Chapitre 3 : Structures de contrôles répétitives

Les structures itératives

➢ **Les structures itératives (les boucles)** permettent de répéter un bloc d'instructions plusieurs fois.

Pour	On connait d'avance le nombre de répétions: On répète les instructions en faisant évoluer un compteur entre une valeur initiale et une valeur finale
Tant Que	On ne connait pas d'avance le nombre de répétions: On répète des instructions tant qu'une certaine condition est réalisée
Répéter jusqu'à	On ne connait pas d'avance le nombre de répétions: On répète des instructions jusqu'à ce qu'une certaine condition soit réalisée

La boucle : Pour

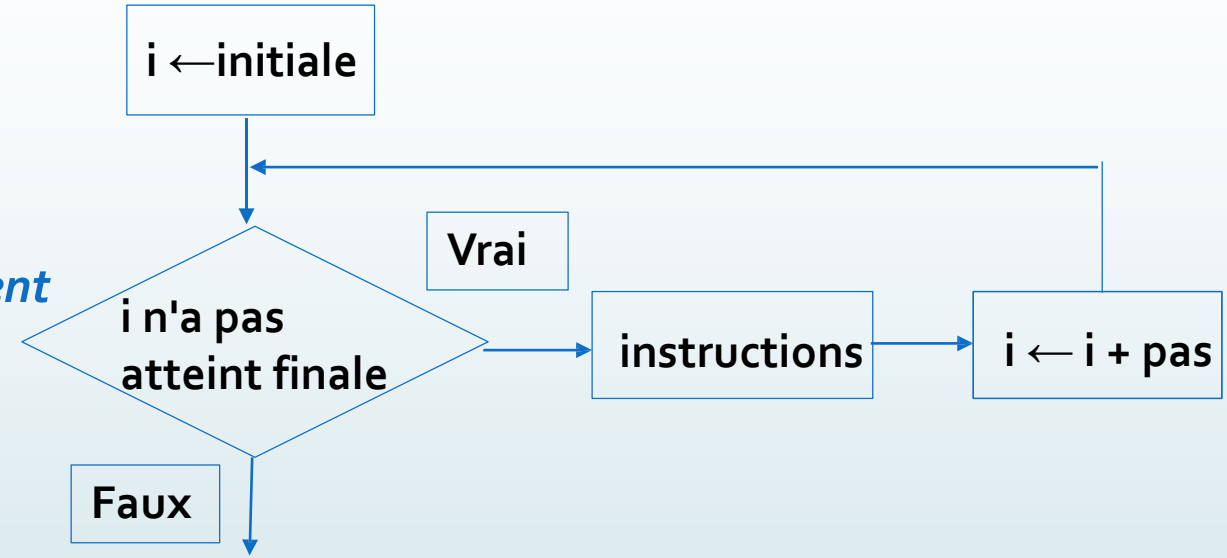
- Utilisée lorsqu'on connait d'avance le nombre de répétitions

- **Syntaxe**:

POUR *compteur* = *val_initiale* **A** *val_finale* **PAS DE** *incrément*

Instructions à répéter

FIN POUR



- **compteur** : variable permettant de compter le nombre d'itérations

- **val_initiale** et **val_finale** sont respectivement la valeur initiale et finale prises par le compteur. Elles peuvent être des valeurs, des variables, ou des expressions de même type que le compteur

- **incrément** est la valeur d'augmentation progressive du compteur. Le pas est un entier qui peut être positif ou négatif. La valeur par défaut du pas est de 1

La boucle : Pour

1) La valeur initiale est affectée à la variable compteur

2) On compare **la valeur du compteur** et **la valeur de finale** :

a) Si la valeur du **compteur est >** à la valeur finale dans le cas d'un pas positif (ou si compteur est < à finale pour un pas négatif), on sort de la boucle et on continue avec l'instruction qui suit FinPour

b) Si **compteur est <=** à finale dans le cas d'un pas positif (ou si compteur est >= à finale pour un pas négatif), instructions seront exécutées

i. Ensuite, la valeur du compteur est incrémentée de la valeur du pas si pas est positif (ou décrémenté si pas est négatif)

ii. On recommence l'étape 2 : La comparaison entre compteur et finale est de nouveau effectuée, et ainsi de suite …

La boucle : Pour

- **Exemple 1: Ecrire un algorithme qui calcule la moyenne des notes de 200 étudiants**

```
VARIABLES mat, phy, moyenne : REELS
VARIABLE i : ENTIER
        Début
                POUR i = 1 A 200 PAS DE 1
                        ECRIRE ("Entrez la note de math :")
                        LIRE (mat)
                        ECRIRE ("Entrez la note de la physique:")
                        LIRE (phy)
                        moyenne ← (mat + phy) / 2
                        ECRIRE ("La moyenne est : ", moyenne)
                FIN POUR
        Fin
```

La boucle : Pour

- **Exemple 2: Ecrire un algorithme qui calcule la factorielle d'un nombre saisi par l'utilisateur**

```
Algorithme Factorielle
Variables N, i, Fact: Entier
Début
        Ecrire (" Saisir un nombre: ")
        Lire (N)
        Fact ← 1
        Pour i=1 A N
                Fact ← Fact *i
        FinPour
        Ecrire (" La factorielle de ", N, " est : ", Fact)
Fin
```

La boucle : Pour

- Il faut éviter de **modifier la valeur du compteur** à l'intérieur de la boucle. En effet, une telle action peut :
  - Perturber le nombre d'itérations prévu par la boucle Pour
  - Rendre difficile la lecture de l'algorithme
  - Présenter le risque d'aboutir à une boucle infinie
- **Exemple**:

```
Pour i=1 A 5
        i ← i – 1
        Ecrire (" i= ", i)
FinPour
```

La boucle Pour Imbriquée

- **Une structure itérative** peut contenir parmi ses instructions **une autre structure itérative**. Dans ce cas, on aboutit à **des boucles imbriquées**

- **Exemple**

```
Pour i=1 A 5
        Pour j=1 A i
            Ecrire (j, " ")
        FinPour
        Ecrire ("\n")
FinPour
```

Exécution

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Exercices

Q1. Simuler l'exécution de l'algorithme ci-dessous :

```
Algorithme Pour1
Variables nb,i : entier
Début
        nb ← 10
        pour i de 1 à 4
            écrire(nb)
            nb ← nb + 5
        finPour
        écrire("nombre = ", nb)
Fin
```

- **Q2. Ecrire un algorithme qui demande un entier de départ, et qui ensuite affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27**
- **Q3. Ecrire un algorithme qui demande un entier, et qui calcule la somme des entiers jusqu'à ce nombre. Par exemple, si l'on entre 5, le programme doit calculer : 1 + 2 + 3 + 4 + 5 = 15**