

# Support de cours

## Module : ASD1

### Chapitre 3: Les structures itératives

Réalisé par:  
Dr. Sakka Rouis Taoufik

1

## Introduction

- La notion d'itération est une des notions fondamentales de l'algorithmique.
- On l'utilise souvent quand on doit exécuter le même traitement un certain nombre de fois qui peut être connu à l'avance ou non.
- Dans ce dernier cas, l'arrêt de l'itération est déclenché par une condition sur l'état des variables dans le programme.

2

## La structure Pour ... Faire

### Syntaxe générale

**Pour** compteur **De** valeur\_initiale **A** valeur\_finale **Faire**  
Séquence d'instructions

**FinPour**

### Principe de fonctionnement

Le compteur (variable de contrôle) prend la valeur initiale au moment d'accès à la boucle puis, à chaque parcours, il passe **automatiquement** à la valeur suivante dans son domaine jusqu'à atteindre la valeur finale.

3

## La structure Pour ... Faire

### Exemple 1:

**Pour i de 1 à 5 Faire**

Ecrire ( $i*10$ )

**FinPour**

Cette boucle affiche  
respectivement les nombres  
10 20 30 40 50

### Exemple 2:

**Pour i de 5 à 1 Faire**

Ecrire ( $i*10$ )

**FinPour**

Cette boucle affiche  
respectivement les nombres  
50 40 30 20 10

4

## La structure Pour ... Faire

### Exercice 1:

Écrire un algorithme qui lit un entier positif  $n$  puis affiche tous ses diviseurs.

### Exercice 2:

Écrire un algorithme qui permet de déterminer le minimum et le maximum de  $n$  nombres saisies au clavier.

### Exercice 3:

Écrire un algorithme qui lit un entier positif  $n$  puis calcule et affiche son factoriel selon la formule :

$$n! = 1 * 2 * \dots * n.$$

5

## La structure Répéter... Jusqu'à

### Syntaxe générale

#### **Répéter**

Séquence d'instructions

#### **Jusqu'à condition**

### Principe de fonctionnement

La séquence d'instructions est exécutée une première fois, puis l'exécution se répète jusqu'à ce que la condition de sortie soit vérifiée.

➔ Une boucle « répéter » s'exécute toujours au moins une fois.

6

## La structure Répéter... Jusqu'à

- Contrairement à une boucle « pour », dans une boucle « répéter », l'initialisation et l'avancement du compteur doivent être gérés manuellement par le programmeur.

### Exemple

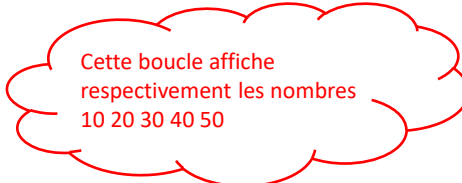
$i \leftarrow 1$

### Répéter

Ecrire ( $i * 10$ )

$i \leftarrow i + 1$

Jusqu'à ( $i > 5$ )



Cette boucle affiche  
respectivement les nombres  
10 20 30 40 50

7

## La structure Répéter... Jusqu'à

- Dans une boucle « répéter », il faut toujours s'assurer que la condition de sortie sera vérifiée après un nombre fini de parcours. Sinon, c'est une **boucle infinie** comme dans le cas suivant :

### Exemple

$c \leftarrow 'A'$

### Répéter

Ecrire ( $c$ )

Jusqu'à ( $c > 'Z'$ )

8

## La structure Répéter... Jusqu'à

### Exercice 4:

Réécrire l'algorithme diviseurs en remplaçant la boucle « pour » par une boucle « répéter ».

### Exercice 5:

Réécrire l'algorithme facto en remplaçant la boucle « pour » par une boucle « répéter ».

9

## La structure TantQue ... Faire

### Syntaxe générale

**TantQue** condition **Faire**

Séquence d'instructions

**FinTQ**

### Principe de fonctionnement

Le traitement est exécuté aussi longtemps que la condition est vérifiée. Si dès le début cette condition est fausse, le traitement ne sera exécuté aucune fois.

➔ Une boucle « tantque » peut s'exécuter 0, 1 ou n fois.

10

## La structure TantQue ... Faire

### Exemple

$i \leftarrow 1$

**TantQue** ( $i \leq 5$ ) **Faire**

Ecrire( $i * 10$ )

$i \leftarrow i + 1$

**FinTQ**

### **Exercice**

Réécrire les algorithmes diviseurs et facto en remplaçant les boucles « répéter » par des boucles « tantque »

11

## Exercices d'application

### Exercice 1

Écrire un algorithme permettant de :

- Lire un nombre fini de notes comprises entre 0 et 20
- Afficher la meilleure note, la mauvaise note et la moyenne de toutes les notes.

### Exercice 2

Calculer  $a^b$  avec  $a$  réel et  $b$  entier par multiplications successives.

### Exercice 3

Écrire un algorithme qui lit un entier positif et vérifie si ce nombre est premier ou non.

Remarque : un nombre premier n'est divisible que par 1 ou par lui-même.

12

## Exercices d'application

### Exercice 4

Écrire un algorithme qui lit deux entiers A et B puis calcule et affiche leur PGCD en utilisant la méthode suivante :

- Si  $A = B$  ;  $\text{PGCD}(A,B) = A$
- Si  $A > B$  ;  $\text{PGCD}(A,B) = \text{PGCD}(A-B,B)$
- Si  $B > A$  ;  $\text{PGCD}(A,B) = \text{PGCD}(A,B-A)$

Exemple :  $\text{PGCD}(18,45) = \text{PGCD}(18,27) = \text{PGCD}(18,9) = \text{PGCD}(9,9) = 9$

### Exercice 5 : nombres cubiques

Parmi tous les entiers supérieurs à 1, seuls 4 peuvent être représentés par la somme des cubes de leurs chiffres. Ainsi, par exemple :  $153 = 1^3 + 5^3 + 3^3$  est un nombre cubique.

Écrire un algorithme permettant de déterminer les 3 autres.

Note : les 4 nombres sont compris entre 150 et 410.

13

## Exercices d'application

### Exercice 6:

Écrire un algorithme qui permet de lire un entier positif et déterminer tous ses facteurs premiers.

#### **Exemples:**

$$30 = 2 * 3 * 5$$

$$36 = 2 * 2 * 3 * 3$$

$$99 = 3 * 3 * 11$$

### Exercice 7

Écrire un algorithme qui détermine tous les nombres premiers inférieurs à une valeur donnée.

14

## Exercices d'application

### **Exercice 8:**

Deux nombres entiers sont premiers entre eux s'ils n'ont pas d'autres diviseurs communs que 1.

7 et 13 n'ont que 1 comme diviseur commun donc 7 et 13 sont premiers entre eux.

12 et 32 ont plusieurs diviseurs communs : 1 ; 2 et 4 donc 12 et 32 ne sont pas premiers entre eux.

Écrire un algorithme qui saisit deux entiers N1 et N2, vérifie et affiche s'ils sont premiers entre eux ou non.

15

- <https://sourceforge.net/projects/asd1/files//>

16