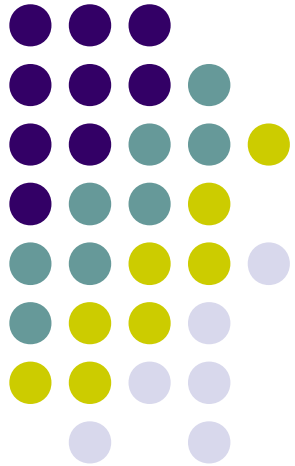
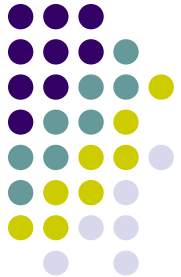
Outils informatiques 2
Cours n°7: Fonctions et procédures

Claire Hanen

Juliette Arnal



Rappel



On peut définir soi même des fonctions et procédures

Fonctions :

Function nomfonc(liste de paramètres formels) **As** nomtype

Instructions permettant de calculer le résultat de la fonction, avec une instruction (ou plusieurs) du type : nomfonc = expression

End Function

Procédures

Sub nomproc(liste de paramètres formels)

Instructions

End Sub

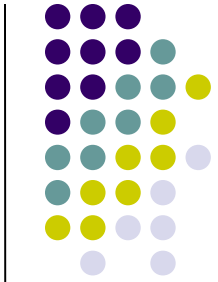
L'appel peut être réalisé en utilisant la syntaxe par paramètres listés ou par paramètres nommés.

Remarque

Sous Excel, **il est possible d'utiliser une fonction programmée comme une fonction de cellule,** avec la même syntaxe d'appel que les autres fonctions d'Excel (paramètres effectifs séparés par des points-virgule).

Par contre, seules les procédures sans paramètres peuvent être appelées depuis la feuille excel (par l'intérmédiaire de macros-exécuter)

Exemple : fonction de conversion dollar vers euro



```
Function conveuro( D As Double) As Double
conveuro=D*0,7457
End Function
```

Deux moyens d’appeler cette fonction.

Exemple : fonction conveuro

Méthode 1 avec Excel :

B1=CONVEURO(A1)

Méthode 2 dans une macro :

```
Sub dialogue
Dim S As double
S=InputBox(« entrez une somme en dollars »)
MsgBox(S & « euros  valent « & conveuro(S) & « dollars »)
End sub
```

Exemple: fonction Bonres

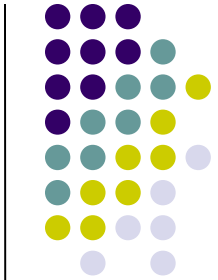
- Function Bonres(x as integer, y as integer, z as integer, operateur as integer) as boolean
- If operateur=0 then
- Bonres=(z=x+y)
- Else
- Bonres=(z=x*y)
- End if
- End Function

Vérifie si z=x+y ou z=x*y selon la valeur du paramètre opérateur.

Procédures avec paramètres

- Lorsque l'utilisateur définit une procédure avec paramètres
- Elle est appelée dans une macro précédée du mot **Call**
- **Cette notion est surtout intéressante lorsqu'on manipule des objets Excel (on y reviendra).**

Exemple de procédure avec et sans paramètre et d'appel



- On a écrit un jeu de l'addition. On pourrait écrire également un jeu de multiplication. Le code serait presque identique. Afin de ne pas réécrire le tout deux fois on peut utiliser un paramètre, valeur 0 pour addition, 1 pour multiplication.

```
Sub jeu(operation as integer)
…
End sub
```

- On peut alors écrire une macro sans paramètre:

```
Sub jouer()
Dim queljeu
queljeu=MsgBox(prompt:=« pour jouer avec des additions, bouton oui, pour jouer avec des multiplications,
      bouton non », buttons:=vbyesno)
If queljeu=vbYes then
Call jeu(0)
Else
Call jeu(1)
End If
End sub
```

Jeu simplifié

```
Sub jeu(operation as integer)
Dim x as integer, y as integer, z as integer
Dim op as string
If operation =0 then
op =«  plus  »
else
op=«  fois  »
End if
do
x = Int(Rnd() * 100) + 1
y = Int(Rnd() * 100) + 1
Z=inputbox(« combien font «  & x & op & y & « ? »)
If bonres(x,y,z,op) then
MsgBox(« bravo !»)
Else
MsgBox(« raté! »)
End if
Loop Until (MsgBox(prompt:="voulez-vous recommencer?", Title:="jeu", Buttons:=vbYesNo) = vbNo)
End sub
```

Exemple : fonction maximum

Définir une fonction maximum(n1,n2,n3) qui retourne le maximum entre trois nombres.
Utiliser ensuite cette fonction dans une macro avec des nombres donnés par l'utilisateur.

La fonction :

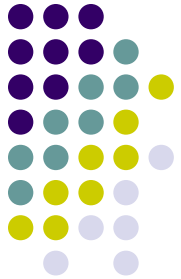
```
Function max3(x As Integer, y As Integer, z As Integer) As Integer
If x >= z And x >= y Then
max3 = x
End If
If y >= z And y >= x Then
max3 = y
End If
If z >= y And z >= x Then
max3 = z
End If
End Function
```

Exemple : fonction maximum

Le programme qui appelle la fonction :

```
Sub maximum()
Dim x As Integer, y As Integer, z As Integer
x = InputBox("Donner le premier nombre")
y = InputBox("Donner le deuxième nombre")
z = InputBox("Donner le troisième nombre")
MsgBox ("Le maximum est : " & max3(x, y, z))
End Sub
```

Exemple d'une économie d'échelle: fonction factorielle



On souhaite calculer 3!+5!+8!

Méthode :

- Définir un programme qui calcul le factorielle d'une valeur
- Ecrire le programme en recopiant le calcul pour 3!, 5! et 8!

… Ou penser à passer par une fonction

Exemple : factorielle

Le programme qui effectue le factorielle de 5 :

```
Sub PgrFactorielle()
        Dim f As Double
        Dim i As Integer , x as integer
        f = 1
        x = 5
        For i = 1 To x
            f = f * i
        Next
        MsgBox(f)
End Sub
```

On utilise le type Double car les nombres sont grands

Exemple : factorielle
le programme qui calcule : 3! + 5! + 8!

```
Sub PgrFactorielle()
        Dim f As Double Dim i, somme As Integer
        somme =0,
        f = 1
        For i = 1 To 3
            f = f * i
        Next
        somme = somme + f
        f = 1
        For i = 1 To 5
            f = f * i
        Next
        somme = somme + f
        f = 1
        For i = 1 To 8
            f = f * i
        Next
        somme = somme + f
        MsgBox(somme)
End Sub
```

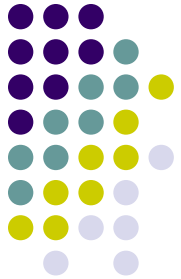
Exemple : utilisation d'une fonction factorielle

```
Sub calcul()
    Dim res As Double
    res = facto(3) + facto(5) + facto(8)
    MsgBox ("le résultat vaut " & res)
End Sub
```

Le code est constitué de la procédure calcul() et de la fonction facto(n). Cette fonction est définie avec une variable n de type Integer ; la valeur calculée par la fonction est de type Double.

```
Function facto(n As Integer) As Double
    Instructions …
    facto = resultat
End Function
```

Exemple : fonction factorielle Programme final



```
Sub calcul()
    Dim res As Double
    res = facto (3) + facto (5) + facto (8)
    MsgBox ("le résultat vaut " & res)
End Sub
Function facto (n As Integer) As Double
    Dim f As Double
    Dim i As Integer
    f = 1
    For i = 1 To n
        f = f * i
    Next
    facto = f
End Function
```

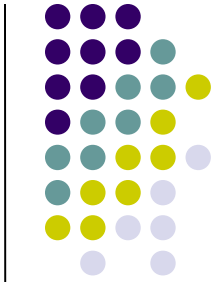
Récursivité

- Le mécanisme d'appel de fonction permet une écriture récursive: chaque fois que le nom de la fonction apparait dans une expression, le programme correspondant est exécuté.

```
Function facto(n as integer) as double
If n=0 then
Facto=1
Else
Facto=n*facto(n-1)
End If
End Function
```

Attention, dans ce type d'écriture, il faut veiller à ce que la fonction s'arrête. Ici, il y a n appels successifs de facto

Récursivité



- Facto (5)
  - Facto(4)
    - Facto(3)
      - Facto (2)
        - Facto(1)

Facto(0)=1

=1*1

=2*1

=3*2

=4*3*2

=5*4*3*2

Exemple : fonction capi (valeur d'un placement)

Calculer la valeur acquise d'un placement (m = 1000 €) aux taux (i = 5%) pendant (n = 10 années)

```
Function capi (m As Double, i As Double, n As Integer) As Double
    If n = 1 Then
      capi = m * (1+i)
    Else
      capi = capi (m, i, n - 1)* (1+ i)
    End If
End Function

Sub valeur ()
    Dim res As Double
    res = capi (1000, 0.05, 10)
MsgBox (res)
End Sub
```

Performance d'une écriture récursive.

```
Function capi (m As Double, i As Double, n As
    Integer) As Double
    If n = 1 Then
     capi = m * (1+i)
    Else
     capi = capi (m, i, n - 1) + capi(m, i, n-1)* i
    End If
End Function
```

Ici, le nombre d'appels pour n années serait en fait 2n (à chaque fois la fonction est appelée deux fois) Rappel pour n=10 au lieu d'avoir 10 appels (avec l'écriture précédente), il y en aurait 1024. Pour n =20, plus d'un million (au lieu de 20)

Ecriture itérative de la fonction

```
Function capi (m As Double, i As Double, n As Integer)
    As Double
Dim k as integer, res as double
res=m
    for k=1 to n
     res=res * (1+i)
    Next k
capi=res
End Function
```

Exercice d'Annales : Pile ou face équilibré

On veut simuler un lancer de pièce avec les fonctions de tirage aléatoire. Pour cela, on utilise la fonction aléatoire Rnd() qui renvoie un nombre aléatoire entre 0 et 1. Si le tirage est inférieur à 1/2 alors le tirage est assimilé à pile, sinon à face.

Question n°1 : Considérons la fonction suivante:

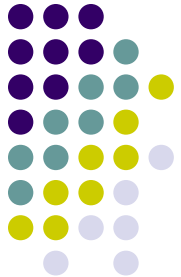
```
Function Pileface() As Integer
Dim x as double
x=Rnd()
If x<1/2 Then
Pileface=0
Else
Pileface=1
End If
End Function
```

Solution 1:

```
sub tirage()
Dim x as integer
x=pileface()
if x=1 then
MsgBox(″face″)
Else
MsgBox(″Pile″)
End If
End Sub
```

Ecrire une macro qui utilise cette fonction, en effectuant un tirage puis affiche, selon le cas, un message indiquant pile ou face.

Question n°2 : Ecrire une macro qui effectue 100 lancers de pièces successifs et qui affiche le pourcentage de lancers ayant donné face.



Solution 2:

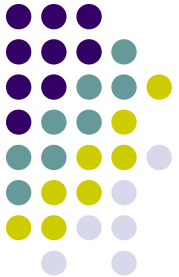
```
sub centirage()
Dim nbface As integer
Dim i As integer
Dim x As integer
nbface=0
For i=1 to 100
nbface=nbface+pileface()
Next
MsgBox("il y a eu "& nbface&" % de face" )
End Sub
```

Question n°3 : Ecrire une fonction appelée propface a un paramètre entier n, qui effectue n lancers de pièce et retourne la proportion de tirages ayant donné face.

ATTENTION cette fonction ne doit pas réaliser d'entrée-sorties.

Solution 3 :

```
Function propface(n as integer) As double
Dim nbface As integer
Dim i As integer
Dim x As integer
nbface=0
For i=1 to n
nbface=nbface+pileface()
Next
propface=nbface/n
End Function
```

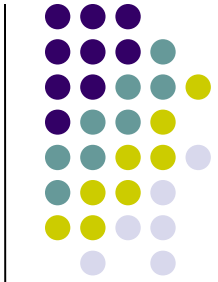


Question n°4 : En supposant qu'un entier x soit stocké dans la cellule A1 de la feuille de calcul, comment afficher en B1 la proportion de face obtenue avec x tirages ?

Solution 4:

il suffit d'écrire dans la cellule B1:

=PROPFACE(A1)



Question n°5 : On cherche à mesurer l'écart entre la proportion observée de lancers face, et la proportion théorique lorsque le tirage est bien uniforme : 1/2. Ecrire une macro qui calcule et affiche cet écart pour 100 lancers, puis 200, puis 300, … jusqu'à 1000. A chaque étape l'affichage réalisé doit être sous la forme : « pour x tirages l'écart est de y », où x et y sont les valeurs calculées.

Solution 5:

```
sub ecart()
Dim i as Integer
Dim e as double
For i=1 to 10
e=propface(i*100)-1/2
MsgBox (″pour ″& i*100&″tirages l'écart est de  ″ & e)
Next
End Sub
```