ANNÉE SCOLAIRE 2023/2024

COURS D'ALGORITHMIQUE

Pape Abdoulaye **BARRO**
Docteur en Informatique et Télécommunications
Spécialiste en Télémétrie & Systèmes Intelligents

STRUCTURES ITÉRATIVES

STRUCTURES ITÉRATIVES
DÉFINITIONS

- Une itération consiste en la répétition d'un bloc d'instructions jusqu'à ce qu'une certaine condition soit vérifiée.
- Il existe 2 sortes d'itérations:
  - Le nombre de répétitions est connu dés le départ

    Exemple : Calcul de 1+2+3+…+n
  - Le nombre de répétitions est méconnu

    Exemple : Recherche de PGCD de deux entiers

STRUCTURES ITÉRATIVES
DÉFINITIONS

Considérons le problème suivant:

On se propose d'afficher tous les entiers naturels strictement plus petits que 100. Il est évident que, même s'il est possible de résoudre ce problème en écrivant toutes les instructions comme suit :

Ecrire(''1'')

Ecrire(''2'')

..

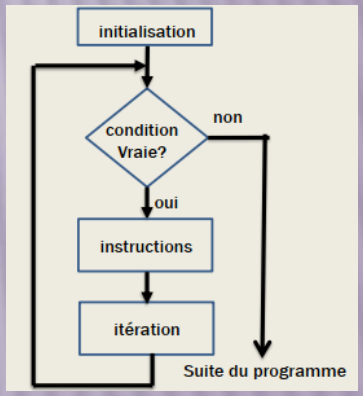
Ecrire(''99'')

Ceci reste une solution non désirée, d'où la nécessité de trouver une alternative. C'est là que la notion de boucle va prendre toute son importance.

ALGORITHMIQUE
STRUCTURES ITÉRATIVES: LA STRUCTURE POUR

La structure POUR

On répète les instructions en faisant évoluer un compteur entre une valeur initiale et une valeur finale. Le nombre d'itérations est connu avant le début de la boucle.



- La syntaxe est:

```
Pour i allant de MIN à MAX par pas de PAS faire
        {instructions}
FinPour
```

- Remarque:
  - Le compteur est une variable de type entier (ou caractère). Elle doit être déclarée.
  - Lorsque le pas d'itération vaut 1, la syntaxe peut être simplifiée comme suit:

```
Pour i allant de MIN à MAX faire
        {instructions}
FinPour
```

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES: LA STRUCTURE POUR

- **Exemple**:

```
Algorithme  Factorielle
{Cet algorithme permet de calculer la factorielle d'un entier naturel}
Variable n, i, resultat : entier
Début
  Ecrire("Entrez un entier naturel")
  Lire(n)
  resultat ← 1
  Pour i allant de 2 à n faire
     resultat ← resultat *i
  FinPour
  Ecrire(n, "! =", resultat )
Fin
```

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES: LA STRUCTURE POUR

- **Exemple**:

  Écrire un programme qui permet de calculer la somme S=1+2+3+4+….+ N.
  où N est saisie au clavier par l'utilisateur.

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES: LA STRUCTURE POUR

- **Exemple**:

  Écrire un programme qui permet de calculer la somme S=1+2+3+4+….+ N.
  où N est saisie au clavier par l'utilisateur.

- **Solution**:

```
Algorithme Somme_des_N_entiers
Variable i, S, N: entier
Debut
        S ← 0
        Ecrire("Donner un entier")
        Lire (N)
        Pour i allant de 1 à N faire
            S ← S + i
        FinPour
        Ecrire("La somme est:", S)
Fin
```

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE FAIRE TANTQUE

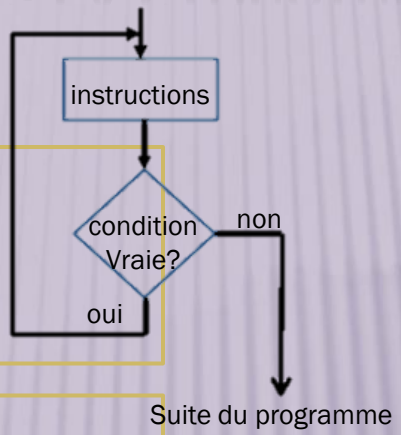
La structure Faire TantQue

- **La syntaxe est :**

Faire

{instructions}

TantQue (condition_de_reprise)

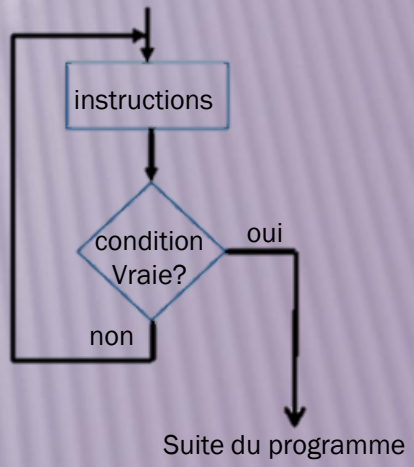


- Ou

Répéter

{instructions}

Jusqu'à(condition_de_sortie)



- Remarque :
  - Les instructions sont exécutées au moins une fois et peuvent être répétées jusqu'à ce que la condition soit fausse (tant qu'elle est vraie): le cas de Faire TantQue.
  - Les instructions sont exécutées au moins une fois et peuvent être répétées jusqu'à ce que la condition soit vraie (tant qu'elle est fausse): le cas de répéter jusqu'à.
  - Préférez la première syntaxe, elle est plus proche de l'implémentation fournie par la plupart des langages de programmation.

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE FAIRE TANTQUE

- **Exemple: avec Faire … TantQue**

```
Algorithme Factorielle
{Cet algorithme permet de calculer la factorielle d'un entier naturel}
Variable n, i, result : entier
Debut
        Ecrire(''Entrez un entier naturel'')
        Lire(n)
        result ← 1
        i← 1
        Faire
                result ← result *i
                i← i+1
        TantQue(i<=n)
        Ecrire(n, "! =",result)
Fin
```

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE FAIRE TANTQUE

- **Exemple: Avec Répéter … Jusqu'à**

```
Algorithme Factorielle
{Cet algorithme permet de calculer la factorielle d'un entier naturel}
Variable n, i, result : entier
Debut
        Ecrire("Entrez un entier naturel")
        Lire(n)
        result ← 1
        i← 1
        Répéter
                result ← result *i
                i← i+1
        Jusqu'à(i>n)
        Ecrire(n,"! =",result)
 Fin
```

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE FAIRE TANTQUE

- **Exemple**:

  Écrire un algorithme qui affiche la table de multiplication de 8.

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE FAIRE TANTQUE

- **Exemple**:

  Écrire un algorithme qui affiche la table de multiplication de 8.

- **Solution**:

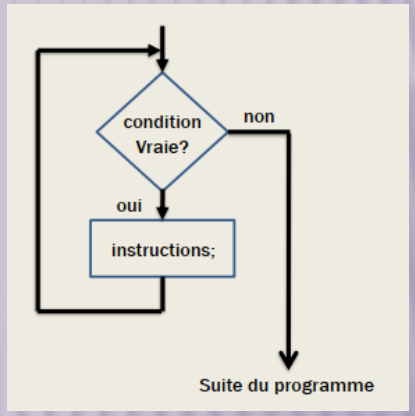
```
Algorithme Table_Multiplication_de_8
Variable i:entier
Début
        i ← 0
        Répéter
                Ecrire("8*",i,"=",i*8)
                i ← i+1
        Jusqu'à ( i > 10 )
Fin
```

RAPPEL ALGORITHMIQUE

STRUCTURES ITÉRATIVES: LA STRUCTURE TANTQUE

- La syntaxe est :

```
TantQue (condition_d_entree) faire
        {instructions}
FinTantQue
```



- **Remarques** :
  - La condition d'entrée doit être définie au préalable sinon, en implémentant votre algorithme, vous risquez d'avoir des comportements étranges.
  - Si la condition est vraie, on exécute les instructions (corps de la boucle) puis, on retourne tester la condition. Si elle est encore vraie, on répète l'exécution, …
  - Si la condition est fausse, on sort de la boucle et on exécute l'instruction qui est après FinTantQue.
  - Il est possible que les instructions à répéter ne soient jamais exécutées.

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES

Exemple:

```
Algorithme Factorielle
{Cet algorithme permet de calculer la factorielle d'un entier naturel}
Variable n, i, result : entier
Debut
        Ecrire("Entrez un entier naturel")
        Lire(n)
        result ← 1
        i← 1
        TantQue(i<=n) faire
                result ← result *i
                i← i+1
        FinTantQue
        Ecrire(n,"! =",result)
Fin
```

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES

- **Exemple**:

  Écrire un programme permettant de calculer la somme S=1+2+3+...+ N, où N saisie par l'utilisateur.

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES

- **Exemple**:

  Écrire un programme permettant de calculer la somme S=1+2+3+...+ N, où N saisi par l'utilisateur.

- **Solution**:

```
Algorithme Somme_de_1_jusqu'au_N
Variables i, S, N: entiers
Debut
        i ← 1
        S ← 0
        Ecrire("Donner un entier:")
        Lire (N)
        TantQue ( i <= N  ) faire
                S ← S + i
                i ←  i + 1
        FinTantQue
        Ecrire("La somme de 1 à N est:",S)
fin
```

RAPPEL ALGORITHMIQUE
STRUCTURES ITÉRATIVES

QUELLE BOUCLE CHOISIR ?

Chaque boucle a un contexte dans lequel son utilisation est plus adéquate bien qu'il soit possible d'utiliser l'une comme l'autre dans certains contextes.

- **Pour** : Lorsque le nombre d'itérations est connu;
- **Faire … tantQue** : Lorsque nous sommes certains d'exécuter le bloc d'instruction au moins une fois;
- **TantQue … faire**: Lorsque le bloc d'instruction peut ne pas du tout être exécuté.

RAPPEL ALGORITHMIQUE
CAS PRATIQUES N°3

- **Application 16 :**

Écrire un algorithme permettant de calculer la somme des nombres impairs de 1 à n.

- **Application 17:**

Écrire un algorithme permettant de calculer la moyenne des n premiers entiers. n étant strictement positif.

- **Application 18:**

Écrire un algorithme permettant de lire 20 nombres au clavier et d'afficher le carré des nombres pairs uniquement. Attention, on ne mémorisera pas les 20 valeurs saisies.

- **Application 19:**

Écrire un algorithme qui demande un nombre à l'utilisateur, puis vérifie et affiche que les nombres paires.

Le programme s'arrête lorsque l'utilisateur donne -1.

- **Application 20:**

Écrire l'algorithme permettant de lire puis d'afficher une valeur comprise entre 1 et 31; on recommencera la saisie jusqu'à ce que la valeur soit bien dans les bornes imposées.

Affaires à suivre