

Exercices : thème 2 - Question 3 (excel et vba)

Question 3 : La résolution de tous les problèmes de gestion est-elle automatisable ?

Cette activité a double vocation. D'une part, elle vise à apporter des éléments de réponse à la question 3 du programme. D'autre part, elle constitue une introduction à la programmation. Le langage choisi est VBA et l'outil de développement Excel. Afin de réaliser cette activité, il convient de se munir du support de cours associé : « [Cours Q3 - excel et vba.pdf](#) ».

Introduction : pourquoi et comment automatiser un problème de gestion ?

Qu'est-ce qu'un problème de gestion ?

Au travers de la question 1 du programme : "en quoi la qualité du système d'information est-elle un enjeu pour l'organisation ?", outre le fait que nous ayons défini et manipulé de nouveaux termes et concepts informatiques, nous avons en particulier constaté que les organisations, entreprises comme associations, devaient mettre en œuvre des procédures et processus afin d'assurer la gestion de leurs activités quotidiennes : processus de vente, de production, de relance client, de comptabilité, etc. En cela, au sein d'une organisation, l'on se heurte potentiellement à divers problèmes ou problématiques qu'il convient de résoudre en vue d'en assurer la bonne gestion.

Pourquoi automatiser un problème de gestion ?

Afin d'analyser, de comprendre voire de pouvoir améliorer les processus de gestion des organisations, nous avons découvert deux formalismes distincts permettant de les représenter : le schéma événements-résultats d'une part, le logigramme d'autre part. Finalement, nous avons entrevu la possibilité d'automatiser des processus simples au moyen d'un tableur tel qu'Excel faisant intervenir des formules, des données mais encore des validations de données.

Il est apparu de manière assez évidente que l'automatisation d'un processus, c'est-à-dire la création d'un outil informatisé, en l'occurrence un logiciel développé sous Excel, procurait plusieurs avantages :

- L'automatisation des calculs permet un gain de temps significatif et évite les erreurs que l'on pourrait commettre en effectuant les calculs manuellement.
- La structuration des données (sous forme tabulaire avec Excel ou encore sous forme XML) permet de représenter et conserver les données de manière structurée afin de pouvoir plus facilement les exploiter.
- La mise en œuvre de validation de données et de restrictions de saisie contribue en outre à la fiabilité des processus et à la qualité de l'information en ce qu'elle permet d'éviter les erreurs de saisie et de s'assurer de l'intégrité des données.
- L'automatisation permet en outre la standardisation des méthodes de travail ainsi que la standardisation et l'homogénéité des résultats ce qui peut constituer un gage de qualité.

Comment automatiser un problème de gestion ?

Vous l'aurez désormais compris, l'automatisation d'un processus de gestion passe en outre par l'analyse du processus et l'identification des informations qui lui sont nécessaires. Il convient ensuite d'établir une

solution. La réalisation d'une telle solution est en soi un projet. Or, au travers de la question 9 du programme : « en quoi un projet [...] est-il une réponse au besoin [...] de l'organisation ? », nous avons vu qu'un projet de SI se déroulait en plusieurs étapes.

Aujourd'hui, nous nous préoccupons d'une seule étape d'un projet informatique : le codage, c'est-à-dire l'automatisation à proprement parler. De fait, dans le cadre d'un projet logiciel, on retiendra bien qu'on ne se lance normalement pas « bille en tête »... Il faut avant toute chose définir la solution logicielle à concevoir : dessiner des maquettes, définir le fonctionnement du logiciel, etc. Ceci fait, on peut coder !

En quelque sorte, c'est le « code » et les algorithmes qui sont le fondement de l'automatisation. Coder, c'est, pourrait-on dire, parler à la machine pour qu'elle fasse ce qu'on lui a dit de faire. Au travers de cette activité, vous allez dire à Excel ce qu'il doit faire, au moyen du langage de programmation VBA.



Avertissement ! La présente activité est introductive. Elle vise à vous familiariser avec des concepts que nous réétudierons lorsque nous aborderons la programmation PHP. C'est un moment parfois difficile dans la vie du terminale STMG SIG... Mais un passage obligé pour tous. Alors, accrochez-vous et essayez de comprendre un maximum de choses !

Quelles sont les limites de l'automatisation sous Excel ?

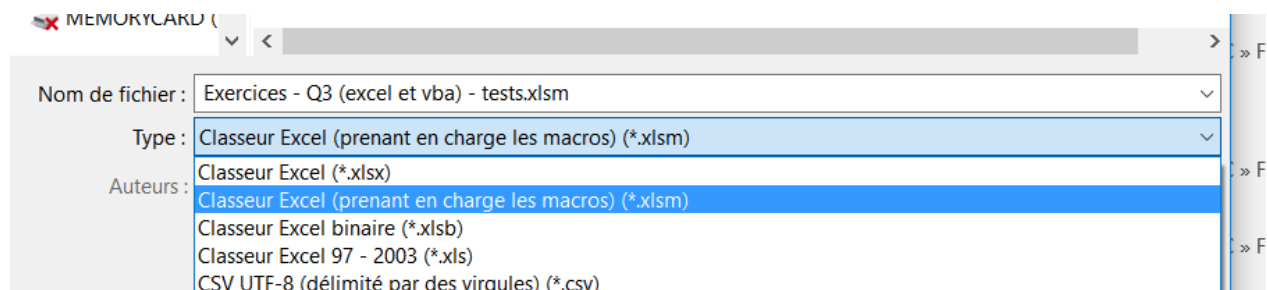
Une organisation est constituée et/ou en relation avec divers acteurs, internes et/ou externes. Ainsi, un processus est très vite amené à faire intervenir plusieurs voire de nombreux acteurs. C'est bien là la plus grande limite d'un simple tableur... Un logiciel tel qu'Excel est peu approprié au travail collaboratif et au partage d'informations. C'est pourquoi, une fois cette introduction à la programmation sous Excel terminée, notre attention se portera progressivement sur l'étude des réseaux puis sur l'étude de notions, concepts et solutions permettant le travail à plusieurs, à savoir le travail collaboratif.

Tests : programmons ensemble

Avant de « vous lâcher dans la nature », nous allons créer ensemble vos premiers programmes VBA. Ces premiers programmes vont consister tout simplement à reproduire et à comprendre les exemples du cours. A ce titre, voici quelques manipulations pratiques.

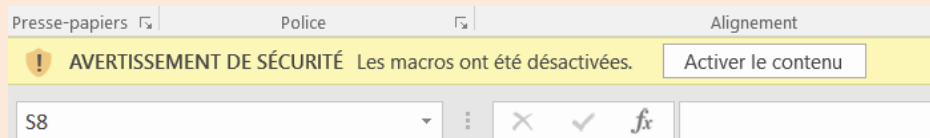
Enregistrer un classeur prenant en charges les macros :

- Ouvrir Excel
- Enregistrer le classeur en format Excel prenant en charge les macros (voir capture ci-dessous). Vous disposez désormais de votre fichier « .xlsm », lequel enregistre bien les macros (programmes VBA).



Avertissement !

- Par défaut (fichiers Excel « xls » ou «xlsx»), Excel n'enregistre pas les programmes VBA, c'est-à-dire les macros. Veillez à bien enregistrer au format « xlsx ».
- Quand vous ouvrez votre classeur « xlsx », pensez le cas échéant à activer les macros :



Afficher l'éditeur VBA et la boîte de contrôles :

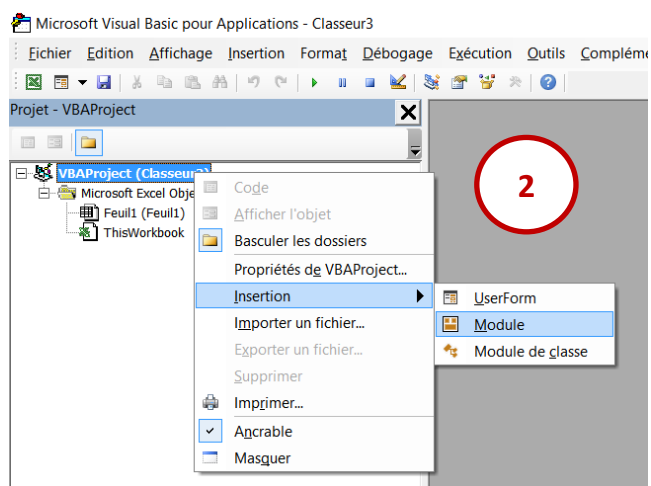
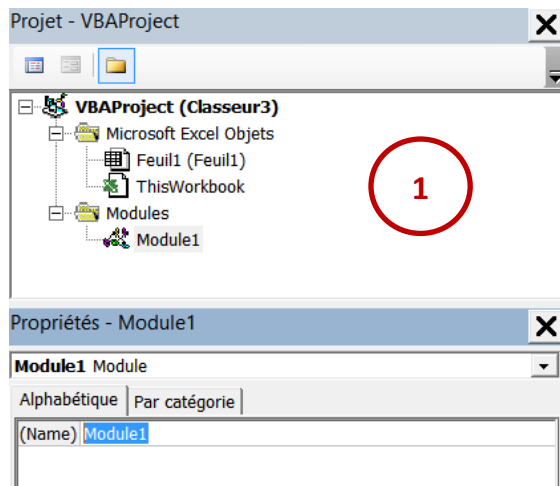
- L'éditeur VBA vous permet de rédiger vos programmes en VBA. La boîte de contrôles vous permet quant à elle d'ajouter des « contrôles » (champs de formulaire, boutons, etc.) à vos feuilles Excel.
- Versions récentes (Excel 2011, 2013 et 2016) : il suffit d'activer l'onglet « Développeur » pour qu'il apparaisse dans le ruban. Pour ce faire :
Clic droit sur le ruban > Personnaliser le ruban > Cocher « Développeur » dans les onglets principaux
- Versions anciennes (Excel 2007) : il faut également activer l'onglet « Développeur ». Pour ce faire :
Bouton office > Options Excel > Standard > Cocher « Afficher l'onglet Développeur dans le ruban »
- Versions archaïques (Excel 2003) : trifouiller...

Ouvrir l'éditeur VBA* :

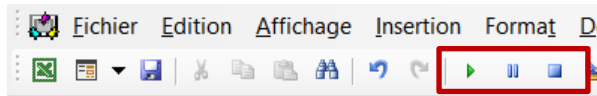
- Par le ruban : **Onglet Développeur > Visual Basic**
- Raccourci : **ALT + F11**
- Versions archaïques : **Outils > Macros > Editeur Visual Basic**

Créer et renommer un module :


- Un module est une sorte de fichier dans lequel le développeur peut écrire des programmes.
- Ouvrir l'éditeur VBA.
- Clic droit sur le projet > Insertion > module (voir capture 1 ci-dessous).
- Ouvrir le module (double clic).
- Changer la propriété *Name* du module afin de le renommer (voir capture 2 ci-dessous).



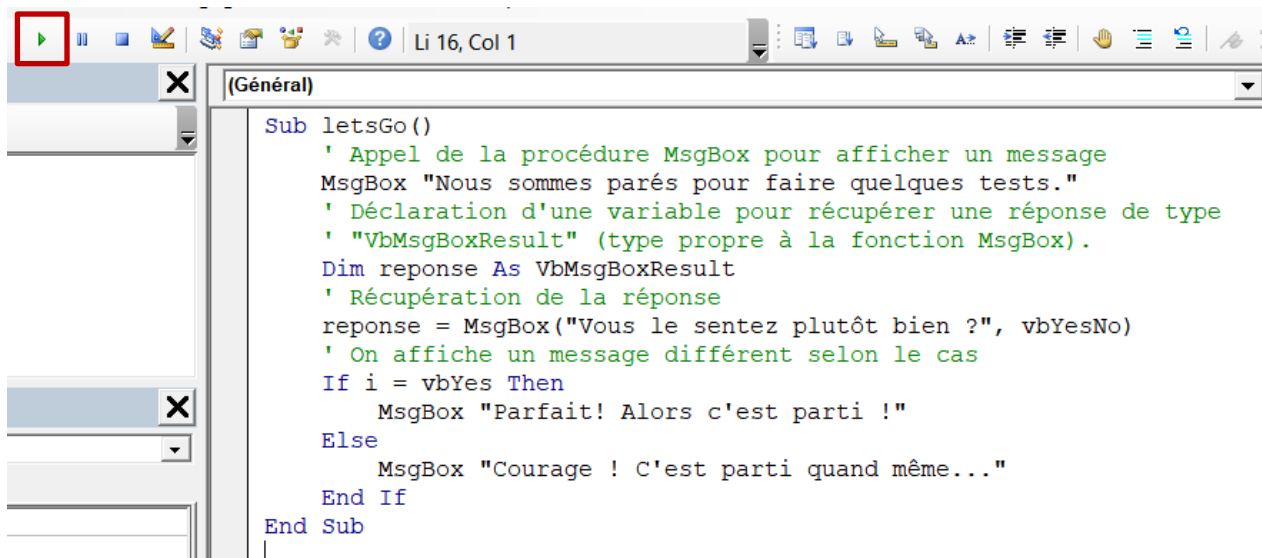
Exécuter un programme :



A gauche, les boutons encadrés vous permettent de démarrer un programme (une procédure « Sub ... ») ou encore de le stopper après qu'il a bogué.


 **Avertissement !** Si votre programme a complètement bogué (exemple classique : boucle infinie), vous serez contraint de redémarrer Excel. Pensez à bien enregistrer !

Pour conclure, un exemple de programme que nous pouvons tester : la procédure `Sub letsGo()`.



Travail à faire :

- Créer un classeur « Exercices - Q3 (excel et vba) - tests.xlsm » prenant en charge les macros.
- Créer un module « tests ».
- Reproduisons et discutons ensemble des exemples de programme du cours ainsi que de l'exemple figurant ci-dessus. Nous rédigerons ces tests dans notre module « tests ».

 **Conseil !** Au fil de toute cette activité, il est conseillé de commenter et de conserver vos codes sources : captures d'écran ou copier/coller du code. Cela peut amplement faciliter vos révisions et/ou relectures ultérieures.

Exercice 1 : un personnage haut en couleurs

Sujet : le résultat final de votre programme doit être similaire à celui présenté ci-dessous.

Vous aurez deux procédures à rédiger :

- La procédure `Sub dessiner_smiley()` permet de dessiner un smiley dont les couleurs sont saisies par l'utilisateur. Plus exactement, l'utilisateur peut saisir le code couleur des 4 parties du smiley. L'utilisateur ne peut saisir que des nombres entiers compris entre 0 et 255.
- La procédure `Sub effacer_smiley()` permet d'effacer le smiley qui a été dessiné.

Ces deux procédures (macros donc...) sont à affecter respectivement aux boutons « Dessiner le smiley » et « Effacer le smiley ». Vous rédigerez ces procédures au sein d'un module « smiley ». Vous travaillerez sur une feuille du classeur que vous appellerez « Smiley ».

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
1																
2													R	V	B	
3													Couleur du visage :	255	255	0
4													Couleur des yeux :	255	0	0
5													Couleur du nez :	0	255	0
6													Couleur de la bouche :	0	0	255
7																
8													Dessiner le smiley			
9													Effacer le smiley			
10																
11																

Travail à faire : rédiger le programme décrit ci-avant.

Vous procéderez comme suit :

- (1) Créer le classeur « Exercices - Q3 (excel et vba).xlsm ».
- (2) Créer la feuille « Smiley » puis le module « smiley ».
- (3) Reproduire le tableau et les deux boutons comme dans le modèle figurant ci-dessus.
- (4) Formater la plage de cellules M3:O6 afin d'afficher des nombres avec 0 chiffre après la virgule.
- (5) Ajouter une validation de données sur la plage de M3:O6 afin que l'utilisateur ne puisse saisir que des nombres entiers compris entre 0 et 255. Vérifier que la validation fonctionne convenablement.
- (6) Rédiger la procédure `Sub effacer_smiley()` permettant d'effacer la plage de cellules B2:J10. Vous pourrez utiliser la « commande » suivante : `Range("plage de cellule").Clear`. Affecter ensuite la macro au bouton « Effacer le smiley ». Vérifier le bon fonctionnement de cette fonctionnalité.

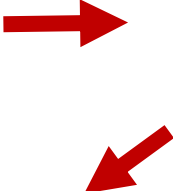
(7) Rédiger la procédure `Sub dessiner_smiley()` en plusieurs étapes, initialement sans tenir compte des couleurs saisies par l'utilisateur : colorier le visage, colorier les yeux, le nez et enfin la bouche. Inspirez-vous des tests précédemment effectués. Affecter la procédure au bouton « Dessiner le smiley ». Vérifier le bon fonctionnement de cette fonctionnalité.

(8) Adapter finalement votre procédure `Sub dessiner_smiley()` afin de tenir compte des codes couleurs saisis par l'utilisateur. Pour ce faire, vous pourrez par exemple procéder comme suit :

```
' Déclaration de variables pour
' récupérer les couleurs
Dim rouge As Integer
Dim vert As Integer
Dim bleu As Integer
Dim couleur As Long

' Récupération de la couleur du visage
rouge = Range("...").Value
vert = Range("...").Value
bleu = Range("...").Value
couleur = RGB(rouge, vert, bleu)

' Coloriage du visage
Range("...").Interior.Color = couleur
```



Exercice 2 : vendre plus pour gagner plus

Sujet : vous devez rédiger le calculateur de prime dont le modèle figure ci-dessous pour automatiser le calcul de prime des commerciaux d'une organisation. L'interface utilisateur (le visuel) est à placer dans une feuille nommée « Prime » du classeur « Exercices - Q3 (excel et vba).xlsm ».


Vous devez suivre les règles de gestion suivantes. Les commerciaux reçoivent une prime annuelle calculée en pourcentage du chiffre d'affaires HT (CA HT) qu'ils sont parvenus à réaliser. Le barème à appliquer est le suivant :

- Entre 0 et 100000 euros de chiffres d'affaires, le commercial ne touche aucune prime.
- Entre 100000 et 250000 euros de chiffres d'affaires, le commercial touche une prime de 10% sur le chiffre d'affaires au-delà de 100000 euros qu'il a réalisé.
- Au-delà de 250000 euros de chiffre d'affaires, le commercial touche en plus une prime de 20% sur le chiffre d'affaires qu'il a réalisé au-delà de 250000 euros.
- Un commercial ne peut faire un chiffre d'affaires inférieur à 0 euros.
- Un commercial est supposé ne pouvoir faire un chiffre d'affaire supérieur à 1000000 d'euros.

Vous aurez deux procédures à rédiger, dont une vous est fournie :

- La procédure `Sub dessiner_prime()` calcule et affiche la prime du commercial dans la cellule C10. Cette procédure est à rédiger dans un module « prime » du classeur.
- La procédure `Private Sub Worksheet_Change(ByVal plage As Range)` dont le code est fourni ci-après est à placer dans le code (via l'éditeur VBA) de la feuille « Prime » :

```
' Code s'exécutant lorsque le contenu d'une cellule de la feuille « Prime » est modifié
Private Sub Worksheet_Change(ByVal plage As Range)
    ' Lorsque c'est la cellule C6 qui a été modifiée, on recalcule et réaffiche la prime du commercial
    If plage.Address = "$C$6" Then
        dessiner_prime
    End If
End Sub
```

	A	B	C	D	E
1					
2		Calculateur de prime			
3		 Chiffre d'affaires HT réalisé par le commercial : <input type="text" value="125 000,00 €"/> Prime du commercial : <input type="text" value="2 500,00 €"/>			
4					
5					
6					
7					
8					
9					
10					
11					
12					

Travail à faire : rédiger le programme décrit ci-avant après avoir répondu aux 3 questions préliminaires.

Questions préliminaires :

- (1) Quel est le montant de la prime du commercial s'il a réalisé un CA HT de 75000€ ? Détailler le calcul.
[Votre réponse]
- (2) Quel est le montant de la prime du commercial s'il a réalisé un CA HT de 200000€ ? Détailler le calcul.
[Votre réponse]
- (3) Quel est le montant de la prime du commercial s'il a réalisé un CA HT de 300000€ ? Détailler le calcul.
[Votre réponse]

Vous procéderez comme suit :

- (1) Créer la feuille « Prime » puis le module « prime ».
- (2) Reproduire approximativement le modèle figurant ci-avant dans la feuille « Prime ». On ne s'attarde pas sur les détails graphiques mais on veille bien à utiliser les mêmes cellules que dans le modèle.
- (3) Formater les cellules C6 et C10 afin qu'elles affichent des « nombres monétaires ».
- (4) Ajouter une validation de données afin que l'utilisateur soit contraint de saisir un chiffre d'affaires conforme aux règles de gestion précédemment décrites. Vérifier le bon fonctionnement de cette fonctionnalité.
- (5) Créer la procédure `Sub dessiner_prime()` sans pour le moment la compléter.
- (6) Dans le code de la feuille « Prime » (voir dans l'éditeur VBA), copier/coller le code de la procédure `Private Sub Worksheet_Change(ByVal plage As Range)` fourni ci-avant.
- (7) Au moyen de la procédure `MsgBox` "Message à afficher", vérifier en affichant un simple message que la procédure `Sub dessiner_prime()` est bien appelée en cas de modification de la cellule C6, c'est-à-dire en cas de modification du CA HT réalisé par le commercial.
- (8) Au moyen d'une ou plusieurs variables et de conditions, compléter `Sub dessiner_prime()` afin qu'elle calcule et affiche le montant de la prime du commerciale. On rappelle que $10\% = 0.01$ et $20\% = 0.02$. Vérifier le bon fonctionnement de cette fonctionnalité.

Exercice 3 : do... do.. domino!

Sujet : vous devez rédiger un programme capable d'afficher un domino comme dans le modèle figurant ci-dessous.

Dans les cellules K3 et K5, l'utilisateur peut saisir les chiffres devant figurer sur le domino. Lorsqu'il clique sur le bouton « Dessiner le domino », le domino est effacé puis redessiné. Lorsque l'utilisateur clique sur le bouton « Effacer le domino », la partie haute puis la partie basse du domino sont effacées.

	A	B	C	D	E	F	G	H	I	J	K
1											
2										Afficher le domino n°	
3										Partie haute :	4
4											
5										Partie basse :	6
6											
7											
8										Dessiner le domino	
9											
10											
11										Effacer le domino	
12											
13											
14											
15											
16											

Travail à faire : rédiger le programme décrit ci-avant.

Pour réaliser ce programme, il va falloir réfléchir un peu et une fois encore bien faire les choses étape par étape :

- (1) Créer la feuille « Domino » et le module « domino ».
- (2) Reproduire le modèle figurant ci-dessus.
- (3) Créer les procédures `Sub dessiner_domino()` et `Sub effacer_domino()` puis les affecter aux boutons.
- (4) Rédiger la procédure `Sub effacer_domino()` qui permet d'effacer la partie haute puis la partie basse du domino.

(5) Compléter la procédure `Sub dessiner_domino()` afin qu'elle permette déjà d'afficher la partie haute du domino. Valider le bon fonctionnement de cette fonctionnalité.

(6) Compléter la procédure `Sub dessiner_domino()` afin qu'elle permette aussi d'afficher la partie basse du domino. Valider le bon fonctionnement du tout. Il risque de falloir pas mal de conditions au total...

Exercice 4 : on compte sur vous.

Sujet : vous devez rédiger plusieurs procédures de comptage afin d'essayer de vous familiariser avec la notion de boucle, soit en VBA le **For ... Next** (Pour) et le **Do While ... Loop** (Tant que). Les conditions et les boucles font parties des concepts les plus difficiles de votre programme de Terminale. Courage !

Visuellement, les procédures à rédiger doivent permettre d'obtenir les résultats suivants :

	A	B	C	D	E	F	G	H	I
1									
2		Compter de 1 en 1		Compter de 5 en 5		Compter de N en N		Compter jusqu'à N	
3									
4		Effacer		Effacer		Effacer		Effacer	
5									
6						N		N	
7						3		13	
8									
9		0		0		0		0	
10		1		5		3		1	
11		2		10		6		2	
12		3		15		9		3	
13		4		20		12		4	
14		5		25		15		5	
15		6		30		18		6	
16		7		35		21		7	
17		8		40		24		8	
18		9		45		27		9	
19		10		50		30		10	
20		11		55		33		11	
21		12		60		36		12	
22		13		65		39		13	
23		14		70		42			
24		15		75		45			
25		16		80		48			
26		17		85		51			
27		18		90		54			
28		19		95		57			
29		20		100		60			

Ce programme est constitué des 8 procédures suivantes :

- La procédure `Sub dessiner_compter_1_en_1()` permet de compter de 1 en 1 de 0 à 100 (ou jusqu'à 100). La procédure `Sub effacer_compter_1_en_1()` permet d'en effacer le résultat.
- La procédure `Sub dessiner_compter_5_en_5()` permet de compter de 5 en 5 jusqu'à 100. La procédure `Sub effacer_compter_5_en_5()` permet d'en effacer le résultat.
- La procédure `Sub dessiner_compter_n_en_n()` permet de compter de n en n. Elle affiche les 100 premiers nombres comptés. Le nombre n est saisi par l'utilisateur dans la cellule F7. La procédure `Sub effacer_compter_n_en_n()` permet d'en effacer le résultat.
- La procédure `Sub dessiner_compter_jusque_n()` permet de compter de 1 en 1 de 0 jusqu'à n, n étant saisi par l'utilisateur (cellule H7). La procédure `Sub effacer_compter_jusque_n()` en efface le résultat.

Pour n'efface que le texte, on utilisera plutôt la « commande » `Range("plage de cellules").clearContents`.

Travail à faire : créer une feuille « Compteur » semblable au modèle ci-dessus, créer un module « compteur » et rédiger successivement les 8 procédures décrites et illustrées ci-avant.

Exercice 5 : comme à l'école

Sujet : vous devez cette fois-ci rédiger un programme qui permettra d'afficher une table de multiplications. L'utilisateur choisit (cellule C2) la table de multiplications à afficher. Il peut saisir un nombre entier quelconque, mettons compris entre -100 et 100 par exemple. Vous devrez :

- Créer une feuille « Multiplication » et un module « multiplication ». Le feuille « Multiplication » devra être semblable au modèle figurant ci-dessous.
- La procédure `Sub dessiner_multiplication()` permet d'afficher la table de multiplication. Elle n'affiche que les 11 premiers multiples.
- La procédure `Sub effacer_multiplication()` permet d'effacer la table de multiplication.
- Ces deux procédures sont respectivement affectées aux boutons « Calculer » et « Effacer ».

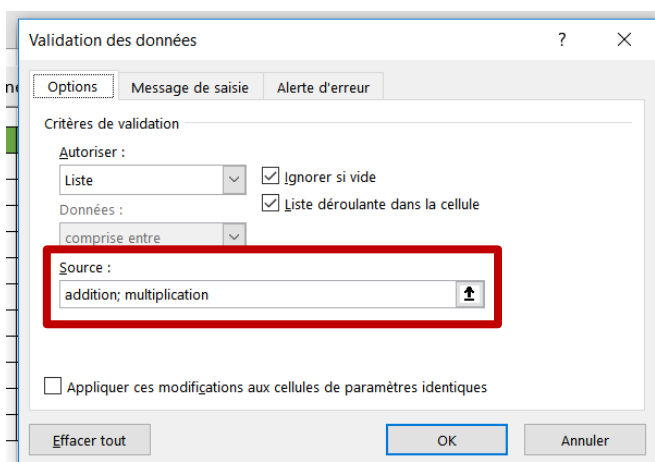
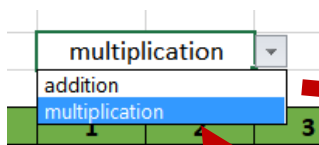
	A	B	C	D	E	F	G
1							
2		Table des :	27		Calculer	Effacer	
3							
4		27	x	0	=	0	
5		27	x	1	=	27	
6		27	x	2	=	54	
7		27	x	3	=	81	
8		27	x	4	=	108	
9		27	x	5	=	135	
10		27	x	6	=	162	
11		27	x	7	=	189	
12		27	x	8	=	216	
13		27	x	9	=	243	
14		27	x	10	=	270	
15							

Travail à faire : rédiger le programme ci-avant décrit.

Exercice 6 : Pythagore

Sujet : vous devez cette fois-ci rédiger un programme qui permettra d'afficher (de générer) une table de Pythagore, célèbre tableau dont vous avez peut-être encore le vague souvenir, lointain, d'un passé révolu en école primaire, époque où nous apprîmes tous nos tables de multiplications et d'additions, etc., etc., etc. Encore du calcul me direz-vous... Certes, mais du calcul simple toutefois. L'utilisateur peut choisir d'afficher soit la table des additions soit celles des multiplications.

Ci-dessous et à droite, le modèle à reproduire, feuille « Pythagore », ainsi que la validation de données à mettre en place.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2		Opération :		multiplication			Générer		Effacer					
3														
4			0	1	2	3	4	5	6	7	8	9	10	
5		0	0	0	0	0	0	0	0	0	0	0	0	
6		1	0	1	2	3	4	5	6	7	8	9	10	
7		2	0	2	4	6	8	10	12	14	16	18	20	
8		3	0	3	6	9	12	15	18	21	24	27	30	
9		4	0	4	8	12	16	20	24	28	32	36	40	
10		5	0	5	10	15	20	25	30	35	40	45	50	
11		6	0	6	12	18	24	30	36	42	48	54	60	
12		7	0	7	14	21	28	35	42	49	56	63	70	
13		8	0	8	16	24	32	40	48	56	64	72	80	
14		9	0	9	18	27	36	45	54	63	72	81	90	
15		10	0	10	20	30	40	50	60	70	80	90	100	
16														

Vous devrez cette fois-ci :

- Créer la feuille « Pythagore » et un module « pythagore ».
- Implémenter la procédure `Sub dessiner_pythagore()` permettant l'affichage de la table de Pythagore (plage C5:M15) des additions ou des multiplications selon ce que l'utilisateur à choisi.
- Les en-têtes (en vert) sont saisis manuellement.
- Songez qu'il vous faut parcourir les lignes mais aussi les colonnes du tableau. Cela fait donc deux boucles imbriquées !
- Il faut également développer la procédure `Sub effacer_pythagore()`, laquelle permet d'effacer le contenu de la table de Pythagore.

Travail à faire : rédiger le programme ci-avant décrit.

Exercice 7 : plus ou moins

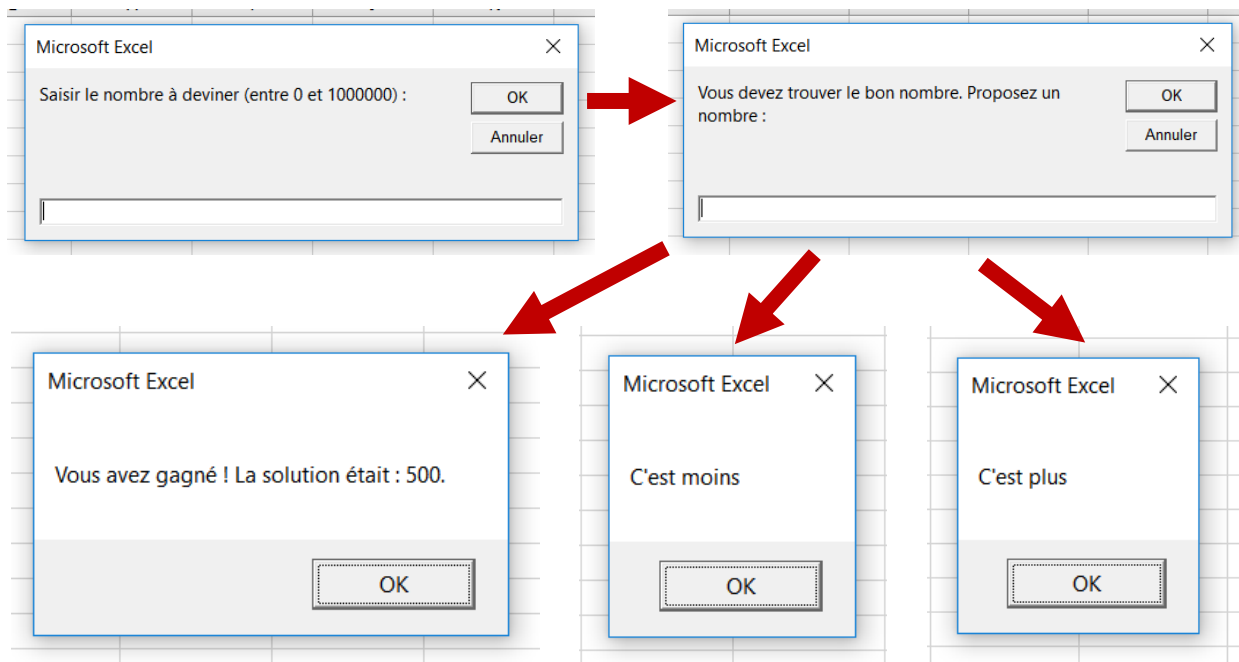
Sujet : il s'agit de développer le jeu du « plus ou moins » dont les interfaces sont décrites ci-après.

Le processus est le suivant :

- Le joueur 1 doit saisir un nombre (ici compris entre 0 et 1000000) que le joueur 2 doit deviner. On demande donc au joueur de « Saisir le nombre à deviner ». Pour ce faire, on pourra s'inspirer du fragment de code suivant :

```
' Déclaration d'un grand nombre entier.  
Dim nombreAttendu As Long  
' Ouverture d'une boîte de dialogue (comme les deux ci-dessous) et récupération du nombre saisi.  
nombreAttendu = InputBox("Message à afficher")
```

- Une fois le nombre à deviner saisi, le joueur 2 doit le deviner. Autrement dit, tant qu'il n'a pas trouvé le résultat, la partie continue ! On doit ainsi récupérer le nombre proposé par le joueur 2 et voir s'il correspond bien à celui à deviner.
- Dans le cas où le joueur 2 propose un nombre trop petit, on lui affiche le message « C'est plus ». Dans le cas contraire, on lui affiche « C'est moins ». On affiche ces messages encore et toujours grâce à la procédure `MsgBox "Message à afficher"`.
- Quand le joueur 2 a trouvé le bon nombre, la partie s'arrête. On affiche un message indiquant au joueur qu'il a gagné.



Pour réaliser le programme, toujours bien penser à procéder étape par étape et à bien vérifier au fur et à mesure que chaque partie de votre programme fonctionne bien (avec des `MsgBox` par exemple).

Travail à faire : créer le module « `plusOuMoins` » et rédiger le programme décrit ci-dessus au sein d'une procédure que vous appellerez `Sub jouer_plus_ou_moins()`.

Exercice 8 : escalier

Sujet : chers architectes, il vous faut à présent dessiner un escalier et compter le nombre de briques nécessaires à sa construction... Parviendrez-vous à gravir les échelons ? Il vous faudra deux boucles et surtout colorier les bonnes cases sans en perdre une vous-même !

Les règles de gestion sont les suivantes :

- L'escalier devra comporter autant de marches que l'utilisateur en a demandé (dans la cellule F3). L'utilisateur peut saisir un nombre compris entre 1 et 10. L'escalier ne fera donc ici que 10 marches au plus, 1 au moins.
- 1 cellule coloriée = 1 brique requise !

Vous devez :

- Créer une feuille « Escalier » et un module « escalier ».
- Rédiger la procédure `Sub dessiner_escalier()` qui est affectée au bouton « Dessiner l'escalier ». Cette procédure permet d'afficher l'escalier et le nombre de briques requises, c'est-à-dire le nombre de briques ayant été nécessaire à sa construction.
- Rédiger la procédure `Sub effacer_escalier()` qui permet d'effacer l'escalier et qui est affectée au bouton « Effacer l'escalier ».
- Traiter d'abord l'affichage de l'escalier sans vous préoccuper du nombre de briques. Ensuite seulement, ajuster votre algorithme pour qu'il effectue en même temps le comptage des briques et affiche à la fin le nombre de briques requises.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2														
3			Nombre de marches :			7		Nombre de briques requises :			28			
4														
5														
6														
7														
8														
9														
10														
11														
12														
13														
14														
15														
16														
17														
18														
19														
20														

Travail à faire : créer la feuille, le module et les deux procédures décrites ci-avant.