FACULTE DES SCIENCES AIN CHOCK
UNIVERSITE HASSAN II DE CASABLANCA

Département: Mathématiques & Informatique

Séance 4:
Les boucles

Licence : Physique Chimie

Pr: Youssef Ouassit

Plan Séance 4

Les structures de contrôles :

1. **Les structures conditionnelles (de choix)**
   a. L’instruction conditionnelle simple
   b. L'instruction conditionnelle alternative
   c. L'instruction conditionnelle sélective

2. **Les instructions d’itération ou de répétition (les boucles)**
   a) La boucle Pour … Faire
   b) La boucle Tant Que

Structures répétitives (Itératives ou Boucles)

Définition :

- Une boucle est une structure de contrôle qui permet de répéter une séquence d'instructions.

Pourquoi les utiliser ?

- Pour automatiser des tâches répétitives.
- Réduire la duplication de code.
- Traiter des structures de données comme les listes, tuples, chaînes de caractères,...

Structures répétitives (Itératives ou Boucles)

Deux types principaux :

Structure déterministe :

- La boucle: Pour

Répétez un nombre de fois déterminé
Exemple : Répéter un mouvement 13 fois

Structure indéterministe :

- La boucle : TantQue

Répétez tant que une condition est vrai
Exemples :

- **Répéter un mouvement tant qu'il n y a pas d'insuffisance musculaire**
- **Préparer le crème pâtissière**

La boucle POUR

La boucle Pour

Motivation :

Afficher le mot Informatique 5 fois

Solution naïve (sans boucle) :

```
Algorithme Répéter
Début
        Ecrire("Informatique")
        Ecrire("Informatique")
        Ecrire("Informatique")
        Ecrire("Informatique")
        Ecrire("Informatique")
Fin
```

Inconvénients :

- Duplication de code
- Difficile à maintenir
- Cette approche est peu flexible, non réutilisable, et difficile à modifier.

La boucle Pour

Motivation :

Si on veut calculer la somme des 6 premiers nombres entiers :

Solution naïve (sans boucle) :

```
Algorithme Somme
Variables S : Entier
Début
    S ← 1 + 2 + 3 + 4 + 5 + 6
    Ecrire("La somme est ", S)
Fin
```

Inconvénients :

- Si on veut calculer la somme des 100 premiers entiers, ou plus, il faudrait écrire une longue série d'additions.
- Duplication de code
- Difficile à maintenir
- Cette approche est peu flexible, non réutilisable, et difficile à modifier.

La boucle Pour

Motivation : La table de multiplication d'un entier :

Solution naïve (sans boucle) :

```
Algorithme Table Multiplication
Variables N : Entier
Début
        Ecrire("Donner un entier : ")
        Lire(N)
        Ecrire(N, "x 1 = ", 1*N)
        Ecrire(N, "x 2 = ", 2*N)
        Ecrire(N, "x 3 = ", 3*N)
        Ecrire(N, "x 4 = ", 4*N)
        Ecrire(N, "x 5 = ", 5*N)
        Ecrire(N, "x 6 = ", 6*N)
        Ecrire(N, "x 7 = ", 7*N)
        Ecrire(N, "x 8 = ", 8*N)
        Ecrire(N, "x 9 = ", 9*N)
        Ecrire(N, "x 10 = ", 10*N)
Fin
```

Inconvénients :

- Il y a beaucoup de répétitions dans le code.
- Si vous voulez afficher la table de multiplication pour un plus grand nombre d'itérations (par exemple, jusqu'à 20), vous devrez tout réécrire ou modifier manuellement chaque ligne.

La boucle Pour

Définition :

En informatique, la boucle Pour est une structure de contrôle qui permet de répéter l'exécution d'une séquence d'instructions.

Dans cette forme de boucle, une variable prend des valeurs successives sur un intervalle. Cette forme est souvent utilisée pour exploiter les données d'une collection indexée.

La boucle Pour

Syntaxe de base:

```
Pour compteur ← VI à VF Faire
    SEQUENCE
FinPour
```

- Une « boucle for » a deux parties : une entête qui spécifie la manière de faire l'itération, et un corps qui est exécuté à chaque itération.
- Le compteur i varie de valeur_initiale (VI) à valeur_finale (VF).
- Les instructions à l'intérieur de la boucle sont exécutées pour chaque valeur du compteur.

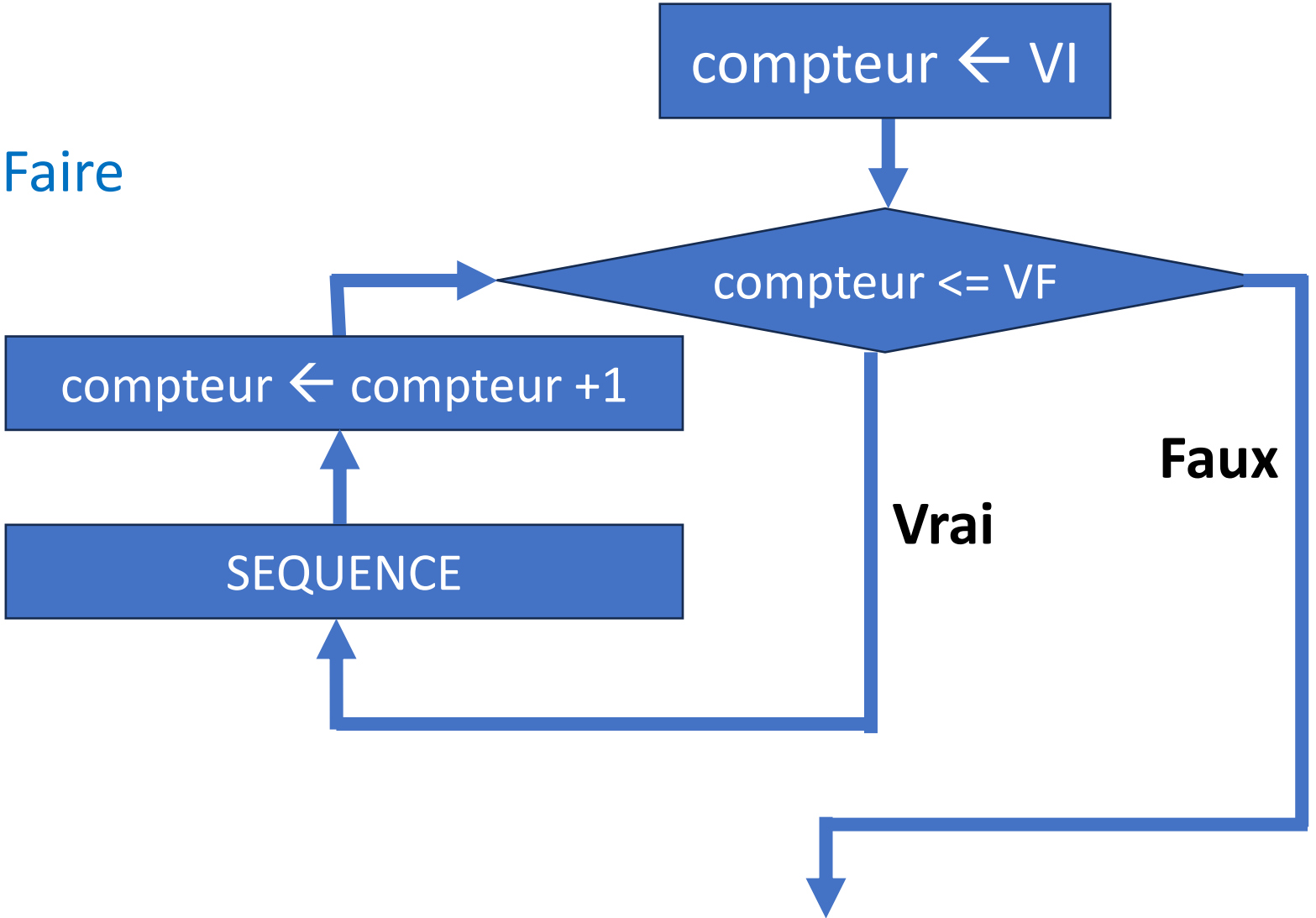
La boucle Pour

Organigramme:

Pour compteur ← VI à VF Faire

SEQUENCE

FinPour



Structures répétitives (Les boucles)

La boucle Pour

Exemple 1: Répéter une séquence 10 fois

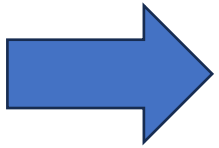
```
Pour i ← 1 à 10 Faire
        SEQUENCE
FinPour
```

Fonctionnement :
Pour chaque valeur i de l'intervalle [1, 10] exécuter la SEQUENCE

La boucle Pour

Exemple 2:

```
Pour i ← 1 à 5 Faire
        Ecrire("Informatique")
FinPour
```



5 répétitions :

Informatique
Informatique
Informatique
Informatique
Informatique

La boucle Pour

Exemple 2: Afficher le mot informatique 500 fois

Il suffit de changer la valeur finale 5 par 500:

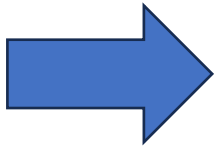
```
Pour i ← 1 à 500 Faire
        Ecrire("Informatique")
FinPour
```

La boucle Pour

Exemple 3:

Il est possible de choisir n'importe quel valeur initiale:

```
Pour i ← 3 à 7 Faire
        Ecrire("Bonjour")
FinPour
```

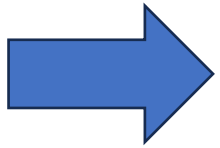


```
Bonjour
Bonjour
Bonjour
Bonjour
Bonjour
```

La boucle Pour

Exemple 4 : Affiches la suite des nombres de 1 à 10

```
Pour i ← 1 à 10 Faire
        Ecrire(i)
FinPour
```



```
1
2
3
4
5
6
7
8
9
10
```

La boucle Pour

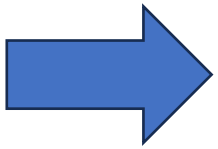
Exemple 5 : Un algorithme qui affiche la suite des nombres de 1 à un entier N entré par l'utilisateur

```
Algorithme Nombres
Variables i, N : Entier
Début
        Ecrire(" Donner le nombre  : ")
        Lire(N)
        Pour  i ← 1 à N Faire
                Ecrire(i)
        FinPour
Fin
```

La boucle Pour

Exemple 6 : Un algorithme qui affiche la table de multiplication d'un entier N.

```
Algorithme Table Multiplication
Variables i, N : Entier
Début
        Ecrire(" Donner le nombre  : ")
        Lire(N)
        Pour  i ← 1 à 10 Faire
                Ecrire(N, "x", i , "=", N*i)
        FinPour
Fin
```



```
Donner un nombre: 5
5x1=5
5x2=10
5x3=15
5x4=20
5x5=25
5x6=30
5x7=35
5x8=40
5x9=45
5x=10=50
```

La boucle Pour

Exemple 7 : Un algorithme qui affiche les nombres multiple de 3 et de 7 entre 1 à un entier N entré par l'utilisateur

```
Algorithme Nombres
Variables i, N : Entier
Début
        Ecrire(" Donner le nombre  : ")
        Lire(N)
        Pour  i ← 1 à N Faire
                Si i mod 3=0 OU i mod 7=0 Alors
                        Ecrire(i)
                FinSi
        FinPour
Fin
```

La boucle Pour

Exemple 8 : Un algorithme qui calcule et affiche la somme des nombres de 1 à un entier N entré par l'utilisateur.

```
Algorithme Somme
Variables i, N, S : Entier
Début
        Ecrire(" Donner le nombre  : ")
        Lire(N)
        S ← 0
        Pour  i ← 1 à N Faire
                S ← S + i
        FinPour
        Ecrire("La somme est ", S)
Fin
```

La boucle Pour

Exemple 9 : Un algorithme qui calcule et affiche la somme des nombres de pairs dans l'intervalle 1 et N entré par l'utilisateur.

```
Algorithme Somme
Variables i, N, S : Entier
Début
        Ecrire(" Donner le nombre  : ")
        Lire(N)
        S ← 0
        Pour  i ← 1 à N Faire
                Si i mod 2 = 0 Alors
                        S ← S + i
                FinSi
        FinPour
        Ecrire("La somme est ", S)
Fin
```

Structures répétitives (Les boucles)

La boucle Pour

Syntaxe complet:

```
Pour compteur ← VI à VF  pas VP  Faire
    SEQUENCE
FinPour
```

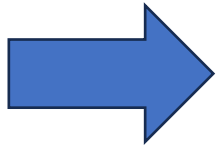
- VI : Valeur initiale
- VF : Valeur finale
- VP : Valeur du pas (Par défaut c'est 1)

La pas c'est la valeur avec laquelle le compteur de la boucle avance après chaque itération.

La boucle Pour

Exemple 10 : Compter avec un pas égal à 2

```
Algorithme Tester Pas
Variables i, N : Entier
Début
        Pour  i ← 1 à 10 pas 2    Faire
                Ecrire(i)
        FinPour
Fin
```



```
1
3
5
7
9
```

La boucle TantQue

Motivation :

Vous voulez écrire un programme qui demande à un utilisateur d'entrer un mot de passe. Tant que le mot de passe n'est pas correct, il doit redemander le mot de passe. Le nombre de tentatives est inconnu à l'avance.

Combien de fois l'utilisateur peut se tromper ?

Structures répétitives (Les boucles)

La boucle TantQue

Définition :

La boucle TantQue est une structure de contrôle en programmation qui permet d'exécuter un bloc de code répétitif tant qu'une condition donnée est vraie. Cela signifie que le bloc de code à l'intérieur de la boucle TantQue est exécuté de manière répétée tant que la condition spécifiée reste vraie. Une fois que la condition devient fausse, l'exécution de la boucle s'arrête et le programme passe à l'instruction suivante après la boucle.

Structures répétitives (Les boucles)

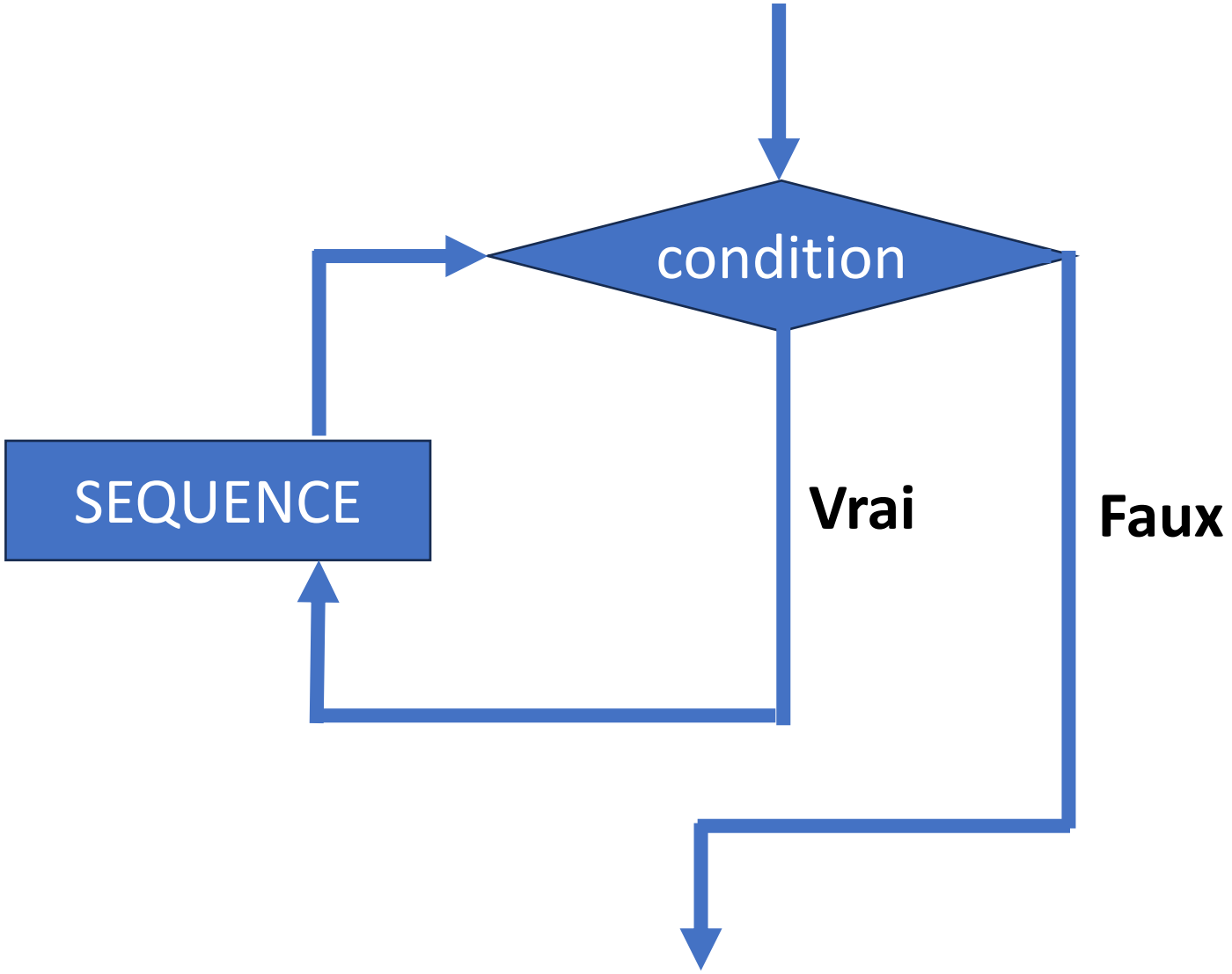
La boucle TantQue

Syntaxe :

TantQue condition Faire

SEQUENCE

FinTanque



La boucle TantQue

Exemple 1 : Mot de passe

```
Algorithme Indemnités
Variables pwd : Entier
Début
        mot_de_passe_correct ← "abc123"
        Ecrire(" Votre mot de passe:")
        Lire(pwd)
        TantQue pwd != mot_de_passe_correct Faire
                Ecrire("Mot de passe incorrect, réessayez.")
                Lire(pwd)
         FinTantQue
         Ecrire("Accès autorisé.")
Fin
```

La boucle TantQue vérifie la validité du mot de passe.

La boucle TantQue

Exemple 2 :

Calculer les indemnités sociales à verser sachant que pour chaque enfant l'employer bénéfice d'une somme de 300 DH.

```
Algorithme Indemnités
Variables N : Entier
Début
        Ecrire(" Donner le nombre de vos enfants : ")
        Lire(N)
        S = N*300
        Ecrire("Le montant à verser est :", S)
Fin
```

La boucle TantQue

Exemple 2 :

Nous avons ajouté un contrôle de saisie

```
Algorithme Indemnités
Variables N : Entier
Début
        Ecrire(" Donner le nombre de vos enfants : ")
        Lire(N)
        TantQue N < 0 Faire
                Ecrire("Erreur, donnez une valeur valide : ")
                Lire(N)
        FinTantQue
        S = N*300
        Ecrire("Le montant à verser est :", S)
Fin
```

La boucle TantQue

Exemple 3 :

Écrire un algorithme qui calcule la somme des valeurs saisies par l'utilisateur. La saisie se termine lorsque l'utilisateur entre la valeur 0.

La boucle TantQue

Exemple 3 :

```
Algorithme Somme Plusieurs Valeurs
Variables N, S : Entier
Début
    S ← 0
    N ← -1
    TantQue N!=0  Faire
        Ecrire("Donner un autre nombre: ")
        Lire(N)
        S ← S + N
    FinTantQue
    Ecrire("La somme est ", S)
Fin
```

Equivalence entre les boucles

Une boucle TantQue équivalente à la boucle Pour:

```
Pour i ← 1 à 5 Faire
        Ecrire("Bonjour")
FinPour
```

```
i ← 1
TantQue i <= 5 Faire
        Ecrire("Bonjour")
        i ← i + 1
FinTantQue
```