Thème numéro 13

Points essentiels traités dans le thème :

1. **Enregistrement d'une macro VBA**
2. **Ecriture de code VBA**
3. **Utilisation des procédures de mise au point**

Cette séance reprend le modèle élaboré lors de l'étude du thème numéro 6 (tableau d'amortissement linéaire). Par conséquent, vous devez charger ledit modèle, et en vérifier les fonctionnalités, avant d'entreprendre la suite de la séance.

13.1. Enregistrement d'une macro

- assurez-vous que la feuille de calcul sur laquelle se trouve le tableau d'amortissement est bien protégée, sinon protégez-la
- Activez la rubrique **Développeur→Code→Enregistrer une macro (Developer→Code→Record Macro)**

Ou **Affichage→Macros→Macros→Enregistrer une macro (View→Macros→Macros→Record Macro)**

Vous voyez apparaître une boite de dialogue qui vous propose d'enregistrer une nouvelle macro.

- Introduisez **Encadrer** dans la zone de saisie **Nom de la macro**.
- Assurez-vous que le menu-déroulant « enregistrer la macro dans » **Ce Classeur** est bien activé.
- Validez votre choix en cliquant sur le bouton **Ok**
- Déprotégez la feuille
- Reprotégez-la
- Cliquez sur le bouton **Arrêter l'enregistrement** dans **Développeur→Code→Enregistrer une macro (ou Affichage→Macros→Macros→Enregistrer une macro)**
- Ouvrez ensuite l'Editeur Visual Basic, par exemple avec **[alt][F11]**

- Expansez le dossier **Modules**, en cliquant sur le signe [+] qui se trouve à sa gauche, puis ouvrez le module **Module1** par un double clic sur son icône. Vous voyez apparaître une série d'instructions Visual Basic :

```
Sub encadrer()
'
' encadrer Macro
'

'
  ActiveSheet.Unprotect
  ActiveSheet.Protect DrawingObjects:=True, Contents:=True,Scenarios:=True
End Sub
```

13.2. Ecriture d'instructions VBA

Ajoutez juste après **Sub Encadrer()** les trois lignes suivantes. Pour ce faire :

- positionnez le curseur au début de la ligne qui contient **ActiveSheet.Unprotect**
- frappez **[entrée]**
- frappez la flèche vers le haut
- insérez une tabulation
- Introduisez ensuite vos trois lignes, en frappant <Entrée> à la fin de chacune d'entre elles.

Les trois lignes à introduire se trouvent en page suivante

♦ `Dim i As Integer`

♦ `Dim csg As Range`

♦ `Dim cid As Range`

Ces trois lignes déclarent trois variables dont une est de type ***entier (Integer)*** et les deux autres de type ***plage (Range)***.

- Ajoutez sous la ligne **ActiveSheet.Unprotect** les lignes suivantes :

♦ `'éliminer l'ancien cadre`

*Cette ligne, qui commence par une apostrophe, est un **commentaire** destiné à faciliter la compréhension du programme par toute personne appelée à le lire*

♦ `ActiveSheet.Cells.Borders.LineStyle = xlNone`

*Cette ligne affecte la valeur prédéfinie **xlNone** à la propriété **LineStyle** des bordures de toutes les cellules de la feuille active. Ceci efface les éventuelles bordures préexistantes*

⇒**Cells** est une **méthode** de la classe **WorkSheet** à laquelle appartiendra l'objet **ActiveSheet** lors de l'exécution de la macro. Cette méthode, appelée sans argument, renvoie l'ensemble des cellules de la feuille, sous la forme d'un objet de classe **Range** (plage de cellules)

⇒**Borders** est une méthode de la classe **Range**, qui renvoie l'ensemble des bordures de la plage, sous la forme d'un objet de classe **Borders**, qui est une classe conteneur pour les objets de classe **Border**.

⇒**LineStyle** est une propriété de la classe **Border**, et aussi de la classe **Borders**

⇒La valeur **xlNone** est une des valeurs que peut prendre la propriété **LineStyle**, et signifie qu'aucun trait ne figure dans la bordure.

```
♦ 'trouver l'étendue du bloc utile
```

Il s'agit à nouveau d'un commentaire

```
♦ i = 10
```

dans la boucle qui suit, i sera le numéro de la ligne à laquelle on s'intéresse

```
♦ While ActiveSheet.Cells(i, 1).Value >= 0
♦    i = i + 1
♦ Wend
```

⇒La méthode **Cells** de la classe **WorkSheet**, lorsqu'on l'appelle avec deux arguments numériques **L** et **C**, dans cet ordre, renvoie un objet de classe **plage** qui contient l'unique cellule située à l'intersection de la ligne de numéro **L** et de la colonne de numéro **C** de l'objet de classe **WorkSheet** en cause, ici **ActiveSheet**.

⇒**Value** est une propriété de la classe **Range**. La lecture de cette propriété renvoie une valeur d'un type simple, comme ici, seulement si l'objet **Range** en cause ne contient qu'une seule cellule.

*La boucle **While** qui précède a donc pour effet d'augmenter l'indice de ligne **i** jusqu'à lui faire atteindre le numéro de la ligne où se trouve, dans la première colonne, une cellule dont la valeur est négative. Rappelez-vous que les formules utilisées dans cette feuille attribuent aux cellules de la colonne A une valeur négative sr les lignes inutilisées du tableau.*

♦ `i = i - 1`

Cette instruction fait pointer i sur la dernière ligne utile du tableau

♦ `Set csg = ActiveSheet.Cells (9, 1)`

*La variable csg (**C**oin **S**upérieur **G**auche) désigne à présent l'objet **Range** constitué de l'unique cellule située à l'intersection de la première colonne et de la 9ième ligne*

♦ `Set cid = ActiveSheet.Cells (I, 5)`

*La variable cid (**C**oin **I**nférieur **D**roit) désigne à présent l'objet **Range** constitué de l'unique cellule située à l'intersection de la cinquième colonne et de la **i**ième ligne*

♦ `ActiveSheet.Range(csg, cid).Select`

⇒La méthode **Range** de la classe **WorkSheet**, appelée comme ici avec deux arguments de classe **Range** constitués d'une seule cellule, renvoie un objet **Range** qui contient l'ensemble des cellules du bloc délimité par les deux cellules des objets **Range** passés en arguments

⇒La méthode **Sélect** de la classe **Range** provoque la sélection de l'ensemble des cellules de la plage.

A présent, la partie utile du tableau est sélectionnée. Nous pouvons tracer notre cadre

- Introduisez encore les lignes suivantes :

♦ `With Selection.Borders`

♦ `    .Weight = xlThin`

♦ `    .ColorIndex = xlAutomatic`

♦ `End With`

⇒L'instruction **With...End With** permet de se placer dans le contexte d'un objet. Les quatre lignes qui précèdent sont équivalentes aux deux lignes qui suivent :

```
Selection.Borders.Weight = xlThin
Selection.Borders.ColorIndex = xlAutomatic
```

Ces deux lignes étant équivalentes aux quatre lignes qui précèdent, n'ont pas à âtre introduites

⇒La méthode **Borders** de la classe **Range** renvoie, comme nous l'avons vu plus haut, l'ensemble des bordures des cellules de la plage, sous la forme d'un objet de classe **Borders**.

⇒**Weight** est une propriété de la classe **Border** ainsi que de la classe **Borders**. Cette propriété fixe l'épaisseur du trait de bordure

⇒**ColorIndex** est également une propriété des classes **Border** et **Borders**. Elle fixe la couleur des traits de bordure.

A présent toutes les cellules de la partie utile du tableau sont bordées par un trait fin, de couleur standard. Il nous reste à encadrer l'ensemble d'un trait épais, et à dessiner un trait moyen sous la première ligne.

- Introduire encore la ligne qui suit :

♦ `Selection.BorderAround LineStyle:=xlDouble, Weight:=xlThick`

Attention : *ce qui précède doit être introduit sur une seule ligne. Il n'y a pas de point, mais un espace entre les mots* ***BorderAround*** *et* ***LineStyle****.*

⇒**BorderAround** est une méthode de la classe **Range** à laquelle on passe ici des **arguments nommés**, d'où la syntaxe un peu particulière de cette ligne. Cette méthode fixe les propriétés des traits de contour de la plage.

♦ `Selection.Rows(1).Borders(xlEdgeBottom).Weight = xlMedium`

⇒**Rows** est une méthode de la classe **Range**. Dans notre cas (appelée avec un argument numérique **L**), elle renvoie une sous-plage de la plage initiale, composée de la ligne numéro **L** de la plage initiale, ici la première ligne de la partie utile du tableau

⇒La méthode **Borders** de la classe **Range**, appelée comme ici avec un argument numérique (constante prédéfinie **xlEdgeBottom**), renvoie l'ensemble des bordures des cellules de la plage, positionnées en conformité avec l'argument, ici les bordures du bord inférieur des cellules de la plage.

⇒Nous avons déjà vu plus haut la propriété **Weight** de la classe **Border** (ou **Borders**).

⇒**xlMedium** est une constante prédéfinie pour l'épaisseur des traits.

♦ `ActiveSheet.Cells(3, 4).Activate`

*La méthode **Activate** de la classe **Range** a pour effet de rendre active la cellule nord-ouest de la plage en cause, laquelle est ici constituée uniquement de la cellule D3 (troisième ligne, quatrième colonne), cellule dans laquelle l'utilisateur commencera à introduire les données d'un nouveau bien à amortir. Ceci a pour effet secondaire de déselectionner la sélection en cours, rendant ainsi pleinement visible le superbe cadre que nous venons de tracer.*

La macrocommande est en principe prête à l'emploi. Si vous l'avez introduite sans faute de frappe, elle devrait fonctionner du premier coup. Nous allons néanmoins l'utiliser pour faire la démonstration du système de mise au point de VBA pour EXCEL

13.3. Mise au point

Remarque : *pour observer l'effet sur votre tableau EXCEL des instructions VBA, il est recommandé de réduire la taille de la fenêtre d'EXCEL et celle de l'éditeur Visual Basic, et de » placer ces fenêtres côte à côte.*

- Sélectionnez la feuille dans laquelle se trouve le tableau d'amortissement
- Activez la rubrique de menu **Développeur→Code→Macros (**ou **Affichage→Macros→Macros→Afficher les macros)**
- Sélectionner la macro **Encadrer**, qui doit à présent se trouver dans la liste
- Cliquez sur le bouton **Pas à pas** (**Step Into**)
- L'éditeur Visual Basic s'ouvre sur votre macro, avec l'instruction **Sub Encadrer** sur fond jaune
- Activez la rubrique **Affichage→Fenêtre espions** (**View→Watch Window**)
- Introduisez trois espions sur les expressions **i**, **csg.Address** et **cid.Address**. Pour ce faire :

- cliquez avec le bouton droit de la souris, n'importe où dans le volet **Espion**

Ceci fait apparaître un menu contextuel

- Cliquez sur la rubrique **Ajouter un espion**

Ceci provoque l'ouverture d'une boite de dialogue

- Introduisez l'expression à espionner dans la zone de saisie **Expression**

les autres zones de la boite de dialogue sont déjà garnies comme il convient.

- validez en cliquant sur le bouton **Ok**

Chaque appui sur la touche **[f8]** fait avancer d'une instruction la partie sur fond jaune, et les espions dans la fenêtre correspondante montrent les valeurs des expressions. Vous pourrez ainsi voir évoluer la valeur de **i** lors du passage dans la boucle **While**, et voir l'adresse des plages **csg** et **cid** lorsque vous serez passé sur les instructions **Set**

L'appui sur **<F8>** lorsque la ligne **End Sub** est sur fond jaune termine la procédure. Pour voir à nouveau votre tableau, il faut ou bien fermer l'éditeur VBA, ou bien cliquer sur le titre **Microsoft Excel…** dans la barre de tâches. Le tableau est à présent agrémenté d'un cadre aux dimensions correctes, et la feuille est à nouveau protégée.

Pour exécuter votre programme sans le mode pas-à-pas, il suffit de cliquer sur le bouton **Exécuter** de la boite de dialogue ouverte par l'activation de la rubrique de menu **Développeur→Code→Macros (**ou **Affichage→Macros→Afficher les macros)**.

13.4. Création d'un bouton pour lancer la procédure

- **Enlevez la protection de la feuille de calcul**
- Activer la rubrique **Développeur→Controles→Insérer (Developer→controls→Insert)**

- Cliquez sur le bouton **bouton (contrôle de formulaire)**

- Lorsque vous ramenez votre curseur de souris sur la feuille de calcul, il prend la forme d'une croix en trait fin (***Crosshair***). Utilisez-le pour délimiter votre bouton.
    - Amenez la croix sur l'angle d'une cellule inoccupée
    - enfoncez le bouton gauche de la souris
    - en maintenant le bouton enfoncé, déplacer la souris pour étendre le cadre en trait tireté qui apparaît jusqu'à ce qu'il couvre la cellule.
    - Relâchez alors le bouton de la souris

Une boite de dialogue s'ouvre, pour vous demander quelle macro vous voulez affecter au nouveau bouton.

- Sélectionnez le nom de votre procédure, **Encadrer**
- Validez en cliquant sur le bouton **Ok**

Il reste à affecter à ce bouton un autre texte que **Bouton 1**. Pour ce faire :

- double-cliquez sur le bouton
- Introduisez sur la face du bouton, comme dans une zone de texte ordinaire, le mot **Encadrer**
- Cliquez avec la souris dans une autre cellule, quelconque, pour déselectionner le bouton

A présent, il vous suffira de cliquer sur ce bouton pour encadrer votre tableau, ou refaire un cadre adapté à une nouvelle durée d'amortissement.

- Sauvegardez définitivement ce classeur sous forme de modèle, éventuellement sous un nouveau nom

Annexe : texte du module Module1

```
Sub Encadrer()
'
' Encadrer Macro
'

'
  Dim i As Integer
  Dim csg As Range
  Dim cid As Range
  ActiveSheet.Unprotect
  ActiveSheet.Cells.Borders.LineStyle = xlNone
  'trouver l'étendue du bloc utile
  i = 10
  While ActiveSheet.Cells(i, 1).Value >= 0
      i = i + 1
  Wend
  i = i - 1
  Set csg = ActiveSheet.Cells(9, 1)
  Set cid = ActiveSheet.Cells(i, 5)
  ActiveSheet.Range(csg, cid).Select
  With Selection.Borders
      .Weight = xlThin
      .ColorIndex = xlAutomatic
  End With
  Selection.BorderAround LineStyle:=xlDouble, Weight:=xlThick
  Selection.Rows(1).Borders(xlEdgeBottom).Weight = xlMedium
  ActiveSheet.Cells(3, 4).Activate
  ActiveSheet.Protect DrawingObjects:=True, Contents:=True, Scenarios:=True
End Sub
```