Thème numéro 11

Points essentiels traités dans le thème :

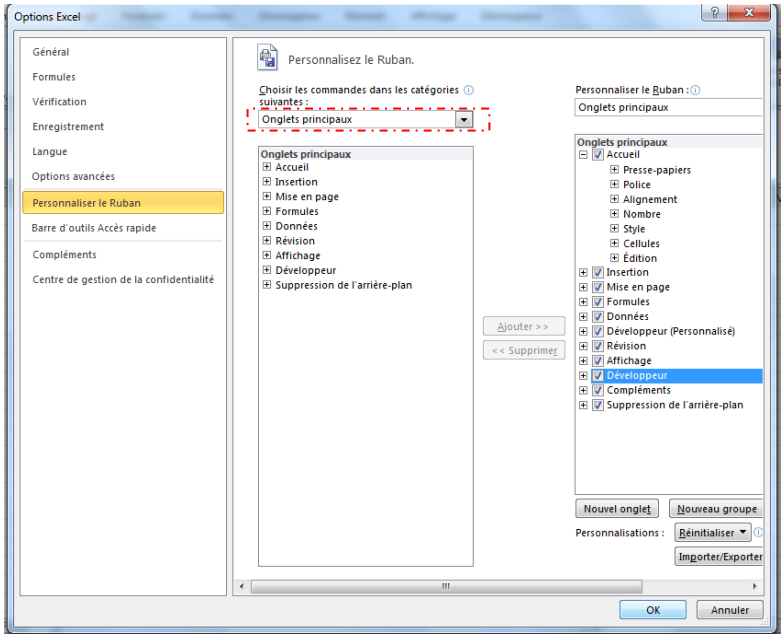
1. **Ecriture de code VBA**
2. **Utilisation de la fonction InputBox et de la procédure MsgBox**
3. **Instructions de gestion des boucles**
4. **Instructions de reprise sur erreur**
5. **Utilisation des procédures de mise au point**

11.1. Ecriture de code Visual Basic for Applications

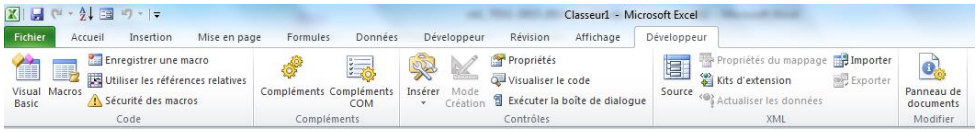
- Lancez Excel avec un classeur vide
- Faire apparaitre l'onglet du ruban dédié à VBA (s'il n'apparait pas)

Onglet **Fichier→Options**[1]**→**onglet **Personnaliser le Ruban**

Choisir **"Onglets Principaux",** Cocher **« Développeur »,** puis **OK**



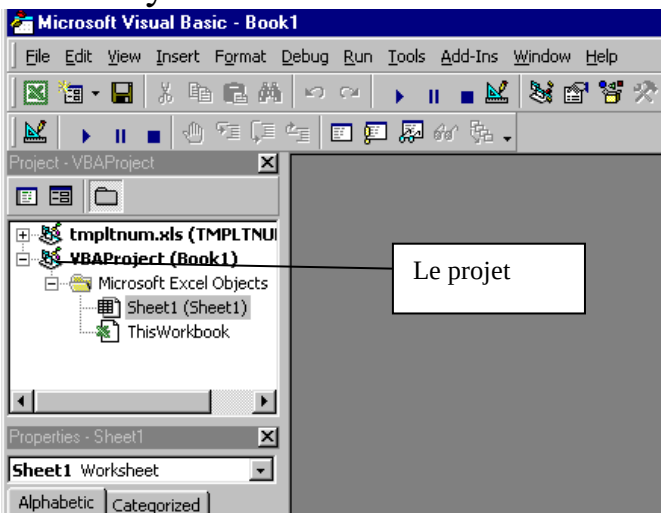
Le ruban **Développeur** apparait.



[1] *« Options» se trouve en bas à droite de l'onglet « Fichier ».*

- Activer l'éditeur Visual Basic, en activant la rubrique : **Développeur→Code→Visual Basic** ou en frappant **[Alt][F11]**

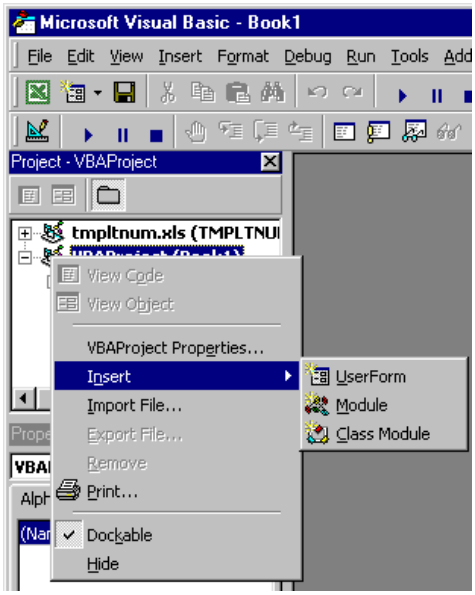
Vous voyez s'ouvrir une fenêtre avec l'éditeur Visual Basic :



Vous devez voir, à gauche et en haut, l'***explorateur de projets*** ; à gauche, et en bas, la ***fenêtre des propriétés***. Si ces fenêtres n'apparaissent pas, utilisez la rubrique de menus **Affichage→Barres d'outils (View→Toolbars)** pour les faire apparaître

Insérez dans le classeur un ***Module***, par la méthode qui suit :

- Sélectionnez dans le volet supérieur gauche le projet.
- Ouvrez son menu contextuel, par un clic droit.
- Positionnez la souris sur le choix **insérer** (**Insert**), ce qui ouvre un sous-menu.
- Activer le choix **Module** dans ce sous-menu.

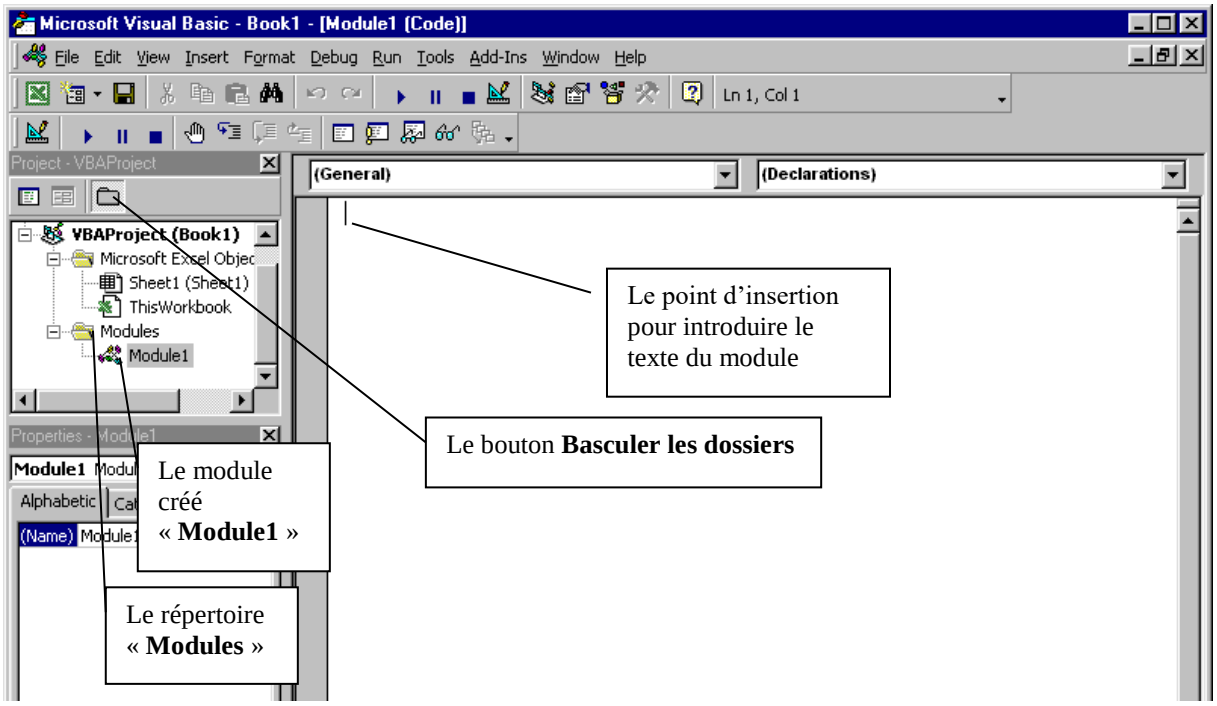


L'insertion d'un module dans un classeur : on voit ici le menu contextuel ouvert par un clic droit sur l'icône du projet lié au classeur.

Vous voyez apparaître, subordonné au **projet**, un répertoire nommé **Modules** qui contient à présent un module **Module1**. Si ce

répertoire n'apparaît pas, cliquez une fois sur le bouton **Basculer les dossiers (Toggle Folders)**

En même temps, le volet droit de la fenêtre se prépare à recevoir les instructions VBA que vous allez introduire, un point d'insertion clignotant y figure :



- Introduisez dans le module que vous venez de créer les lignes qui suivent (précédées par un losange), que nous commenterons à la volée (ne tentez pas d'introduire le losange, il n'est là que pour vous signaler les lignes de code)

```
♦ Sub factorielle()
♦     Dim ok As Boolean
♦     Dim chainesaisie As String
♦     Dim nombresaisi As Integer
♦     Dim resultat As Long
```

La première ligne est la déclaration d'une procédure sans paramètre.

Les lignes qui suivent déclarent, dans cette procédure, quatre variables

- une variable booléenne ***ok***
- une variable de chaine ***chainesaisie***
- une variable entière ***nombresaisi***

- une variable de type entier long, ***resultat***

♦ `Do`

Cette ligne commence une boucle qui se terminera peu avant la fin de la procédure. Cette boucle a pour but de s'assurer que l'utilisateur saisit un nombre dont on puisse calculer la factorielle.

♦ `ok = False`

Cette ligne initialise à l'intérieur de la boucle la variable ***ok*** à **False (Faux)**. Nous affecterons la valeur **True (Vrai)** à cette variable lorsque nous aurons décelé que la valeur saisie est correcte.

```
♦ chainesaisie = InputBox( _
♦     "introduisez un entier", _
♦     "Calcul de factorielle", _
♦     chainesaisie)
```

Ces quatre lignes constituent une seule instruction. Vous pouvez en effet constater que les trois premières se terminent par le caractère de continuation de ligne **" _ "**, précédé d'un espace.

Cette instruction est l'appel de la fonction ***InputBox.*** Cette fonction sert à la saisie d'un texte par l'intermédiaire d'une ***boite de saisie*** standard d'EXCEL.

⇒Le premier argument est un texte qui s'affichera dans la boite pour guider l'utilisateur sur ce qu'il doit faire.

⇒le second argument est le *titre* de la boite de saisie. Il s'affichera dans la barre de titre de celle-ci.

⇒Le troisième argument est la valeur par défaut du texte saisi. Il s'affichera dans la zone de saisie de la boite, et l'utilisateur pourra le modifier à volonté avant de valider sa saisie par **[entrée]** ou en cliquant sur le bouton **Ok** de la boite. Cette valeur par défaut est ici l'ancienne valeur de la variable ***chainesaisie***.

⇒Le résultat de la fonction est le texte saisi par l'utilisateur dans la zone de saisie de la boite. Il est ici rangé dans la variable ***chainesaisie***, dont il viendra remplacer l'ancienne valeur.

Nous allons à présent vérifier que l'utilisateur a bien introduit un nombre

♦ `If chainesaisie = "" Then GoTo fini`

si l'opérateur clique sur le bouton **annuler** ou presse la touche **[Echap]**, la fonction ***InputBox*** renvoie une chaîne vide. On désire dans ce cas sortir de la procédure. L'étiquette ***fini*** se situe juste avant la fin de celle-ci.

♦ `If Not IsNumeric(chainesaisie) _`

♦ `Then GoTo errnn`

⇒L'étiquette ***errnn*** est placée plus bas, elle précède une instruction qui affiche un message d'erreur.

⇒la fonction ***IsNumeric*** rend **True (Vrai)** si son paramètre (de type ***chaîne***) est entièrement numérique, c'est-à-dire entièrement constitué de caractères numériques.

⇒L'opérateur logique ***Not*** transforme **True** en **False** et **False** en **True**, ce qui fait que, si la chaîne saisie n'est pas numérique :

⇒***IsNumeric*** rend **Faux**

⇒***Not*** transforme ce **False** en **True**

⇒la ***condition*** *du* **If** vaut **True**, et le branchement ***GoTo errnn*** s'exécute, provoquant l'affichage du message d'erreur.

Bien que numérique, il est encore possible que notre chaîne saisie ne puisse être convertie en un ***entier*** (**123456** est numérique, mais trop grand pour un ***entier***). C'est pourquoi nous allons prévoir ce cas.

♦ `On Error GoTo errne`

Cette instruction avertit le système de traitement des erreurs de VBA qu'une erreur d'exécution, dans les lignes qui suivent, doit, au lieu de provoquer un arrêt prématuré du programme, provoquer un branchement à l'étiquette ***errne*** (erreur, non entier). Cette étiquette est définie plus bas.

Nous pouvons à présent tenter de transformer notre chaîne en entier.

```
♦          nombresaisi = CInt(chainesaisie)
```

La fonction ***CInt*** convertit une chaîne en un entier. Elle interprète la chaîne comme la représentation décimale de l'entier. Pour qu'aucune erreur de conversion ne se produise, la chaîne doit être constituée uniquement de caractères numériques, et la valeur après conversion ne doit pas excéder la valeur maximale d'un ***entier***.

```
♦          resultat = 1
```

Cette instruction initialise la variable ***résultat***, dans laquelle nous allons essayer de calculer la factorielle du nombre introduit.

Le seul problème qui puisse encore nous arriver est un dépassement de capacité dans la multiplication. C'est pourquoi nous prévenons à nouveau le système de gestion des erreurs, de manière à provoquer en cas d'erreur dans ce qui suit un branchement à l'étiquette ***errtg*** (erreur, trop grand) :

```
♦          On Error GoTo errtg
```

A présent commence le calcul de la factorielle proprement dit, constitué par une boucle ***While*** :

```
♦          While nombresaisi > 1
♦              resultat = resultat * nombresaisi
♦              nombresaisi = nombresaisi - 1
♦          Wend
```

Si le programme arrive à la fin de cette boucle, sans avoir provoqué d'erreur, c'est que la factorielle du nombre introduit a pu être calculée. Le nombre introduit était donc correct, nous pouvons positionner à **True** (**Vrai)** la variable ***ok***.

```
♦           ok = True
```

L'instruction qui suit permet, lorsque le calcul s'est bien terminé, de sauter les instructions qui affichent les messages d'erreurs

```
♦           GoTo finboucle
```

Les quatre lignes qui suivent gèrent les erreurs survenues lors de la tentative de calcul de la factorielle.

```
♦ errtg:
♦          MsgBox "l'entier " + _
♦              chainesaisie + " est trop grand"
♦          Resume finboucle
```

⇒***errtg:*** est une *étiquette* à laquelle se réfère l'instruction **On Error GoTo errtg** située plus haut. Ne pas oublier le caractère [:] qui suit le nom de l'étiquette. Quelle que soit la position où vous placez ce texte dans la ligne, l'éditeur de modules d'EXCEL, après avoir reconnu le caractère **« : »** qui annonce une étiquette, le ramène en début de ligne.

⇒***MsgBox*** est une fonction dont le but est l'affichage d'une *boite de message* standard EXCEL. Le premier argument passé à cette fonction (ici le seul) est le texte du message à afficher. Ce texte est la concaténation de deux textes fixes et du texte contenu dans la variable ***chainesaisie.***

Remarque : *Ici la* ***fonction MsgBox*** *est appelée avec la* ***syntaxe d'un appel de procédure****, car on se désintéresse totalement de la valeur qu'elle renvoie.*

On remarquera l'utilisation du caractère de continuation ***espace souligné*** **(« _ »)** pour passer à la ligne dans la description de la chaîne à afficher (attention : la continuation de ligne est marquée par un espace souligné, précédé d'un espace ordinaire), et l'utilisation de l'opérateur [+] comme opérateur de concaténation de chaînes.

⇒L'instruction ***Resume*** est spécifique à la fin des séquences de traitement d'erreur. Dans le cas actuel, où elle est suivie du nom d'une étiquette, elle demande au système de gestion des erreurs de ne pas conserver trace de l'erreur qui est survenue, et de réaliser un branchement à l'étiquette précisée.

Les quatre lignes qui suivent, analogues aux quatre précédentes, gèrent le cas dans lequel la chaîne introduite ne peut pas être convertie en un entier.

```
♦ errne:
♦         MsgBox chainesaisie + _
♦             " n'est pas un entier"
♦         Resume finboucle
```

Les quatre lignes qui suivent traitent le cas d'une chaîne non numérique. **Il ne s'agit pas d'une séquence de reprise d'erreur d'exécution**, car celles-ci sont activées par l'instruction ***On Error GoTo ...***, alors qu'on atteint l'étiquette ***errnn:*** par le branchement ***GoTo*** de l'instruction ***If Not IsNumeric...***, c'est pourquoi les lignes qui suivent se terminent par une instruction ***GoTo*** et non par une instruction ***Resume***.

```
♦ errnn:
♦         MsgBox chainesaisie + _
♦             " n'est pas un nombre"
♦         GoTo finboucle
```

Cette instruction ***GoTo*** était d'ailleurs superflue, car l'étiquette de branchement correspondante se trouve juste en séquence. Si vous le désirez, vous pouvez vous passer de l'instruction ***GoTo*** qui précède.

```
♦ finboucle:
♦     Loop Until ok
```

Ceci termine la boucle, et provoque un bouclage vers l'instruction ***Do*** tant que ***ok*** vaut **False**.

```
♦     MsgBox "la factorielle de " + chainesaisie + _
♦         " est " + CStr(resultat)
```

⇒L'affichage du résultat utilise la fonction ***MsgBox***

⇒L'élaboration de la chaîne à afficher utilise la concaténation et la fonction ***CStr***

⇒La fonction ***CStr*** assure la conversion d'un nombre en une chaîne, qui en est la représentation décimale.

```
♦ fini:
♦ End Sub
```

La première de ces deux lignes est une étiquette utilisée pour sortir de la procédure dans le cas où l'utilisateur clique sur le bouton **Annuler** ou appuie sur la touche **[Echap]** au lieu d'introduire un nombre dans la boite de saisie.

La seconde de ces deux lignes termine la procédure.

Remarque : *l'instruction* ***End Sub*** *a été automatiquement introduite par l'éditeur de modules de Visual Basic lorsque vous avez tapé* ***[Retour]*** *à la fin de la ligne* ***Sub factorielle()****. Vous n'avez pas à*

l'introduire une nouvelle fois par vous-même, ce serait inutile et nuisible : le compilateur détecterait une instruction ***End Sub*** *surnuméraire, et répondrait par l'indication d'une erreur de compilation.*

La sauvegarde de ce code se fait avec l'extension **.xlsm** (*classeur Excel prenant en compte les macros*).

11.2. Mise au point en pas-à-pas

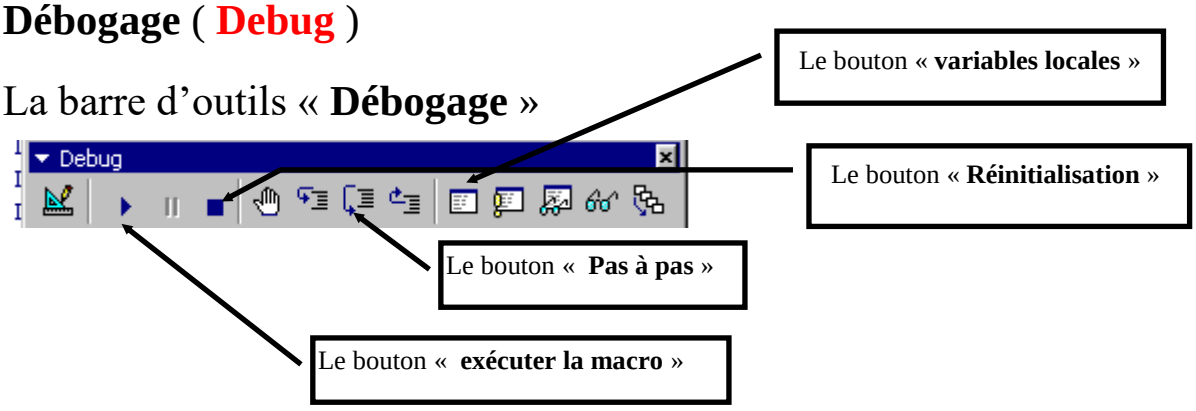
Même si votre procédure fonctionne du premier coup, sa mise au point en pas à pas vous permettra de mieux comprendre l'exécution des instructions qui la composent.

Avant de vous lancer dans ce qui suit, sauvegardez votre classeur*. Il est en effet possible qu'une erreur de saisie de votre programme vous entraîne dans une boucle sans fin, dont vous ne pourrez sortir qu'en redémarrant votre ordinateur. Si vous n'aviez pas encore sauvegardé le classeur à cet instant,* ***tout votre travail depuis le début de cette séance serait perdu.***

Remarque : *la combinaison de touches [Ctrl]-[Break] permet parfois de se sortir de ce genre de mauvais pas, en arrêtant une boucle sans fin.*

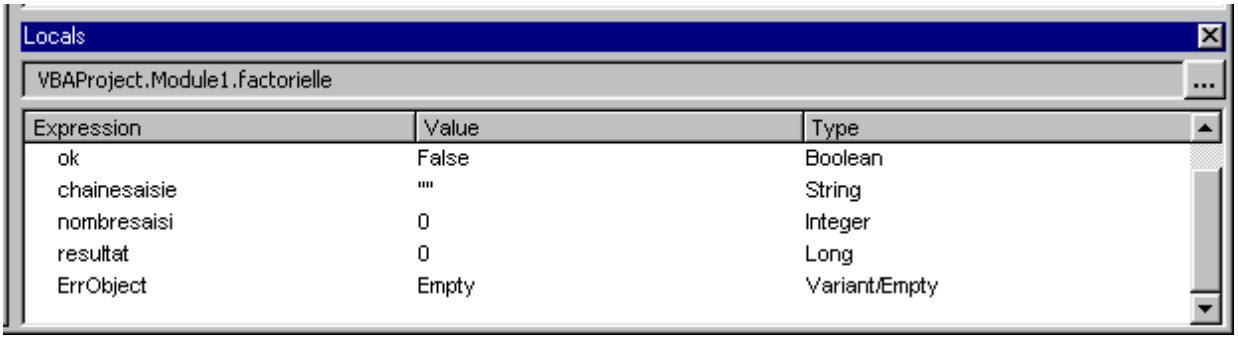
Alors que le module est affiché, et le curseur d'insertion à l'intérieur de la procédure, cliquez sur le bouton **pas à pas** de la barre d'outils **Débogage** (**Debug**)

La barre d'outils « **Débogage** »



Dans votre procédure, la première instruction s'affiche sur un fond jaune (ceci est la couleur par défaut, il serait possible de la changer).

Cliquez ensuite sur le bouton **Variables locales**. Une nouvelle fenêtre apparaît, en bas et à droite de l'écran, pour afficher les variables locales à la procédure :



En haut de cette fenêtre de variables locales s'affichent le nom du projet, celui du module et celui de la procédure. En dessous s'affichent les noms, les valeurs et les types des variables locales.

> Si des variables inattendues apparaissent, c'est que vous avez fait des fautes de frappe quelque part dans la procédure.
>
> Rectifiez ces fautes de frappe, puis introduisez en tête du module la commande **`Option Explicit`**. Ceci assurera la « remise à plat » de la liste des variables locales détectées par le système de débogage.

Faites à présent avancer le programme pas à pas, en appuyant sur la touche **[F8]** ou en cliquant à nouveau sur le bouton **Pas à pas**. A chaque appui sur cette touche, vous exécutez une instruction. Après l'exécution d'une instruction, le contenu de la fenêtre « **Variables Locales** » est mis à jour.

Lorsque vous aurez demandé l'exécution de l'instruction où se trouve l'appel à la fonction ***InputBox***, la boite de saisie s'ouvrira et vous pourrez y introduire une chaîne de caractères.

Commencez par y introduire une chaîne non numérique et validez par **[Entrée]** ou en cliquant sur le bouton **Ok**

Faites ensuite avancer le programme pas à pas.

La chaîne introduite n'étant pas numérique, l'instruction ***GoTo errnn*** s'exécute, puis l'appel de procédure ***MsgBox...*** qui se trouve derrière l'étiquette ***errnn:***. Le message **" XXXX n'est pas un nombre "** apparaît à l'écran.

Fermez la boite de dialogue en cliquant sur le bouton **Ok**, puis continuez l'exécution, toujours pas à pas.

Le programme exécute l'instruction ***GoTo finboucle****,* passe par ***Loop Until ok****, puis retourne en début de boucle, car* **ok** *vaut toujours* ***False****, comme vous pouvez le constater d'ailleurs dans la fenêtre « **Variables Locales** »*

Continuez en pas-à-pas jusqu'à ouvrir à nouveau la boite de saisie, puis introduisez une chaîne numérique trop grande pour être convertie en un entier (par exemple « **123456** »), et validez la.

Continuez encore d'avancer pas à pas.

Le programme passe le cap de l'instruction ***If Not IsNumeric...***, et, après la tentative de conversion par ***CInt***, se débranche à ***errne***. Il affiche le message **123456 n'est pas un entier**.

Fermez la boite de message en appuyant sur **[Entrée]** ou en cliquant sur le bouton **Ok**

Continuez d'avancer pas à pas, jusqu'à afficher à nouveau la boite de saisie. Introduisez dans la zone de saisie une chaîne qui puisse être convertie en un entier trop grand pour qu'on puisse en calculer la factorielle, soit par exemple **" 12345 "**

Continuez encore d'avancer pas à pas.

Vous constatez à présent que le programme passe le cap de la conversion de ***chainesaisie*** en entier, que le calcul commence bien, mais qu'après quelques tours de boucle le branchement à ***errtg*** se déclenche, et que le message **" l'entier 12345 est trop grand "** s'affiche (si toutefois vous n'avez pas oublié l'initialisation à 1 de la variable ***resultat***).

Fermez la boite de message, puis continuez d'avancer pas à pas jusqu'à afficher à nouveau la boite de saisie.

Introduisez enfin une chaîne qui puisse être convertie en un entier suffisamment petit pour qu'on puisse en calculer la factorielle dans un ***long***, par exemple **" 12 "** (c'est la plus grande valeur admissible), puis validez.

Faites progresser l'exécution pas à pas. Cette fois le programme doit arriver sans encombre à la fin de l'exécution de la boucle ***While***, en sortir, affecter à ***ok*** la valeur **True**, puis sortir de la boucle ***Do*** et afficher le résultat **" la factorielle de 12 est 479001600 "**, enfin se terminer.

Remarque : *Si à une étape quelconque de ce processus vous constatez une anomalie, réinitialisez le débogage en cliquant sur le bouton* **réinitialisation** *de la barre d'outils de débogage, puis rectifiez le texte de votre programme et* ***sauvegardez à nouveau le classeur*** *avant de reprendre la mise au point* ***depuis son début****. Seule cette démarche vous donnera la certitude d'avoir un programme correct.*

Si vous ouvrez à nouveau un classeur contenant des macros VBA, Excel affiche un « **Avertissement de sécurité** » vous précisant que « **Les macros sont désactivées** ». Pour les activer, après avoir cliqué sur le bouton « **options …** », cocher la case «**Activer ce contenu** » puis validez par **OK**.

Annexe : texte du module Module1

```
Sub factorielle()
    Dim ok As Boolean
    Dim chainesaisie As String
    Dim nombresaisi As Integer
    Dim resultat As Long
    Do
        ok = False
        chainesaisie = InputBox( _
            "introduisez un entier", _
            "Calcul de la factorielle", _
            chainesaisie)
        If chainesaisie = "" Then GoTo fini
        If Not IsNumeric(chainesaisie) _
            Then GoTo errnn
        On Error GoTo errne
        nombresaisi = CInt(chainesaisie)
        resultat = 1
        On Error GoTo errtg
        While nombresaisi > 1
            resultat = resultat * nombresaisi
            nombresaisi = nombresaisi - 1
        Wend
        ok = True
        GoTo finboucle
errtg:
    MsgBox "l'entier " + _
        chainesaisie + " est trop grand "
    Resume finboucle
errne:
    MsgBox chainesaisie + _
        " n'est pas un entier"
    Resume finboucle
errnn:
    MsgBox chainesaisie + _
        " n'est pas un nombre "
    GoTo finboucle
finboucle:
    Loop Until ok
    MsgBox "la factorielle de " + chainesaisie + _
        " est " + CStr(resultat)
fini:
End Sub
```

--- fin du thème ---