Programmation structurée en Visual Basic Fonctions et procédures

Licence — Université Lille 1
Pour toutes remarques : Alexandre.Sedoglavic@univ-lille1.fr

Première année DEUST — 2008-09

Fonction

Une fonction est une forme d'instruction composée dont l'exécution dépends de paramètres :

```
Function Identificateur (DeclarationParametres) As TypeDeRetou
    Instructions
    Affectation d'Identificateur
    Exit Function
End Function
```

Par exemple, on peut calculer la somme de 2 entiers avec le code suivant :

```
Function Som(ByVal X As Integer, ByVal Y As Integer) As Intege
   Som = X+Y
   Exit Function
End Function
```

Le mot clef `Return`**.** Il est possible d'affecter la valeur de retour et d'interrompre l'exécution de la fonction grâce au mot clef `Return` :

```
Function Somme(ByVal X As Integer) As Integer
   Return X + 1
End Function
```

La fonction `InputBox`**.** Cette fonction permet de créer une boite de dialogue et de récupérer une chaîne de caractères.

```
Sub RecupereEtAffiche()
  Dim s As String
  s=InputBox("Saisissez votre message")
  MsgBox(s)
End Sub
```

On peut utiliser la fonction `Convert.ToInt32` pour convertir une chaîne en entier.

Procédure

Une procédure est une forme d'instruction composée dont l'exécution dépend de paramètres mais qui ne retourne rien (contrairement à une fonction) :

```
Sub Identificateur (DeclarationDeParametres)
    Instructions
    Exit Function
End Sub
```

Mode de passage

Le passage de paramètre peut se faire de deux façons différentes :

1. par valeur : c'est une copie qui est passée en paramètre à la fonction. Cette dernière peut manipuler la copie mais ne peut pas modifier les valeurs des variables de la routine appelantes ;
2. par référence : c'est une copie de l'adresse d'une variable définie dans l'espace d'adressage de la fonction appelante qui est passée en paramètre. Cette fois, la routine appelée peut modifier les variables de la routine appelantes.

Exemples

```
Function SquareFct( ByVal P As Integer ) As Integer
   P *=P      'le param\'etre s'utilise comme
   Return P   ' une variable usuelle
End Function

Sub SquareProc( ByRef Q As Integer )
    Q *=Q
End Function

Sub Main()
   Dim X,Y As Integer
   X=2
   Y=3
   X=SquareFct(X)
   SquareProc(Y)
End Sub
```

Attention, sans précision sur le type de passage :

```
Function SquareFct( P As Integer ) As Integer
```

VB le suppose du type ByRef !!!!!!!!!!!!!!!!!!!!!

La *porté* de la déclaration d'une variable est constituée par l'ensemble d'instructions pouvant manipuler cette variable. Dans l'exemple suivant :

```
Module essai
   Dim var1 As Integer

   Sub foo()
     Dim var2 As Integer
   End Sub

   Sub main()
     Dim var3 As Integer
   End Sub
End Module
```

- La variable `var1` est utilisable par toutes les routines définies dans le module.
- La variable `var2` n'est uniquement utilisable que dans la routine `foo`.
- La variable `var3` n'est uniquement utilisable que dans la routine `main`.

Récursivité

Une routine récursive est une routine qui contient un appel à elle-même

```
Function FactorielleRecursive (ByVal n As ULong) As ULong
  If n=0
  Then Return 1
  ElseIf n=1
  Then Return 1
  Else Return n*FactorielleRecursive(n-1)
End Function
```

Surcharge

La signature d'une routine est constituée d'un identificateur et d'une liste de paramètre — avec de plus d'une valeur de retour pour une fonction.

Il est possible d'utiliser plusieurs fois le même identificateur pour différentes signatures. Dans ce cas, cette identificateur est *surchargé*.

```
Overloads Function Square (ByVal X As Integer) As Integer
   return X*X
End Function

Overloads Function Square (ByVal X As Double) As Double
    return X*X
End Function
```

Lors de l'appel, la première version rencontrée de la fonction correspondant aux paramètres fournis est exécutée.