LES SOUS-PROGRAMMES ET LES STRUCTURES DE DONNÉES

Université Virtuelle de Côte d'Ivoire

Table des matières

Objectifs

Être capable de :

- Créer un sous-programme en Visual Basic.
- Créer des enregistrements en Visual Basic

LES SOUS-PROGRAMMES

Définition : DÉFINITION DES SOUS-PROGRAMMES

Les sous-programmes sont de programmes qui exécutent des tâches particulières et qui sont appelés dans le programme principal. Ainsi, parmi ces sous-programmes, l'on distingue les procédures des fonctions. Lors de la création d'un sous-programme, il peut admettre des arguments comme ne pas en avoir. Les sous-programmes peuvent se définir avant ou après le programme principal (celui défini par Main).

1- LA PROCÉDURE EN VISUALBASIC

Elle exécute une tâche particulière sans retourner de valeur. Lors de sa création, elle peut admettre des paramètres ou ne pas en avoir. La procédure se définit à l'aide du mot clé *sub* suivi du nom de la procédure avec éventuellement des arguments ou pas entre les parenthèses.

1. EXEMPLE DE PROCÉDURE SANS PARAMÈTRE

```
1  Module Module1
2
3      Sub Main()                'Début du programme principal
4          addition()            'appel de la procédure addition dans le
   programme principal
5          Console.Read()
6      End Sub
7      ' Création de la procédure addition()
8      Sub addition()
9          Dim val1 As Integer
10         Dim val2 As Integer
11         Dim val3 As Integer
12         Console.WriteLine("Entrez la première valeur Svp!")
13         val1 = Console.ReadLine()
14         Console.WriteLine("Entrez la seconde valeur")
15         val2 = Console.ReadLine
16         val3 = val1 + val2
17         Console.WriteLine("la somme de " & val1 & " et " & val2 & " donne "
     & val3)
18     End Sub
19 End Module
20
```

2. EXEMPLE DE PROCÉDURE AVEC DES PARAMÈTRES

Reprenons la même procédure en définissant deux arguments cette fois-ci.

```
1  Module Module1
```

```
    Sub addition(ByVal val1 As Integer, ByVal val2 As Integer)
        Dim val3 As Integer
        val3 = val1 + val2
        Console.WriteLine("La somme de " & val1 & " et " & val2 & " donne "
& val3)
    End Sub
    Sub Main()
        Dim a As Integer
        Dim b As Integer
        Console.WriteLine("Saisissez la première valeur svp!")
        a = Console.ReadLine()
        Console.WriteLine("saisissez la seconde valeur svp!")
        b = Console.ReadLine()
        addition(a, b)
        Console.Read()
    End Sub

End Module
```

NB : Dans le deuxième exemple

- La procédure addition est créée avec des paramètres val1 et val2, tous les deux de type ENTIER
- Lors de la définition des arguments, ByVal définit des paramètres par valeur (c'est à dire que le sous-programme n'a aucun impact sur leurs valeurs initiales) tandis que ByRef définit des paramètres par référence (c'est à dire que le sous-programme peut modifier les valeurs initiales des arguments)

LES FONCTIONS

Elle exécute une tâche particulière avec une valeur de retour. Lors de sa création, elle peut admettre des paramètres ou ne pas en avoir. La fonction se définit à l'aide du mot clé *function* suivi du nom de la fonction avec éventuellement des arguments ou pas entre les parenthèses.

EXEMPLE DE FONCTION SANS ARGUMENT

```
Module Module1

    Sub Main()              'Début du programme principal
        Dim nb As Integer
        nb = dble()          'appel de la fonction dble dans le programme
principal
        Console.WriteLine("le double de est " & nb)
        Console.Read()
    End Sub
    ' Création de la fonction dble()
    Function dble() As Integer
        Dim val1 As Integer
        Dim val2 As Integer
        Console.WriteLine("Entrez une nombre entier Svp!")
        val1 = Console.ReadLine()
        val2 = val1 * 2
        Return val2
    End Function
End Module
```

EXEMPLE DE FONCTION AVEC DES ARGUMENTS

```
Module Module1

    Sub Main()              'Début du programme principal
        Dim nb1, nb2, nb3 As Integer
        Console.WriteLine("Saisissez la longueur du rectangle svp!")
        nb1 = Console.ReadLine()
        Console.WriteLine("Saisissez la largeur du rectangle svp!")
        nb2 = Console.ReadLine()
        nb3 = surface(nb1, nb2) 'appel de la fonction surface() dans le
programme principal
        Console.WriteLine("La surface du rectangle dont la largeur est " &
nb1 & " et la longueur est " & nb2 & " donne " & nb3)
        Console.Read()
    End Sub
    ' Création de la fonction surface()
    Function surface(ByVal n1 As Integer, ByVal n2 As Integer) As Integer
        Dim val1 As Integer
        val1 = n1 * n2
        Return val1
    End Function
End Module
```

Remarque

Lors du passage des paramètres :

- Lorsque les arguments sont déclarés à l'aide du mot clé *ByVal* : il s'agit d'une définition de paramètres par valeur .La modification des paramètres dans le sous-programme n'a aucun impact sur leurs valeurs réelles.
- Lorsqu'il s'agit d'une définition de paramètres par référence, les arguments sont définis à l'aide du mot clé *ByRef*. La modification des paramètres dans le sous-programme entraîne une modification des valeurs réelles.

Pour illustrer tout ceci, écrivons un sous-programme qui retourne le carré d'un nombre entier.

EXEMPLE DE FONCTION AVEC DES PARAMÈTRES PAR VALEURS

```
Module Module1

    Sub Main()              'Début du programme principal
        Dim nb1 As Integer
        Console.WriteLine("Saisissez un nombre svp!")
        nb1 = Console.ReadLine()
        'appel de la fonction surface() dans le programme principal
        Console.WriteLine(carre(nb1) & " est le carré de " & nb1)
        Console.Read()
    End Sub
    ' Création de la fonction carre()
    Function carre(ByVal n As Integer) As Integer
        n = n * n
        Return n
    End Function
End Module
```

A l'appel de la fonction carre() dans le programme principal, si l'utilisateur saisi la valeur 2, à l'affichage, nous aurons "4 est le carré de 2", du fait des paramètres par valeur définis.

EXEMPLE DE FONCTION DÉFINIE AVEC DES PARAMÈTRES PAR RÉFÉRENCE

```
Module Module1

    Sub Main()              'Début du programme principal
        Dim nb1 As Integer
        Console.WriteLine("Saisissez un nombre svp!")
        nb1 = Console.ReadLine()
        'appel de la fonction surface() dans le programme principal
        Console.WriteLine(carre(nb1) & " est le carré de " & nb1)
        Console.Read()
    End Sub
    ' Création de la fonction carre()
    Function carre(ByRef n As Integer) As Integer
        n = n * n
        Return n
    End Function
End Module
```

A l'appel de la fonction carre() dans le programme principal, si l'utilisateur saisi la valeur 2, à l'affichage, nous aurons "4 est le carré de 4", du fait des paramètres par référence définis.

Exercice portant sur les sous-programmes

II

Exercice

[solution n°1 p.14]

Compléter le programme de sorte que cette fonction retourne le double d'un nombre entier saisi par l'utilisateur, en utilisant une assignation par référence

```
Module Module1

Sub Main()

Dim nb1 As 

Console.WriteLine("Saisissez un nombre svp!")

nb1 = Console.ReadLine()

Console.WriteLine( (nb1) & " est le double de "  nb1)

Console.Read()

End Sub

Function double( n As  ) As 

n = n 

 n

End 

End Module
```

LES STRUCTURES EN VISUAL BASIC

III

Définition

Une structure de données est un est type particulier de données permettant de contenir en son sein plusieurs propriétés ou variables.

1- CRÉATION ET MANIPULATION D'UNE STRUCTURE DE DONNÉES SIMPLE

Une structure de données se crée avant ou après le programme principal, à l'aide du mot clé *structure* suivi des différentes propriétés.

Exemple de création d'une structure de données.

```
Structure employe
        Dim mat As String
        Dim nom As String
        Dim pnom As String
        Dim cont As String
End Structure
```

Dans le programme principal, la structure de données s'exploite comme un type de données. c'est à dire qu'il faudra créer une variable et la déclarer comme étant de type structure_créée..

Exemple de manipulation de la structure *employe*

```
Sub Main()
        'Déclaration de la variable eemp de type employe (créé plus haut)
        Dim eemp As employe
        'Assignation de valeurs aux propriétés du type employe
        Console.Write("Saisie du nom:")
        eemp.mat = Console.ReadLine()
        Console.Write("Saisie du nom:")
        eemp.nom = Console.ReadLine()
        Console.Write("Saisie du prénom:")
        eemp.pnom = Console.ReadLine()
        Console.Write("Saisie du contact:")
        eemp.cont = Console.ReadLine()
        'AFFICHAGE des informations
  ------------------------------------------------------
        Console.WriteLine("MATRICULE: " & eemp.mat)
        Console.WriteLine("NOM: " & eemp.nom)
        Console.WriteLine("PRENOM: " & eemp.pnom)
        Console.WriteLine("CONTACT: " & eemp.cont)
```

```
        Console.ReadLine()
    End Sub
```

2- MANIPULATION D'UN TABLEAU DE STRUCTURE

Pour manipuler un tableau de structure, il faut au préalable avoir créé la structure de données. Ensuite créé le tableau de type la structure créée.

L'exemple que nous allons étudier à présent, va consister à effectuer plusieurs enregistrements de la structure *EMPLOYE* que nous avons créé.

```
Module Module1
    Structure employe
        Dim mat As String
        Dim nom As String
        Dim pnom As String
        Dim cont As String
    End Structure

    Sub Main()
        Dim eemp(3) As employe
        Dim i As Integer
        Console.WriteLine("_____________________________SAISIE DES
ENREGISTREMENTS_____________________________________")
        For i = 0 To 2
            Console.WriteLine("Enregistrement N° " & i + 1)
            Console.Write("Saisie du matricule: ")
            eemp(i).mat = Console.ReadLine()
            Console.Write("Saisie du nom: ")
            eemp(i).nom = Console.ReadLine()
            Console.Write("Saisie du prénom: ")
            eemp(i).pnom = Console.ReadLine()
            Console.Write("Saisie du contact: ")
            eemp(i).cont = Console.ReadLine()
            Console.WriteLine(
"_____________________________________________________________")
        Next
        'AFFICHAGE DES
INFOS-----------------------------------------------------
        Console.WriteLine()
        Console.WriteLine()
        Console.WriteLine("_____________________________AFFICHAGE DES
ENREGISTREMENTS_____________________________________")
        For i = 0 To 2
            Console.WriteLine("Enregistrement N° " & i + 1)
            Console.WriteLine("MATRICULE: " & eemp(i).mat)
            Console.WriteLine("NOM: " & eemp(i).nom)
            Console.WriteLine("PRENOM: " & eemp(i).pnom)
            Console.WriteLine("CONTACT: " & eemp(i).cont)
            Console.WriteLine(
"_____________________________________________________________")
        Next
        Console.ReadLine()
    End Sub

End Module

```

Dans ce code défini ci-dessus :

- Nous avons créé un enregistrement de type EMPLOYE
- Nous avons ensuite déclaré une variable eemp de type EMPLOYE
- Ensuite, nous avons procédé à un enregistrement de trois employés en nous servant de la structure itérative (FOR)
- Et pour terminer, nous avons affiché ces enregistrements

Exercice portant sur les enregistrements

IV

Exercice

[solution n°2 p.14]

Compléter le tableau de sorte qu'il effectue l'enregistrement de cinq étudiants et les affiche par la suite en tenant compte du matricule (matetud), du nom(nometud) et des prénoms(pnometud).

```
Module Module1
'Debut de la structure ----
                etudiant
Dim matetud As String
Dim              As String
Dim pnometud As String
'Fin de la structure-----
'Début du programme principal
Sub Main()
Dim eetud As etudiant
Console.WriteLine("Enregistrement ")
Console.Write("Saisie du matricule: ")
                = Console.ReadLine()
Console.Write("Saisie du nom: ")
                = Console.ReadLine()
Console.Write("Saisie du prénom: ")
                = Console.ReadLine()
Console.WriteLine("_____")
Console.WriteLine("_____AFFICHAGE DES ENREGISTREMENTS_____")
Console.WriteLine("MATRICULE: " &              )
Console.WriteLine("NOM: " &              )
Console.WriteLine("PRENOM: " &              )
Console.WriteLine("_____")
Console.Read()
End Sub
End Module
```

Solutions des exercices

> Solution n°1

Exercice p. 8

Compléter le programme de sorte que cette fonction retourne le double d'un nombre entier saisi par l'utilisateur, en utilisant une assignation par référence

```
Module Module1
Sub Main()
Dim nb1 As Integer
Console.WriteLine("Saisissez un nombre svp!")
nb1 = Console.ReadLine()
Console.WriteLine(double(nb1) & " est le double de " &nb1)
Console.Read()
End Sub
Function double(ByRef n As Integer) As Integer
n = n * 2
Return n
End Function
End Module
```

> Solution n°2

Exercice p. 13

Compléter le tableau de sorte qu'il effectue l'enregistrement de cinq étudiants et les affiche par la suite en tenant compte du matricule (matetud), du nom(nometud) et des prénoms(pnometud).

```
Module Module1
'Debut de la structure ----
Structure etudiant
Dim matetud As String
Dim nometud As String
Dim pnometud As String
```

```
End Structure
```

'Fin de la structure-----

'Début du programme principal

Sub Main()

Dim eetud As etudiant

Console.WriteLine("Enregistrement ")

Console.Write("Saisie du matricule: ")

`eetud.matetud` = Console.ReadLine()

Console.Write("Saisie du nom: ")

`eetud.nometud` = Console.ReadLine()

Console.Write("Saisie du prénom: ")

`eetud.pnometud` = Console.ReadLine()

Console.WriteLine("_____")

Console.WriteLine("_____AFFICHAGE DES ENREGISTREMENTS_____")

Console.WriteLine("MATRICULE: " & `eetud.matetud`)

Console.WriteLine("NOM: " & `eetud.nometud`)

Console.WriteLine("PRENOM: " & `eetud.pnometud`)

Console.WriteLine("_____")

Console.Read()

End Sub

End Module