المدرسة العليا للتربية والتكوين - اكادير
ⵜⵜⵉⵛⵏ ⵜⴰⵡⴰⵔⵉⵜ ⴷ ⵜⵅⵛⵛ ⴰ ⵓⵎⵛⵓⵔⵅ - ⴰⵅⴰⴷⵉⵔ
ECOLE SUPÉRIEURE DE L'ÉDUCATION ET DE LA FORMATION - AGADIR

ALGORITHMIQUE
INSTRUCTIONS ITÉRATIVES

Pr. Hasna ABIOUI
E-mail: h.abioui@uiz.ac.ma
Année Universitaire 2019/2020

LES BOUCLES
INSTRUCTIONS ITÉRATIVES

- Les boucles servent à répéter l'exécution d'un groupe d'instructions un certain nombre de fois
- On distingue trois sortes de boucles en langages de programmation:
    - Les boucles **Pour** (ou avec compteur) : on y répète des instructions en faisant évoluer un compteur (variable particulière) entre une valeur initiale et une valeur finale
    - Les boucles **TantQue** : on y répète des instructions tant qu'une certaine condition est réalisée
    - Les boucles **Répéter… Jusqu'à** : on y répète des instructions jusqu'à ce qu'une certaine condition soit réalisée.

LA BOUCLE POUR
INSTRUCTIONS ITÉRATIVES

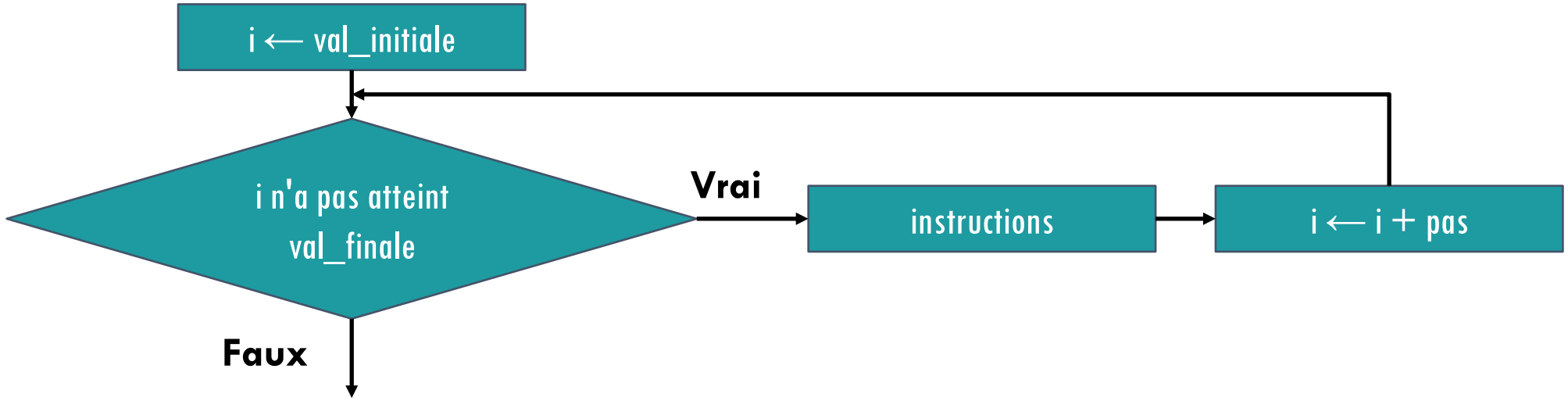
- **Syntaxe:**

Pour compteur **allant de** val_initiale **à** val_finale *[par pas val_du_pas]* **faire**

Instruction_1

Instruction_2

FinPour



LA BOUCLE POUR
INSTRUCTIONS ITÉRATIVES

Remarque :

- Le nombre d'itérations dans une boucle Pour est connu avant le début de la boucle
- **Compteur** est une variable de type entier (ou caractère). Elle doit être déclarée au préalable
- **Pas** est un entier qui peut être positif ou négatif. Pas peut ne pas être mentionné, car par défaut sa valeur est égal à 1. Dans ce cas, le nombre d'itérations est égal à val_finale - val_initiale + 1
- **val_initiale** et **val_finale** peuvent être des valeurs, des variables définies avant le début de la boucle ou des expressions de même type que le compteur

PRINCIPE DE LA BOUCLE POUR
INSTRUCTIONS ITÉRATIVES

- La valeur initiale est affectée à la variable compteur
- On compare la valeur du compteur et la valeur finale :
  - Si la **valeur du compteur** est > à la **valeur finale** dans le cas d'un pas positif (ou si compteur est < à val_finale pour un pas négatif), **on sort de la boucle** et on continue avec l'instruction qui suit **FinPour**
  - Si le **compteur** est <= à **val_finale** dans le cas d'un pas positif (ou si compteur est >= à val_finale pour un pas négatif), les instructions seront exécutées.
    - Ensuite la valeur du compteur est **incrémentée par** la valeur du pas si pas positif (ou décrémenté si pas est négatif)
    - On recommence l'étape 2 : la comparaison entre le **compteur** et **val_finale** est de nouveau effectuée, et ainsi de suite…

LA BOUCLE POUR : EXEMPLE (1)
INSTRUCTIONS ITÉRATIVES

Calcul de x à la puissance n (x^n) où x est un réel non nul et n est un entier positif ou nul

```
Algorithme Calcul_puissance
    Variables x, p : réel
        n, i : entier
    Début
        Ecrire("Entrez respectivement les valeurs de x et n : ")
        Lire(x, n)
        p ← 1
        Pour i allant de 1 à n faire
            p ← p * x
        FinPour
        Ecrire(x, " à la puissance ", n, " est égal à ", p)
    Fin
```

LA BOUCLE POUR : EXEMPLE (1) – PAS NÉGATIF
INSTRUCTIONS ITÉRATIVES

Même question, mais cette fois-ci en utilisant un pas négatif

```
Algorithme Calcul_puissance
Variables x, p : réel
        n, i : entier
Début
        Ecrire("Entrez respectivement les valeurs de x et n : ")
        Lire(x, n)
        p ← 1
        Pour i allant de n à 1 par pas -1 faire
                p ← p * x
        FinPour
        Ecrire(x, " à la puissance ", n, " est égal à ", p)
Fin
```

LA BOUCLE POUR : REMARQUES
INSTRUCTIONS ITÉRATIVES

- Il faut éviter de modifier la valeur du compteur (et de la valeur finale) à l'intérieur de la boucle
- En effet, une telle action :
  - Perturbe le nombre d'itérations prévu par la boucle Pour
  - Rend difficile la lecture de l'algorithme
  - Présente le risque d'aboutir à une boucle infinie

Exemple :

```
Pour i allant de 1 à 5 faire
    i ← i – 1
    Ecrire("i = ", i)
FinPour
```

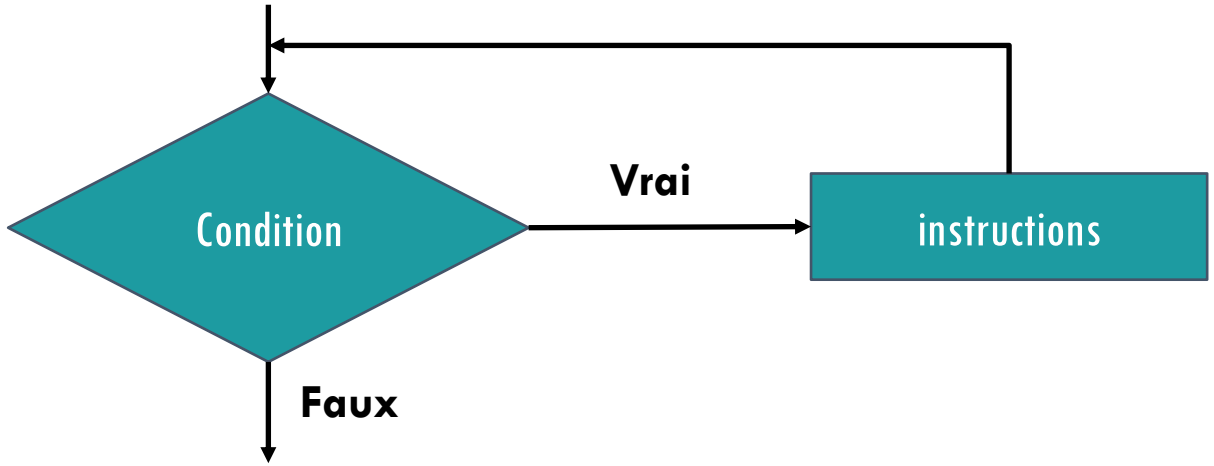
LA BOUCLE TANTQUE
INSTRUCTIONS ITÉRATIVES

- **Syntaxe :**

TantQue (**condition**)

instructions

FinTantQue



- La condition (dite condition de contrôle de la boucle) est évaluée avant chaque itération
- Si la condition est vraie, on exécute les instructions (corps de la boucle), puis on re-teste la condition à nouveau. Si elle est encore vraie, on répète l'exécution et ainsi de suite.

LA BOUCLE TANTQUE : REMARQUES
INSTRUCTIONS ITÉRATIVES

- Le nombre d'itérations dans une boucle **TantQue** n'est pas connu au moment d'entrer dans la boucle. Il dépend de la valeur de la condition

- Une des instructions du corps de la boucle doit absolument changer la valeur de la condition de vrai à faux (après un certain nombre d'itérations), sinon le **programme tourne indéfiniment.**

Exemple de boucle infinie :

```
i ← 2
TantQue (i > 0)
    i ← i+1
FinTantQue
```

LA BOUCLE TANTQUE : EXEMPLE (1)
INSTRUCTIONS ITÉRATIVES

Un algorithme qui calcule et affiche la somme S= 1+2+3+…+100

```
Algorithme Somme_100
    Variables som, i : entier
    Début
        i ← 1
        som ← 0
        TantQue (i<=100)
            som ← som+i
            i ← i+1
        FinTantQue
        Ecrire("La somme de 1 à 100 est ", som)
    Fin
```

LA BOUCLE TANTQUE : EXEMPLE (2)
INSTRUCTIONS ITÉRATIVES

Nous souhaitons à chaque fois, contrôler la saisie de manière à accepter uniquement la saisie d'une lettre majuscule, sinon on demande à l'utilisateur de rentrer à nouveau dans le cas inverse

```
Algorithme Lette_majuscule
    Variable c : caractère
    Début
        Ecrire("Entrez une lettre majuscule : ")
        Lire(c)
        TantQue (c < 'A' OU c > 'Z')
            Ecrire("Saisie erronée. Recommencez")
            Lire(c)
        FinTantQue
        Ecrire("Saisie valable")
    Fin
```

LA BOUCLE TANTQUE : EXEMPLE (3)
INSTRUCTIONS ITÉRATIVES

Ecrire un algorithme qui détermine le premier nombre entier N tel que la somme de 1à N dépasse strictement 100

```
Algorithme Somme
    Variables som, i : entier
    Début
        i ← 0
        som ← 0
        TantQue (som <=100)
            i ← i+1
            som ← som+i
        FinTantQue
        Ecrire("La valeur cherchée est N= ", i)
    Fin
```

LIEN ENTRE POUR ET TANTQUE
INSTRUCTIONS ITÉRATIVES

- La boucle **Pour** est un cas particulier de **TantQue** (cas où le nombre d'itérations est connu et fixé). Tout ce qu'on peut écrire avec **Pour** peut être remplacé avec **TantQue** (**la réciproque est fausse**)

Pour compteur **allant de** initiale **à** finale **par pas** de valeur du pas **faire**
    instructions
FinPour

Peut être remplacé par :

compteur ← initiale
TantQue (compteur <= finale)
    instructions
    compteur ← compteur + pas
FinTantQue

LIEN ENTRE POUR ET TANTQUE : EXEMPLE
INSTRUCTIONS ITÉRATIVES

- Calcul de x à la puissance n où x est un réel non nul et n est un entier positif ou nul (version avec **TantQue**)

```
Algorithme Calcul_puissance
Variables      x, p : réel
               n, i : entier
Début
    Ecrire("Entrez respectivement les valeurs de x et n : ")
    Lire(x, n)
    p ← 1
    i ← 1
    TantQue (i<=n)
        p ← p * x
        i ← i+1
    FinTantQue
    Ecrire(x, " à la puissance ", n, " est égal à ", p)
Fin
```

BOUCLE IMBRIQUÉE POUR
INSTRUCTIONS ITÉRATIVES

- Les instructions d'une boucle peuvent être des instructions itératives.
- Dans ce cas, on aboutit à des boucles imbriquées

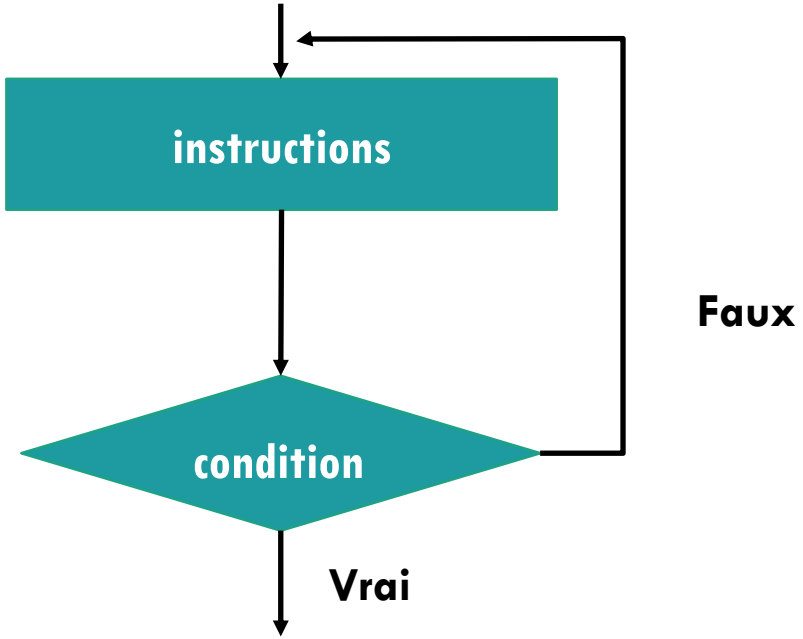
Exemple:

```
Pour i allant de 1 à 5 faire
    Pour j allant de 1 à i faire
        Ecrire("0")
    FinPour
    Ecrire("1")
FinPour
```

Résultat d'exécution

01
001
0001
00001
000001

LA BOUCLE RÉPÉTER …JUSQU'À
INSTRUCTIONS ITÉRATIVES



- **Syntaxe:**

  Répéter

  instructions

  Jusqu'à (**condition**)

- **Condition** est évaluée après chaque itération
- Les instructions entre **Répéter** et **Jusqu'à** sont exécutées **au moins une fois** et leur exécution est répétée jusqu'à ce que condition soit vrai (tant qu'elle est fausse)

LA BOUCLE RÉPÉTER …JUSQU'À : EXEMPLE
INSTRUCTIONS ITÉRATIVES

Un algorithme qui détermine le premier nombre entier N tel que la somme de 1 à N dépasse strictement 100

```
Algorithme nombre_somme
    Variables som, i : entier
    Début
        som ← 0
        i ← 0
        Répéter
            i ← i + 1
            som ← som + i
        Jusqu'à (som > 100)
        Ecrire("La valeur cherchée est N=", i)
    Fin
```

CHOIX D'UN TYPE DE BOUCLE
INSTRUCTIONS ITÉRATIVES

- Si on peut déterminer le nombre d'itérations avant l'exécution de la boucle, il est plus naturel d'utiliser la boucle **Pour**
- S'il n'est pas possible de connaître le nombre d'itérations avant l'exécution de la boucle, on fera appel à l'une des boucles **TantQue** ou **Répéter... Jusqu'à**
- Pour le choix entre **TantQue** et **Répéter... Jusqu'à** :
    - Si on doit tester la condition de contrôle avant de commencer les instructions de la boucle, on utilisera *TantQue*
    - Si la valeur de la condition de contrôle **dépend** d'une première exécution des instructions de la boucle, on utilisera *Répéter... Jusqu'à*

CHOIX D'UN TYPE DE BOUCLE
INSTRUCTIONS ITÉRATIVES