Informatique Générale
Cours 3

- Boucles Tant Que
- Boucles Pour

La Boucle TantQue

Les boucles, à quoi ca sert ?

- Exemple du contrôle de saisie
  - Dans le cas d'une saisie au clavier (une lecture), où par exemple, le programme pose une question à laquelle l'utilisateur doit répondre par O (Oui) ou N (Non), tôt ou tard, l'utilisateur, facétieux ou maladroit, risque de taper autre chose que la réponse attendue.
  - Dès lors, le programme peut planter soit par une erreur d'exécution (parce que le type de réponse ne correspond pas au type de la variable attendu) soit par une erreur fonctionnelle (il se déroule normalement jusqu'au bout, mais en produisant des résultats fantaisistes).
  - On met en place ce qu'on appelle un **contrôle de saisie**, afin de vérifier que les données entrées au clavier correspondent bien à celles attendues par l'algorithme.

Contrôle de saisie avec un SI

Exemple n°1

```
Variable Rep en Caractère
Début
Ecrire "Voulez vous un café ? (O/N)"
Lire Rep
Si Rep <> "O" ET Rep <> "N" Alors
   Ecrire "Saisie erronnée. Recommencez"
Lire Rep
FinSi
Fin
```

- C'est correct tant que l'utilisateur **ne se trompe qu'une seule fois,** et entre une valeur correcte à la deuxième demande.
- Si l'on veut prévoir un cas de deuxième erreur, il faudrait **rajouter un SI.** Et ainsi de suite, on peut rajouter des centaines de SI, et écrire un algorithme très lourd, et sans garantie de prévoir tous les cas d'erreur.
- La solution consistant à **aligner des SI est donc une impasse**

La structure de boucle

- Syntaxe

```
Tant que condition faire
        instructions
FinTantQue
```

- Explication
  - Le programme arrive sur la ligne du **TantQue**. Il examine alors la valeur de la condition. Si cette valeur est **VRAI**, le programme exécute les instructions qui suivent, jusqu'à ce qu'il rencontre la ligne **FinTantQue**.
  - Il retourne ensuite sur la ligne du **TantQue**, procède au même examen, et ainsi de suite.
  - Les exécutions successives du bloc d'instructions ne s'arrêtent que lorsque la condition prend la valeur **FAUX**.
  - Le programme continue alors avec l'exécution des instructions qui suivent la boucle, s'il y en a.

Contrôle de saisie avec boucle naïf

Exemple n°2

```
Variable Rep en Caractère
Début
Ecrire "Voulez vous un café ? (O/N)"
TantQue Rep <> "O" ET Rep <> "N" Faire
  Lire Rep
FinTantQue
Fin
```

- Là, on a le squelette de l'algorithme correct. Mais son principal défaut est de provoquer **une erreur à chaque exécution**.
- En effet, l'expression booléenne qui figure après le **TantQue** interroge la valeur de la variable Rep. Malheureusement, cette variable, si elle est été déclarée, n'a pas été affectée avant l'entrée dans la boucle.
- On teste donc une variable qui n'a pas de valeur, ce qui provoque l'arrêt immédiat de l'exécution.

Contrôle de saisie avec boucle avec une première lecture

- Pour éviter ceci, il faut que la variable **Rep** ait déjà été affectée avant qu'on en arrive au premier tour de boucle.
- Pour cela, on peut faire **une première lecture de Rep avant la boucle**. Dans ce cas, celle-ci ne servira qu'en cas de mauvaise saisie lors de cette première lecture.

Exemple n°3

```
Variable Rep en Caractère
Début
Ecrire "Voulez vous un café ? (O/N)"
Lire Rep
TantQue Rep <> "O" ET Rep <> "N" Faire
    Lire Rep
FinTantQue
Fin
```

Contrôle de saisie avec boucle avec initialisation arbitraire

- Une autre possibilité, fréquemment employée, consiste à ne pas lire, mais à **affecter arbitrairement la variable avant la boucle**. Cette affectation doit avoir pour résultat de **provoquer l'entrée obligatoire dans la boucle**. L'affectation doit donc faire en sorte que le booléen soit mis à VRAI pour déclencher le premier tour de la boucle.
- Dans notre exemple, on peut donc affecter Rep avec n'importe quelle valeur, hormis "O" et "N" : car dans ce cas, l'exécution sauterait la boucle, et Rep ne serait pas du tout lue au clavier.

Exemple n°4

```
Variable Rep en Caractère
Début
Rep ← "X"
Ecrire "Voulez vous un café ? (O/N)"
TantQue Rep <> "O" ET Rep <> "N" Faire
   Lire Rep
FinTantQue
Fin
```

Contrôle de saisie avec boucle avec une première lecture et affichage libellés

Exemple n°5

```
Variable Rep en Caractère
Début
Ecrire “Voulez vous un café ? (O/N)“
Lire Rep
TantQue Rep <> “O“ ET Rep <> “N“ Faire
    Ecrire “Vous devez répondre par O ou N. Recommencez“
    Lire Rep
FinTantQue
Ecrire “Saisie acceptée“
Fin
```

Contrôle de saisie avec boucle avec initialisation arbitraire et affichage libellés

Exemple n°6

```
Variable Rep en Caractère
Début
Rep ← "X"
Ecrire "Voulez vous un café ? (O/N)"
TantQue Rep <> "O" ET Rep <> "N" Faire
    Lire Rep
    Si Rep <> "O" ET Rep <> "N" Alors
        Ecrire "Saisie Erronée, Recommencez"
    FinSi
FinTantQue
Fin
```

Erreurs sur les conditions de boucle

- ERREUR 1 **:** écrire une structure TantQue dans laquelle **la condition n'est jamais VRAI**. Le programme ne rentre alors jamais dans la boucle !
- ERREUR 2 : écrire une boucle dans laquelle **la condition ne devient jamais FAUX**. L'ordinateur tourne alors dans la boucle infiniment et n'en sort plus. Seule solution, quitter le programme de façon brutale (CTRL+C par ex). La « **boucle infinie** » est une des hantises les plus redoutées des programmeurs.
- Comment faire pour éviter ces erreurs ?
  - Exécuter son algorithme « à la main » pour déterminer si le comportement de l'algorithme sera correct, en déterminant un **invariant de boucle** (une proposition qui est vraie à chaque tour de boucle).

Exemple de boucle TantQue

Ecrire un algorithme qui demande un nombre de départ (sans contrôle de saisie), 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.

Exemple n°7

```
Variable n en Numérique
Variable i en Numérique
Debut
Ecrire «  Entrer un nombre :»
Lire n
i ← 1
Tant que i <= 10 Faire
    Ecrire n+i
    i ← i + 1
FinTantQue
Fin
```

Exécution de l'algorithme à la main

Déclarations : n ← ? i ← ?
n ←17
i ← 1

1er tour : (i<=10) vaut (1<=10), condition VRAI
Ecrire 17+1, c'est-à-dire **Ecrire 18**
i ← i+1 c'est-à-dire i ←1+1 donc i ←2
Au début du 1er tour, i vaut 1; à la fin du 1er tour, i vaut 2

2eme tour : (i<=10) vaut (2<=10), condition VRAI
Ecrire 17+2, c'est-à-dire **Ecrire 19**
i ← i+1 c'est-à-dire i ←2+1 donc i ←3
Au début du 2ème tour, i vaut 2; à la fin du 2ème tour, i vaut 3

…

Au début du kème tour, i vaut k; à la fin du kème tour, i vaut k+1

Cette déclaration est vraie à chaque tour de boucle. C'est un ***invariant de boucle.***

Exécution de l'algorithme à la main

D'après l'invariant de boucle, on sait qu'au début du 10ème tour, i vaut 10, à la fin du 10me tour, i vaut 11.

10ème tour : (i<=10) vaut (10<=10), condition VRAI

Ecrire 17+10, c'est-à-dire **Ecrire 27**

i ← i+1 c'est-à-dire i ←10+1 donc i ←11

11ème tour : (i<=10) vaut (11<=10), condition FAUX

Le programme sort de la boucle. **Fin du programme**.

Conclusion : le programme affiche bien ce qui était demandé.

Style de boucle TantQue

- Plutôt que de définir une condition compliquée pour la boucle TantQue, une bonne façon d'écrire les boucles TantQue est **d'utiliser une variable booléenne pour la condition**, initialisée à VRAI.
- Dans la boucle, cette variable passe à FAUX lorsque la condition n'est plus remplie.

Exemple n°8

```
Variables n, i en Numérique
Variable estOK en Booléen
Debut
estOK ← VRAI
Ecrire «  Entrer un nombre :»
Lire n
i ← 1
Tant que estOK Faire
   Ecrire n+i
   i ← i + 1
   Si i > 10 Alors
        estOK ←  FAUX
   FinSi
FinTantQue
Fin
```

La Boucle Pour

La Boucle Pour

- Dans le dernier exemple, vous avez remarqué qu'une boucle pouvait être utilisée pour **augmenter la valeur d'une variable**.
- Cette utilisation des boucles est très fréquente, et dans ce cas, il arrive très souvent qu'on ait besoin d'effectuer **un nombre déterminé de passages**.
- Or, a priori, la structure TantQue ne sait pas à l'avance **combien** de tours de boucle elle va effectuer (puisque le nombre de tours dépend de la valeur d'une condition).
- La boucle Pour est utilisée quand le programmeur peut dénombrer à l'avance le nombre de tours de boucles nécessaires.

La Boucle Pour

- Syntaxe 1 :

```
Var cpt en Entier
Pour  cpt ← debut à fin Faire
      instructions
FinPour
```

- Explication
  - Le programme arrive sur la ligne du **Pour**. Il initialise la valeur du compteur cpt avec la valeur **debut**.
  - Si la valeur de cpt est inférieure ou égale à la valeur **fin**, il rentre dans la boucle : il exécute les instructions qui suivent, jusqu'à ce qu'il rencontre la ligne **FinPour**
  - Le programme augmente cpt de 1 (**incrémentation**) et retourne ensuite sur la ligne du **Pour**. Il recommence à comparer la valeur du compteur cpt avec la valeur fin.
  - Les exécutions successives du bloc d'instructions ne s'arrêtent que lorsque le compteur dépasse la valeur **fin** (pour le dernier tour, cpt vaut fin).
  - Le programme continue alors en sortant de la boucle : il reprend l'exécution des instructions qui suivent la boucle, s'il y en a.

Exemple de Boucle Pour

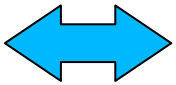
Ecrire un algorithme qui demande un nombre de départ, et qui calcule la somme des entiers jusqu'à ce nombre. Par exemple, si l'on entre 5, le programme doit calculer 1 + 2 + 3 + 4 + 5 = 15

Exemple n°9

```
Variables N, i, Som en Entier
Debut
Ecrire "Entrez un nombre : "
Lire N
Som ← 0
Pour i ← 1 à N Faire
    Som ← Som + i
FinPour
Ecrire "La somme est : " & Som
Fin
```

La boucle Pour est équivalente à une boucle TantQue

```
Variable i en Entier
Début
i ← 1
TantQue i <= 15 Faire
   Ecrire "Passage numéro : " & i
   i ← i + 1
FinTantQue
Fin
```



```
Variable i en Entier
Début
Pour i ← 1 à 15 Faire
   Ecrire "Passage numéro : " &i
FinPour
Fin
```

La structure « Pour … Faire » n'est pas du tout indispensable ; on pourrait fort bien programmer toutes les situations de boucle uniquement avec un « Tant Que ». Le seul intérêt du « Pour » est d'épargner un peu de fatigue au programmeur, en lui évitant de gérer lui-même la progression de la variable qui lui sert de compteur

La boucle Pour

- Syntaxe 1 :

```
Var cpt en Entier
Pour  cpt←debut à fin faire
      instructions
FinPour
```

- Dans une boucle Pour, la progression du compteur est laissée à votre libre disposition. Dans la plupart des cas, on a besoin d'une variable qui **augmente de 1 à chaque tour de boucle.** On ne précise alors rien à l'instruction « Pour » ; celle-ci, par défaut, comprend qu'il va falloir procéder à cette incrémentation de 1 à chaque passage, en commençant par la première valeur et en terminant par la deuxième.

Exemple de boucle Pour

Exemple n°12

```
debut
Var cpt en Entier
Var s en Numerique
s←5
Pour  cpt←1 à 7 faire
    Ecrire "s=" & s
    s ← 2 * s + 1
FinPour
fin
```

Affichage :

s=5

s=11

s=23

s=47

s=95

s=191

s=383

Cet exemple calcule et affiche les 7 premiers termes de la suite s définie par $s_0=5$ et $s_n = s_{(n-1)}*2 +1$

La boucle Pour avec pas personnalisé

- Syntaxe 2 :

```
Var cpt en Entier
Pour  cpt←debut à fin Pas valeurPas faire
      instructions
FinPour
```

- Si vous souhaitez une **progression plus spéciale**, de 2 en 2, ou de 3 en 3, ou en arrière, de –1 en –1, ou de –10 en –10, ce n'est pas un problème : il suffira de le préciser à votre instruction « Pour » en lui rajoutant le mot « Pas » et la valeur de ce pas (Le « pas » dont nous parlons, c'est le « pas » du marcheur, « step » en anglais).

Exemples de boucles Pour avec pas personnalisé

Exemple avec un pas de 4

```
Variable i,duree en Entier
Ecrire "Entrez la duree (entier positif)"
Lire duree
Début
Ecrire "Les jeux olympiques ont lieu aux dates :"
Pour i ← 2000 à (2000+duree) Pas 4 Faire
   Ecrire i
FinPour
Fin
```

- L'écart entre la valeur finale et la valeur initiale (ici la durée) n'est pas nécessairement un multiple du pas. La boucle s'arrête quand la valeur du compteur est strictement supérieure à la valeur finale.

Exemples de boucles Pour avec pas personnalisé

Exemple avec un pas de -1

```
Variable i en Entier
Début
Pour i ← 10 à 0 Pas -1 Faire
    Ecrire i
FinPour
Ecrire "Boom !"
Fin
```

- Avec un pas négatif dans une boucle, la valeur initiale du compteur doit être supérieure à sa valeur finale! Dans le cas contraire, on aura simplement écrit une boucle dans laquelle le programme ne rentrera jamais

Manipulation du compteur de boucle

Attention à ne pas manipuler le compteur de boucle au sein du bloc Pour !

Exemple n°10

```
debut
Variable Truc en Entier
Pour Truc ← 1 à 15 Faire
    Truc ← Truc * 2
    Ecrire "Passage numéro : ", Truc
FinPour
fin
```

Augmente la valeur de Truc de 1 à chaque passage

Double la valeur de Truc à chaque passage.

Affiche successivement :
« Passage numéro : 2»
« Passage numéro : 6»
« Passage numéro : 14»
« Passage numéro : 30»

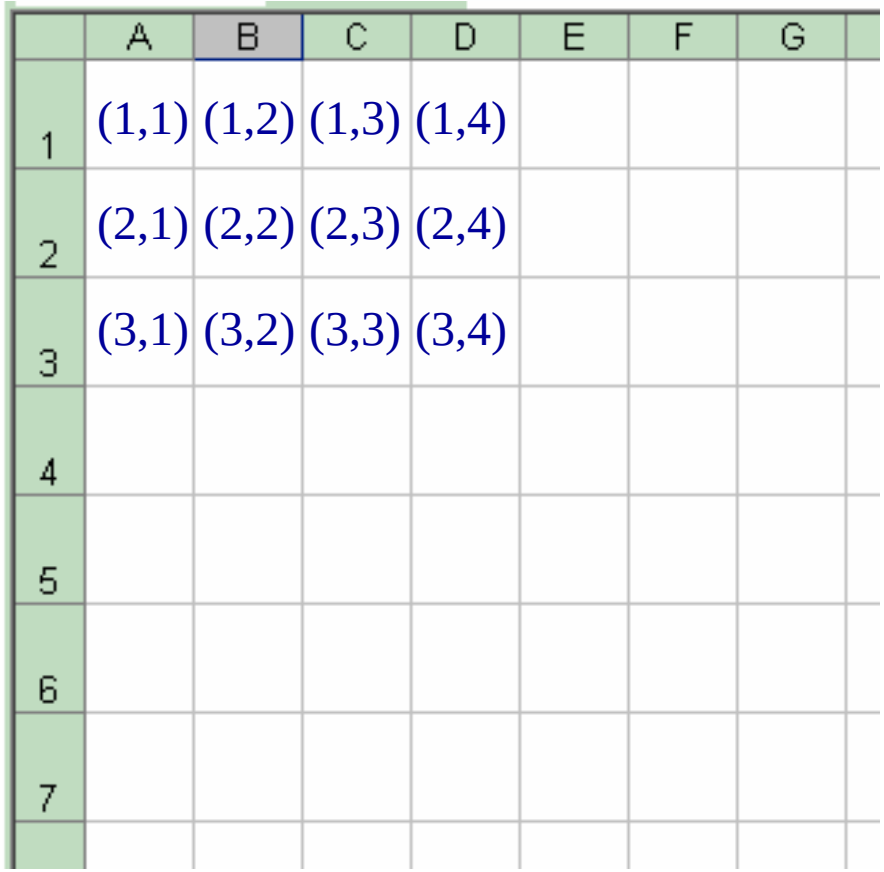
Boucles imbriquées

- Une boucle peut tout à fait contenir d'autres boucles. On dit que les boucles sont **imbriquées**.

Exemple n°11

```
debut
Variables ligne, colonne en Entier
Pour ligne ← 1 à 3 Faire
   Pour colonne ← 1 à 4 Faire
      Cellule(ligne,colonne) ← ″(″ & ligne & ″,″ & colonne & ″)″
   FinPour
FinPour
fin
```

Exécution d'une boucle imbriquée



	A	B	C	D	E	F	G
1	(1,1)	(1,2)	(1,3)	(1,4)			
2	(2,1)	(2,2)	(2,3)	(2,4)			
3	(3,1)	(3,2)	(3,3)	(3,4)			
4							
5							
6							
7							

1ère boucle : 1er tour : ligne=1
    2ème boucle :
        1er tour : colonne=1
        2ème tour : colonne=2
        3ème tour : colonne=3
        4ème tour : colonne=4
1ère boucle : 2ème tour : ligne=2
    2ème boucle :
        1er tour : colonne=1
        2ème tour : colonne=2
        3ème tour : colonne=3
        4ème tour : colonne=4
1ère boucle : 3ème tour : ligne=3
    2ème boucle :
        1er tour : colonne=1
        2ème tour : colonne=2
        3ème tour : colonne=3
        4ème tour : colonne=4

Application à la programmation VBA pour Excel

Boucles

- *Syntaxe :*

```
While Condition
    …
    [Exit While]
    …
Wend
```

```
For cptr=début To fin [Step valeurPas]
    …
    [Exit For]
    …
Next
```

- While...Wend répète un traitement jusqu'à ce qu'une certaine condition soit réalisée tandis que For...Next effectue un traitement un nombre de fois donné, en fonction d'un compteur. Le pas est facultatif. Par défaut, il est de 1.
- Facultativement, l'instruction Exit For permet de quitter directement une boucle For tandis que Exit While quitte directement une boucle While.

Boucles While : exemple

```
Sub chiffre()
' saisie d'un chiffre entre 10 et 20 jusqu'à ce que la valeur rentrée soit correcte
   Dim n As Integer
   Dim ok As Boolean
   ok = False
   While ok = False
      n = InputBox("Entrer un chiffre entre 10 et 20")
      If n < 10 Then
         MsgBox ("Trop petit")
      ElseIf n > 20 Then
         MsgBox ("Trop grand")
      Else
         MsgBox ("Vous avez rentré le chiffre " & n)
         ok = True
      End If
   Wend
End Sub
```

Boucles For : exemple

```
Sub puissance()
   ' calcule x puissance y
   ' y est supposé être un entier positif
   Dim x As Double
   Dim y As Integer
   Dim puissance As Double ‘ pour éviter le dépassement de capacité
   x = InputBox("Valeur de x ?")
   y = InputBox("Valeur de y ?")
   puissance = 1
   For i = 1 To y
      puissance = puissance * x
   Next
   MsgBox (x & " puissance " & y & "=" & puissance)
End Sub
```

Boucles : exercice

Modifier l’exemple précédent pour que le programme vérifie que la valeur donnée pour y soit positive et redemande à l’utilisateur de donner une nouvelle valeur pour y tant que ce n’est pas le cas:

1. Première méthode : en utilisant une condition sur y
2. Deuxième méthode : en utilisant une variable booléenne

Solution : première méthode

```
Sub puissanceVerif1()
    ' calcule x puissance y
    ' y doit être un entier positif
    Dim x As Double
    Dim y As Integer
    Dim puissance As Double
    x = InputBox("Valeur de x ?")
    y = InputBox("Valeur de y ?")
    While y < 0
        y = InputBox("y doit être positif. Recommencez.")
    Wend
    puissance = 1
    For i = 1 To y
        puissance = puissance * x
    Next
    MsgBox (x & " puissance " & y & "=" & puissance)
End Sub
```

Solution : deuxième méthode

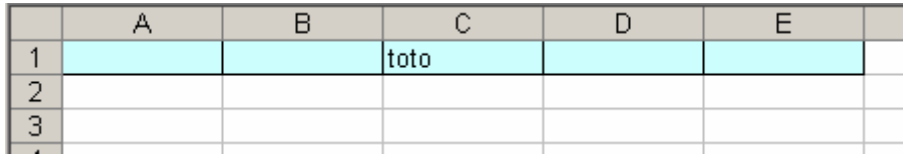
```
Sub puissanceVerif2()
    ' calcule x puissance y
    ' y doit être un entier positif
    Dim x As Double
    Dim y As Integer
    Dim puissance As Double
    Dim estOK As Boolean
    estOK = False
    x = InputBox("Valeur de x ?")
    While Not estOK
       y = InputBox("Valeur de y ?")
       If y >= 0 Then
          estOK = True
       ElseIf
            MsgBox(« Recommencez.")
        End If
    Wend

    puissance = 1
    For i = 1 To y
       puissance = puissance * x
    Next
    MsgBox (x & " puissance " & y & "=" & puissance)
End Sub
```

Boucles : exemple

```
Sub dessineEtoiles()
    ' affiche un tableau d'étoiles dans la feuille Excel courante en demandant
    ' le nombre de lignes à l'utilisateur
    Dim nbLignes As Integer
    Dim i As Integer
    Dim j As Integer
    nbLignes = InputBox("nb de lignes ?")
    j = 1
    For i = 1 To nbLignes
        For j = 1 To i
            Cells(i, j) = "X"
        Next
    Next
End Sub
```

Boucles : exemple avec interruption

'Affiche un message si le mot « toto » apparaît dans les 5 premières cases de la première ligne.




```
Sub testboucle()

    Dim i As Integer
    For i = 1 To 5
        MsgBox ("i=" & i)
        If Cells(1, i) = "toto" Then
            MsgBox "Toto existe sur la plage A1:A5“
            Exit For
        End If
    Next

End Sub
```

Interruption de la boucle si le test réussi. Les derniers tours de boucle ne sont pas effectués.