercice 2

Ecrire un algorithme qui demande à l'utilisateur un nombre compris entre 1 et 3 jusqu'à ce que la réponse convienne.

Début

Variable nombre en Entier

Répéter

Ecrire "Entrer un nombre entre 1 et 3:"

Lire nombre

Si nombre < 1 Ou nombre > 3 Alors

Ecrire "Vous devez saisir un nombre entre 1 et 3"

FinSi

TantQue nombre < 1 Ou nombre > 3

Ecrire "Vous avez saisi " + nombre + " qui est compris entre 1 et 3"

Fin

10. Corrections

Exercice 3

Ecrire un algorithme qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : « Plus petit ! », et inversement, « Plus grand ! » si le nombre est inférieur à 10.

Début

Variable nombre en Entier

Ecrire "Entrer un nombre entre 10 et 20:"

Lire nombre

TantQue nombre < 10 Ou nombre > 20 Faire

Si nombre > 20 Alors

Ecrire "Plus petit !"

Sinon

Ecrire "Plus grand !"

FinSi

Ecrire "Entrer un nombre entre 10 et 20:"

Lire nombre

FinTantQue

Ecrire "Vous avez saisi " + nombre + " qui est compris entre 10 et 20"

Fin

10. Corrections

Exercice 4

Ecrire un algorithme qui demande un nombre de départ, et qui affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Début

```
Variables nombre, debut, fin, compteur en Entiers
Ecrire "Entrer un nombre:"
Lire nombre
debut <- nombre + 1
fin <- nombre + 11
Ecrire "Les dix nombres suivants " + nombre + " sont:"
Pour compteur <- debut à fin Pas 1 Faire
    Ecrire compteur
Suivant
```

Fin

10. Corrections

Exercice 5

Ecrire un algorithme qui demande un nombre de départ, et qui calcule la somme des entiers jusqu'à ce nombre (inclus). Par exemple, si l'on entre 5, le programme doit calculer $1 + 2 + 3 + 4 + 5 = 15$.

Début

Variables nombre, fin, compteur, somme en **Entiers**

Ecrire "Entrer un nombre:"

Lire nombre

fin \leftarrow nombre + 1

somme \leftarrow 0

Pour compteur \leftarrow 1 à fin Pas 1 **Faire**

somme \leftarrow somme + compteur

Suivant

Ecrire "La somme des entiers jusqu'à " + nombre + " est :" + somme

Fin

10. Corrections

Exercice 6

Ecrire un algorithme qui demande un nombre à l'utilisateur et qui ensuite écrit la table de multiplication de ce nombre de 0 à 10.

Début

Variables nombre, compteur en Entiers

Ecrire "Entrer un nombre:"

Lire nombre

Ecrire "La table de multiplication de " + nombre + " est:"

Pour compteur <- 0 à 11 Pas 1 Faire

 Ecrire nombre + "x" + compteur + "=" + (nombre*compteur)

Suivant

Fin

10. Corrections

Exercice 7

Ecrire un algorithme qui demande un nombre à l'utilisateur et qui calcule le factoriel de ce nombre. Par exemple si l'utilisateur saisit 8, le factoriel de 8 noté 8! vaut : $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 40320$. Pour 3 on aura $3! = 3 \times 2 \times 1 = 6$.

Début

Variables nombre, factoriel, compteur en Entiers

Ecrire "Entrer un nombre:"

Lire nombre

factoriel \leftarrow 1

TantQue (compteur < nombre) Ou (compteur = nombre) Faire

 factoriel \leftarrow (factoriel * compteur)

 compteur \leftarrow compteur + 1

FinTantQue

Ecrire "Le factoriel de " + nombre + " est :" + factoriel

Fin

10. Corrections

Exercice 8

Ecrire un programme qui demande successivement 20 nombres à l'utilisateur, et qui lui dise ensuite quel était le plus grand parmi ces 20 nombres.

Début

```
Variables max, nombre, compteur en Entiers
compteur ← 1
Ecrire "Entrer le nombre 1:"
Lire nombre
max ← nombre
Pour compteur ← 2 à 21 Pas 1 Faire
    Ecrire "Entrer le nombre " + compteur + ":"
    Lire nombre
    Si nombre > max Alors
        max ← nombre
    FinSi
Suivant
Ecrire "Le plus grand de ces nombre est :" + max
```

Fin

10. Corrections

Exercice 9

Reprendre l'algorithme précédent mais là il faut afficher la position à laquelle le nombre a été saisi.

Par exemple : le plus grand de ces nombre est 16 et c'est le nombre 2.

Début

```
Variables max, nombre, compteur, position en Entiers
```

```
compteur <- 1
```

```
Ecrire "Entrer le nombre 1:"
```

```
Lire nombre
```

```
max <- nombre
```

```
position <- 1
```

```
Pour compteur <- 2 à 21 Pas 1 Faire
```

```
    Ecrire "Entrer le nombre " + compteur + ":"
```

```
    Lire nombre
```

```
    Si nombre > max Alors
```

```
        max <- nombre
```

```
        position <- compteur
```

```
    FinSi
```

```
Suivant
```

```
Ecrire "Le plus grand de ces nombres est :" + max + " et c'est le nombre " + position
```

```
Fin
```

Citations

« Il est assez difficile de trouver une erreur dans son code quand on la cherche. C'est encore bien plus dur quand on est convaincu que le code est juste. » Steve McConnell

« Si le débogage est l'art d'enlever les bogues, alors la programmation doit être l'art de les créer. » Anonyme
