

VBA 2

Sauf lorsqu'il est porté une mention contraire, les questions de ce TD se font à partir d'un classeur vierge.

1- Instructions conditionnelles

Un exemple d'instruction conditionnelle simple

Dans certains cas, on veut que le code qui s'exécute dépende de certaines conditions. Par exemple, en fonction du fait que l'utilisateur a répondu "non" ou "oui" à une question posée, ce n'est pas le même comportement que la machine doit adopter. Pour prendre en compte cela, on dispose des instructions conditionnelles.

Prenons un premier cas, on dispose d'une note en A1. Si cette note est supérieure à 10, alors il faut afficher "reçu" en B1, sinon, il faut afficher "non admis".

Le code suivant permet de le faire :

```
Sub testNote1()  
Dim note as double, appreciation as string  
note=cells(1,1)  
    if note >=10 then  
        appreciation="Reçu"  
    else  
        appreciation="Non admis"  
    end if  
cells(1,2)=appreciation  
End Sub
```

L'instruction conditionnelle est délimitée entre If et End If.

On peut tester le code tel quel pour vérifier qu'il fonctionne, en prenant soin de renseigner la case A1 avant de lancer le code.

Concernant la présentation du code, on a choisi de décaler d'une tabulation les lignes de code qui correspondent à l'instruction conditionnelle. Ceci n'a aucun impact pour la machine. Il s'agit juste de faciliter la lecture du code pour l'humain. On parle d'"indentation" du code.

Des différentes formes de l'instruction conditionnelle

L'instruction conditionnelle peut prendre différentes formes. Supposons par exemple que l'on veuille, sur la base du même exemple que précédemment, afficher "Reçu" si l'étudiant est reçu et rien sinon. Le code devient :

```
Sub testNote2()  
  
Dim note as double, appreciation as string  
note=cells(1,1)  
    if note >=10 then  
        appreciation="Reçu"  
    end if  
cells(1,2)=appreciation  
End Sub
```

On aurait tout aussi bien pu faire :

```
Sub testNote3()  
    if cells(1,1) >=10 then  
        cells(1,2)="Reçu"  
    end if  
End Sub
```

Supposons désormais, qu'à l'inverse, on veuille complexifier l'appréciation. On veut toujours afficher "Non admis" pour l'étudiant qui n'est pas admis, mais pour celui qui est admis, on veut afficher : "Bien" s'il a plus de 14, "AB" s'il a entre 12 et 14, "Passable" s'il est entre 10 et 12. Le code suivant permet de le faire :

```
Sub testNote4()  
  
Dim note as double, appreciation as string  
note=cells(1,1)  
    if note <10 then  
        appreciation="Non admis"  
    elseif note>=14 Then  
        appreciation="Bien"
```

```

elseif note >= 12 then
    appreciation = "Assez Bien"
else
    appreciation = "Passable"
end if
cells(1,2) = appreciation
End Sub

```

Else correspond à la condition par défaut, si aucun des autres conditions n'a été rempli. On peut spécifier un Else ou on peut ne pas en spécifier.

Ex1

On suppose que l'utilisateur met un patrimoine en A4. On veut calculer l'impôt sur le patrimoine et l'afficher en B4. L'impôt se calcule selon les règles suivantes :

- Pour un patrimoine inférieur à 700 000 €, il n'est pas soumis à imposition. L'impôt sera donc de 0
- Pour un patrimoine entre 700 000 € et 1 200 000 €, il est soumis à un impôt de 3%. Ainsi, par exemple, un patrimoine de 800 000 € donnera un impôt de 24 000 €
- Pour un patrimoine supérieur à 1 200 000 €, il sera imposé à 3% sur le patrimoine jusqu'à 1 200 000, puis à 5% sur le patrimoine au-delà. Ainsi, un patrimoine de 2 000 000 € donnera un impôt de 76 000 €

Implémenter un programme permettant de calculer cet impôt et vérifier qu'il fonctionne sur les exemples donnés.

Des instructions conditionnelles au sein d'instructions conditionnelles

Des cas plus complexes peuvent être prévus. Si l'on reprend le cas de la notation, on peut avoir une note inférieure à 0 ou supérieure à 20 rentrée par l'utilisateur. On voudrait afficher une erreur si jamais c'est le cas.

Le code suivant permet de le faire :

```

Sub testNote5()
Dim note as double, appreciation as string
note = cells(1,1)
    if note <= 20 and note >= 0 then
        if note < 10 then
            appreciation = "Non admis"

```

```

        elseif note >= 14 Then
            appreciation = "Bien"
        elseif note >= 12 then
            appreciation = "Assez Bien"
        else
            appreciation = "Passable"
        end if
    else
        appreciation = "Erreur"
    end if
    cells(1,2) = appreciation
End Sub

```

A noter que le code suivant donne strictement le même résultat :

```

Sub testNote6()
    Dim note as double, appreciation as string
    note = cells(1,1)
    if note > 20 or note < 0 then
        appreciation = "Erreur"
    else
        if note < 10 then
            appreciation = "Non admis"
        elseif note >= 14 Then
            appreciation = "Bien"
        elseif note >= 12 then
            appreciation = "Assez Bien"
        else
            appreciation = "Passable"
        end if
    end if
    cells(1,2) = appreciation
End Sub

```

- Bien noter ici les if imbriqués sur le mode des poupées russes.
- Bien noter ici que l'on peut construire des conditions complexes avec or et and.

Ex2

On suppose que l'utilisateur saisit un revenu en A7 et un genre en B7 (une chaîne de caractères). On veut afficher son impôt en C7, en suivant les règles suivantes :

- Pour un homme de revenu inférieur à 1 200 €, il n'est pas imposé
 - Pour une femme de revenu inférieur à 1 500 €, elle n'est pas imposée
 - Pour tout individu imposé, avec des revenus inférieurs à 3000 €, il est imposé à 20% de son revenu
 - Pour tout individu de revenu de plus de 3 000 €, il est imposé à 33% de son revenu.
 - L'imposition totale ne peut pas dépasser 10000 €
- Proposer une version du code permettant de remplir C7

2- Boucle for

La boucle for permet de répéter les opérations. La forme de la boucle est la suivante :

```
Dim i as integer
[...
    For i=1 to 10
        [...]
    Next
[...]
```

Ici la boucle s'exécutera une première fois pour i=1, une deuxième fois avec i=2, une troisième fois avec i=3 etc... Jusqu'à i=10.

Ex3

On considère le code suivant :

```
Sub testBoucle1()
Dim i as integer
    For i=1 to 10
        Cells(i,1)=i
    Next
```

```
End sub
```

- 1) *Que fait ce code ?*
- 2) *Combien de fois s'exécute la boucle ?*

On considère le code suivant :

```
Sub testBoucle2()  
Dim i as integer  
    For i=1 to 10  
        Cells(i,1)=2^i  
    Next  
End sub
```

- 3) *Que produit ce code ?*
-

Ex4

On considère le code suivant :

```
Sub testBoucle3()  
Dim i as integer, n as integer  
n=1  
    For i=1 to 10  
        Cells(n,1)=i  
    Next  
End sub
```

- 1) *Que fait ce code ?*
 - 2) *La boucle a-t-elle un intérêt ici ?*
-

Ex5

On considère le code suivant :

```
Sub testBoucle4()  
Dim i as integer, n as integer  
n=1  
    For i=1 to 10  
        Cells(n,1)=i  
    Next
```

```
End sub
```

- 1) *Que fait ce code ?*
- 2) *La boucle a-t-elle un intérêt ici ?*

Ex6

On voudrait établir la base d'un échéancier à partir de la date du jour. On voudrait afficher la date du jour en A3, puis on voudrait afficher la date qu'il sera dans un mois en A4, puis la date qu'il sera dans deux mois en A5 etc, jusqu'à afficher 24 dates.

- 1) *Proposer un code pour afficher la date du jour en A3 (fonction Now())*
- 2) *Proposer un code pour réaliser l'échéancier.*

Ex7

On considère le fichier VBA2_1.xlsm. On veut compléter la colonne C. Pour chaque personne, on calcule son impôt en prenant pour impôt :

- *5% du revenu si le revenu est inférieur à 20 000 €*
- *15% si le revenu est entre 20 000 € et 50 000 €*
- *30% au-delà.*

- 1) *Proposer un code pour calculer l'impôt pour la première personne (ligne 2)*
- 2) *Proposer un code pour compléter l'ensemble de la colonne*

3- Boucle while

La boucle while est plus générale. Elle s'exécute tant qu'une condition est vérifiée.

```
[...]  
    While condition  
        [...]  
    Wend  
[...]
```

Par exemple, on peut utiliser le nombre des lignes qui sont complétées dans le fichier testImpot.xlsm. Pour ce faire :

```
Sub testBoucle5()
```

```

Dim i as Integer, compte as Integer
i=1
compte=0
    while cells(i,1)<>" "
        compte=compte+1
        i=i+1
    Wend
Msgbox "Il y a " & compte & "lignes remplies."
End sub

```

La boucle est plus générale. Il n'y a pas de meilleure boucle entre la boucle while et la boucle for.

Attention à la boucle infinie si on enlève i=i+1 : la boucle n'évolue plus, la condition n'évolue plus et donc, elle est toujours vérifiée => boucle infinie.

Ex8

Proposer une nouvelle version pour compléter la colonne C de VBA2_1 avec une boucle while. Quelle est la meilleure solution entre les deux ?

4- Somme, moyenne et max avec les boucles

Somme de valeurs

On suppose que l'on veut faire la somme des revenus et l'afficher en F1 dans la feuille testImpots.xlsm. On commence par une boucle for :

```

Dim i as integer, Somme as double
Somme=0
    for i=2 to 417
        Somme=Somme+cells(i,2)
    next
cells(1,6)=somme

```

La même chose avec une boucle while :

```

Dim i as integer, Somme as double
Somme=0
i=2

```

```

        while cells(i,1)<>" "
            Somme=Somme+cells(i,2)
            i=i+1
        wend
cells(1,6)=somme

```

Moyenne de valeurs

La moyenne des valeurs est obtenue avec un code proche. On veut afficher la moyenne des revenus

```

Dim i as integer, Somme as double
Somme=0
    for i=2 to 417
        Somme=Somme+cells(i,2)
    next
cells(2,6)=somme/416

```

La même chose avec une boucle while :

```

Dim i as integer, Somme as double, c as integer
Somme=0
i=2
c=0
    while cells(i,1)<>" "
        Somme=Somme+cells(i,2)
        c=c+1
        i=i+1
    wend
cells(2,6)=somme/c

```

Max de valeurs

Le max peut s'obtenir de la manière suivante avec une boucle for (pour afficher le revenu max en F3 :

```

Dim i as integer, max as double
max= cells(2,2)

```

```

    for i=2 to 417
        if cells(i,2)>max then
            max=cells(i,2)
        end if
    next
cells(3,6)=max

```

La même chose avec une boucle while :

```

Dim i as integer, max as double
i=2
max=cells(2,2)
    while cells(i,1)<>" "
        if cells(i,2)>max then
            max=cells(i,2)
        end if
        i=i+1
    wend
cells(3,6)=max

```

Ex9

Ecrire un algorithme qui demande 5 valeurs à l'utilisateur et affiche la valeur maximale.

Ex10

Dans testImpot.xlsm, faire un ou plusieurs codes permettant de compléter G5:G8.

Ex11

Ecrire un algorithme qui permet de tester si une chaîne de caractères est un palindrome

Ex12

Un nombre parfait est égal à la somme de ses diviseurs, 1 y compris. Ex : $6=1+2+3$. Proposer un algorithme qui permet de trouver au moins 3 nombres parfaits.

Ex13

Trouver un algorithme qui permet de calculer et d'afficher les nombres premiers dans la première colonne d'un tableau.

Ex14

Faire un algorithme qui demande à l'utilisateur un nombre entier n . L'algorithme va ensuite attendre n secondes puis afficher "Coucou".

Ex15

A partir du fichier VBA2_2 qui consigne des données autoroutières,

- 1) Un accident a eu lieu. Une photographie a permis de voir que l'immatriculation du véhicule en fuite était ??-349-IT. Proposer un algorithme qui permet de retrouver l'immatriculation complète du véhicule
 - 2) On voudrait identifier les véhicules qui ont roulé trop vite, soit ceux qui ont mis moins de 4 heures à parcourir la distance entre les deux péages. Afficher en rouge les lignes avec de tels véhicules. Pour afficher la ligne i en rouge : `Rows(i).interior.color=rgb(255,0,0)`
-

Ex16

- 1) Donner un algorithme qui permet de dire si un nombre est premier.
 - 2) Donner un algorithme qui donne les 100 premiers nombres premiers.
-

Ex17

Donner un algorithme qui permet de dire si une chaîne de caractères est ou non un palindrome
