



A1	BUREAUTIQUE	
	TP14	

Objectif : Utiliser les conditions en VBA

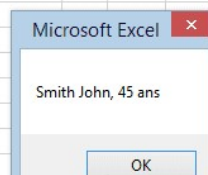
Les conditions sont très utiles en programmation, elles nous serviront à effectuer des actions en fonction de critères précis (même principe que la fonction SI).

La principale instruction est **If**, voici comment elle fonctionne :

```
If [CONDITION] Then '=> SI condition vraie ALORS
    'Instructions si vrai
Else '=> SINON (facultatif)
    'Instructions si faux
End If
```

Passons directement à la pratique et reprenons l'exemple développé à la leçon sur les variables. Il avait pour but d'afficher dans une boîte de dialogue la ligne du tableau correspondant au numéro indiqué dans la cellule F5.

	A	B	C	D	E	F	G	H
1	Nom	Prénom	Age					
2	Smith	John	40					
3	Smith	John	41					
4	Smith	John	42					
5	Smith	John	43			N° : 6		
6	Smith	John	44					
7	Smith	John	45					
8	Smith	John	46					
9	Smith	John	47					
10	Smith	John	48					
11	Smith	John	49					
12	Smith	John	50					
13	Smith	John	51					
14	Smith	John	52					



Si nous entrons une lettre en F5, cela génère un bug et nous voulons éviter cela.

```
Sub exemple()

    'Déclaration des variables
    Dim nom As String, prenom As String, age As Integer, numeroLigne As Integer

    'Valeurs des variables
    numeroLigne = Range("F5") + 1
    nom = Cells(numeroLigne, 1)
    prenom = Cells(numeroLigne, 2)
    age = Cells(numeroLigne, 3)

    'Boîte de dialogue
    MsgBox nom & " " & prenom & ", " & age & " ans"

End Sub
```

Nous allons commencer par ajouter une condition pour vérifier si la valeur de la cellule F5 est bien numérique avant d'exécuter le code.

La fonction **IsNumeric** sera utilisée dans cette condition :

```
Sub exemple()  
  
    'Si la valeur entre parenthèses (cellule F5) est numérique (donc si la condition  
est vraie) alors on exécute les instructions placées entre "Then" et "End If"  
    If IsNumeric(Range("F5")) Then  
  
        'Déclaration des variables  
        Dim nom As String, prenom As String, age As Integer, numeroLigne As Integer  
  
        'Valeurs des variables  
        numeroLigne = Range("F5") + 1  
        nom = Cells(numeroLigne, 1)  
        prenom = Cells(numeroLigne, 2)  
        age = Cells(numeroLigne, 3)  
  
        'Boîte de dialogue  
        MsgBox nom & " " & prenom & ", " & age & " ans"  
  
    End If  
  
End Sub
```

Ajoutons également des instructions pour le cas où la condition n'est pas remplie :

```
Sub exemple()  
  
    'Si F5 est numérique  
    If IsNumeric(Range("F5")) Then  
  
        'Déclaration des variables  
        Dim nom As String, prenom As String, age As Integer, numeroLigne As Integer  
  
        'Valeurs des variables  
        numeroLigne = Range("F5") + 1  
        nom = Cells(numeroLigne, 1)  
        prenom = Cells(numeroLigne, 2)  
        age = Cells(numeroLigne, 3)  
  
        'Boîte de dialogue  
        MsgBox nom & " " & prenom & ", " & age & " ans"  
  
    'Si F5 n'est pas numérique  
    Else  
  
        'Boîte de dialogue : avertissement  
        MsgBox "L'entrée "" & Range("F5") & "" n'est pas valide !"  
  
        'Suppression du contenu de la cellule F5  
        Range("F5") = ""  
  
    End If  
  
End Sub
```

Les valeurs non numériques ne sont désormais plus un problème.

Notre tableau contient 16 lignes de données (de la ligne 2 à la ligne 17), nous allons donc vérifier maintenant si la variable numeroLigne est plus grande ou égale à 2 et plus petite ou égale à 17.

Mais avant, voici les opérateurs de comparaison :

= Est égal à
<> Est différent de
< Est plus petit que
<= Est plus petit ou égal à
> Est plus grand que
>= Est plus grand ou égal à

Ainsi que d'autres opérateurs utiles :

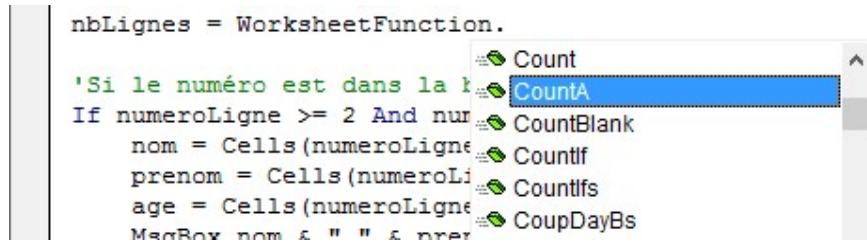
And	Et	[CONDITION 1] And [CONDITION 2] Les 2 conditions doivent être vraies
Or	Ou	[CONDITION 1] Or [CONDITION 2] Au moins 1 des 2 conditions doit être vraie
Not	Faux	Not [CONDITION] La condition doit être fausse
Mod	Modulo	[NOMBRE] Mod [DIVISEUR] Cet opérateur retourne le reste d'une division

Ajoutons maintenant les conditions indiquées un peu plus haut en utilisant And ainsi que les opérateurs de comparaison détaillés ci-dessus :

```
Sub exemple()  
  
    'Si F5 est numérique  
    If IsNumeric(Range("F5")) Then  
  
        Dim nom As String, prenom As String, age As Integer, numeroLigne As Integer  
        numeroLigne = Range("F5") + 1  
  
        'Si le numéro est dans la bonne plage  
        If numeroLigne >= 2 And numeroLigne <= 17 Then  
            nom = Cells(numeroLigne, 1)  
            prenom = Cells(numeroLigne, 2)  
            age = Cells(numeroLigne, 3)  
            MsgBox nom & " " & prenom & ", " & age & " ans"  
  
            'Si le numéro est en dehors de la plage  
        Else  
            MsgBox "L'entrée " & Range("F5") & " n'est pas un numéro valide !"  
            Range("F5") = ""  
        End If  
  
        'Si F5 n'est pas numérique  
    Else  
        MsgBox "L'entrée " & Range("F5") & " n'est pas valide !"  
        Range("F5") = ""  
    End If  
  
End Sub
```

Pour rendre notre macro plus pratique, nous pouvons encore remplacer 17 par une variable contenant le nombre de lignes. Cela nous permettra d'ajouter/retirer des lignes à notre tableau sans avoir à modifier à chaque fois cette limite dans le code.

Pour cela, créons une variable nbLignes et ajoutons cette fonction :



`WorksheetFunction.CountA` ne vous dit probablement rien mais il s'agit en fait de la fonction `NBVAL` que vous connaissez probablement déjà.

Nous demandons à cette fonction de comptabiliser le nombre de cellules non vides de la première colonne et nous remplaçons ensuite 17 par nbLignes :

```
Sub exemple()  
  
    'Si F5 est numérique  
    If IsNumeric(Range("F5")) Then  
  
        Dim nom As String, prenom As String, age As Integer, numeroLigne As Integer,  
        nbLignes As Integer  
  
        numeroLigne = Range("F5") + 1  
        nbLignes = WorksheetFunction.CountA(Range("A:A")) 'Fonction NBVAL  
  
        'Si le numéro est dans la bonne plage  
        If numeroLigne >= 2 And numeroLigne <= nbLignes Then  
            nom = Cells(numeroLigne, 1)  
            prenom = Cells(numeroLigne, 2)  
            age = Cells(numeroLigne, 3)  
            MsgBox nom & " " & prenom & ", " & age & " ans"  
  
            'Si le numéro est en dehors de la plage  
        Else  
            MsgBox "L'entrée " & Range("F5") & " n'est pas un numéro valide !"  
            Range("F5") = ""  
        End If  
  
        'Si F5 n'est pas numérique  
    Else  
        MsgBox "L'entrée " & Range("F5") & " n'est pas valide !"  
        Range("F5") = ""  
    End If  
End Sub
```

ELSEIF

Elseif permet d'ajouter plusieurs conditions à la suite :

```
If [CONDITION 1] Then '=> SI la condition 1 est vraie ALORS
    'Instructions 1
ElseIf [CONDITION 2] Then '=> SINON, SI la condition 2 est vraie ALORS
    'Instructions 2
Else '=> SINON
    'Instructions 3
End If
```

Si la condition 1 est vraie, les instructions 1 sont exécutées puis nous sortons de l'instruction If (qui débute avec If et se termine à End If). Si la condition 1 est fausse, nous passons à la condition 2. Si celle-ci est vraie les instructions 2 sont exécutées si ce n'est pas le cas les instructions 3 sont alors exécutées.

Voici un exemple, avec en A1 une note de 1 à 6 et en B1 un commentaire en fonction de la note :

```
Sub commentaires()

    'Variables
    Dim note As Single, commentaire As String
    note = Range("A1")

    'Commentaire en fonction de la note
    If note = 6 Then
        commentaire = "Excellent résultat !"
    ElseIf note >= 5 Then
        commentaire = "Bon résultat"
    ElseIf note >= 4 Then
        commentaire = "Résultat satisfaisant"
    ElseIf note >= 3 Then
        commentaire = "Résultat insatisfaisant"
    ElseIf note >= 2 Then
        commentaire = "Mauvais résultat"
    ElseIf note >= 1 Then
        commentaire = "Résultat exécration"
    Else
        commentaire = "Aucun résultat"
    End If

    'Commentaire en B1
    Range("B1") = commentaire

End Sub
```

SELECT

Une alternative aux instructions If contenant beaucoup de Elseif existe, il s'agit de Select (cette instruction étant plus adaptée dans ce genre de cas).

Voici la même macro avec Select :

```
Sub commentaires()  
  
    'Variables  
    Dim note As Single, commentaire As String  
    note = Range("A1")  
  
    'Commentaire en fonction de la note  
    Select Case note '  
        Case Is = 6  
            commentaire = "Excellent résultat !"  
        Case Is >= 5  
            commentaire = "Bon résultat"  
        Case Is >= 4  
            commentaire = "Résultat satisfaisant"  
        Case Is >= 3  
            commentaire = "Résultat insatisfaisant"  
        Case Is >= 2  
            commentaire = "Mauvais résultat"  
        Case Is >= 1  
            commentaire = "Résultat exécration"  
        Case Else  
            commentaire = "Aucun résultat"  
    End Select  
  
    'Commentaire en B1  
    Range("B1") = commentaire  
  
End Sub
```

Notez que nous pouvons également entrer plusieurs valeurs :

```
Case Is = 6, 7 'Si la valeur = 6 ou 7  
Case Is <> 6, 7 'Si la valeur est différente de 6 ou 7
```

Ou une plage de valeurs :

```
Case 6 To 10 'Si la valeur = de 6 à 10
```

FONCTION ISNUMERIC

La fonction IsNumeric (vue à la page précédente) renvoie True (vrai) si la valeur est numérique et False (faux) si ce n'est pas le cas :

```
If IsNumeric(Range("A1")) = True Then  
If IsNumeric(Range("A1")) Then
```

Ces 2 lignes sont identiques (il n'est pas nécessaire d'entrer = True puisque que l'on cherche de toute manière à savoir si l'expression est vraie).

Dans le cas où nous voulons vérifier si la valeur n'est pas numérique, nous avons également deux possibilités :

```
If IsNumeric(Range("A1")) = False Then 'Si la valeur n'est pas numérique  
If Not IsNumeric(Range("A1")) Then 'Si la valeur n'est pas numérique
```

Il existe de nombreuses autres fonctions que vous pouvez utiliser dans vos conditions (ou plus généralement dans vos codes VBA).

Vous pourrez retrouver la liste des principales fonctions VBA sur la page [Fonctions VBA](#).

FONCTIONS DE DATES

Il existe de nombreuses [fonctions de dates et d'heures](#) pouvant être utilisées dans des conditions, en voici quelques exemples.

La fonction **ISDate** renvoie True si la valeur est une date ou False si ce n'est pas le cas :

```
If IsDate(Range("A1")) Then 'Si la valeur est une date
```

La fonction **Day** permet d'extraire le jour d'une date :

```
If Day(Range("A1")) = 1 Then 'Si c'est le premier jour du mois
```

La fonction **Year** permet d'extraire l'année d'une date :

```
If Year(Range("A1")) = 2021 Then 'Si c'est une date de l'année 2021
```

La fonction **Weekday** renvoie le numéro du jour de la semaine :

```
If Weekday(Range("A1"), 2) >= 6 Then 'Si c'est un samedi ou un dimanche
```

La fonction **Date** renvoie la date actuelle :

```
If Range("A1") < Date Then 'Si la date est passée
```

FONCTION ISEMPY

La fonction **IsEmpty** renvoie False si la variable a été initialisée ou True si ce n'est pas le cas :

```
If IsEmpty(maVariable) Then 'Si la variable n'a pas été initialisée
```

Dans cet exemple, la condition est vraie car aucun type ni valeur n'ont été attribués à maVariable :

```
Sub exemple()  
  
    Dim maVariable  
  
    If IsEmpty(maVariable) Then  
        MsgBox "Ma variable n'a pas été initialisée !"  
    Else  
        MsgBox "Ma variable contient : " & maVariable  
    End If  
  
End Sub
```

CONDITION EN FONCTION DE LA COMPARAISON DE 2 CHÂÎNES DE CARACTÈRES

Jusque-là nous n'avons vu que cela :

```
maVariable = "Exemple 12345"  
  
If maVariable = "Exemple 12345" Then '=> Vrai
```

Dans ce cas, les 2 chaînes de caractères sont identiques, l'expression est donc vraie.

Maintenant, pour vérifier si la variable contient la valeur 12345 sans tenir compte des autres caractères, nous utiliserons l'opérateur Like ainsi que * devant et derrière la valeur à rechercher.

Le caractère * peut remplacer : aucun, un ou plusieurs caractères :

```
maVariable = "Exemple 12345"  
  
If maVariable Like "*12345*" Then '=> Vrai
```

Le caractère # peut remplacer un caractère numérique de 0 à 9 :

```
maVariable = "Exemple 12345"  
  
If maVariable Like "Exemple 12###" Then '=> Vrai
```

Le caractère ? peut remplacer un caractère quelconque :

```
maVariable = "Exemple 12345"  
  
If maVariable Like "?xemple?1234?" Then '=> Vrai
```

Nous pouvons également remplacer un caractère en fonction d'une plage de caractères ou de caractères précis :

- [abc] : remplace un des caractères suivants : a b c
- [a-g] : remplace un des caractères suivants : a b c d e f g
- [369] : remplace un des caractères suivants : 3 6 9
- [2-5] : remplace un des caractères suivants : 2 3 4 5
- [?*#] : remplace un des caractères suivants : ? * #

```
maVariable = "Exemple 12345"  
  
If maVariable Like "[BIEN]xemple 1234[4-7]" Then '=> Vrai
```

Pour remplacer un caractère non compris dans les valeurs entre crochets, un ! doit être ajouté après [:

```
maVariable = "Exemple 12345"  
  
If maVariable Like "[!FAUX]xemple 1234[!6-9]" Then '=> Vrai
```

Un caractère en majuscule n'est pas égal à ce même caractère en minuscule. Pour ne pas faire de distinctions entre majuscules-minuscules, placez Option Compare Text en début de module.