

# Contrôle Continu 1 - UE47

16 novembre 2012

## 1 Portée et passage (6pts)

On exécute la macro *test* du code VBA suivant :

```
Dim x As Integer
Dim y As Integer

Sub test()
  Dim y As Integer
  y = 3: x = 2
  MsgBox "x=" & x & ", " & "y=" & y
  f
  MsgBox "x=" & x & ", " & "y=" & y
  g x, y
  MsgBox "x=" & x & ", " & "y=" & y
End Sub

Sub f()
  x = 5: y = 1
End Sub

Sub g(ByRef x As Integer, ByVal y As Integer)
  x = y: y = 4: x = x * y
End Sub
```

### 1.1 (1pt)

Qu'affiche le premier *MsgBox* ?

### 1.2 (2pts)

Qu'affiche le deuxième *MsgBox* ? Expliquez.

### 1.3 (3pts)

Qu'affiche le troisième *MsgBox* ? Expliquez.

## 2 Entrer un pourcentage (8pts)

Écrire une procédure qui demande à l'utilisateur de saisir un nombre (réel) avec *Application.InputBox*. Tant que ce nombre n'est pas compris entre 0 et 100, la procédure redemande un nombre. Quand le nombre saisi est bien compris entre 0 et 100, la procédure affiche une demande de confirmation avec *MsgBox*. Si l'utilisateur clique sur *OUI*, la procédure se termine. Si l'utilisateur clique sur *NON*, la procédure redemande un nombre.

Quelques tableaux :

Type	Type de données
0	Formule
1	Valeur numérique
2	Chaîne de caractères
4	Valeur booléenne
8	Référence de cellule
16	Valeur d'erreur

Buttons	Description
0	OK
1	OK et Annuler
2	Abandonner, Recommencer et Ignorer
3	Oui, Non et Annuler
4	Oui et Non
5	Réessayer et Annuler

Valeur de retour de <i>MsgBox</i>	Description
1	OK
2	Annuler
3	Abandonner
4	Recommencer
5	Ignorer
6	Oui
7	Non

## 3 Diviseurs (6pts)

### 3.1 (2 pts)

Écrire une fonction qui prend deux entiers  $x$  et  $y$  en paramètres et qui renvoie le booléen *True* si  $x$  divise  $y$  et *False* sinon. Pour cela, on peut utiliser la fonction infixe *Mod* qui est telle que  $x \text{ Mod } y$  donne le reste de la division euclidienne de  $x$  par  $y$ .

### 3.2 (4 pts)

Écrire une procédure qui récupère la valeur  $x$  contenu dans la cellule de la première ligne, première colonne et qui écrit dans la deuxième ligne de la feuille Excel les diviseurs de  $x$  dans l'ordre croissant. Ainsi, si on exécute cette procédure avec 12 dans la cellule (1,1) on obtient les valeurs 1, 2, 3, 4, 6, 12 dans la plage allant de la cellule (2,1) à la cellule (2,6).

On rappelle que pour accéder à la valeur de la cellule ligne  $i$ , colonne  $j$ , on écrit *Cells(i,j).Value*