REPUBLIQUE DU SENEGAL

UNIVERSITE CHEIKH ANTA DIOP DE DAKAR

ECOLE SUPERIEURE POLYTECHNIQUE

CENTRE DE THIES

DEPARTEMENT DE GENIE ELECTROMECANIQUE

PROJET DE FIN D'ETUDES

En vue de l'obtention du Diplôme d'ingénieur de Conception

CREATION DE LOGICIEL DE PREVISION DE LA DEMANDE ET DE GESTION DES STOCKS

ELEVE INGENIEUR : Amadou Carter CAMARA

ENCADREURS : M. NGor SARR

M. Mamadou Salla GUEYE

Année Universitaire 2002 / 2003

DEDICACES

A mon pére , à ma mére

A mes fréres

Et à ma sœur.

REMERCIEMENTS

Grâce à DIEU, que nous remercions de nous avoir donner la santé pour que nous puissions écrire ce rapport, nos remerciements iront à l'endroit :

- De nos parents : père, mère, frères et sœur qui se sont toujours souciés de nos études.
- De Monsieur Mamadou GUEYE qui a bien voulu nous encadrer dans la réalisation du projet
- De Monsieur Ngor SARR
- De l'ensemble des professeurs dans la qualité de leur enseignement
- De l'ensemble des élèves ingénieurs de ma promotion, de leur esprit de solidarité et de leur dévouement
- De l'ensemble des étudiants de l'école supérieure polytechnique du centre de Thiès ainsi qu'à tous ceux qui ont contribué au bon déroulement de ce travail

Table des matiéres

Liste des figures 1

Sommaire 2

INTRODUCTION 3

CHAPITRE1 : Visual Basic Application 5

1 Introduction a Visual basic application 6

2 Automatisation des tâches répétitives 8

2.1 Introduction aux macros 8

2.2 Simplification de tâches à l'aide de macros 9

2.3 Quand faut-il enregistrer une macro ? 10

3 Création de fonctions personnalisées 15

3.1 Introduction 15

3.2 Rôle d'une fonction personnalisée 16

3.3 Les éléments d'une fonction personnalisée 17

3.4 Comprendre les éléments d'une fonction personnalisée 20

CHAPITRE 2: De la gestion à la programmation 23

1 De la Gestion de la production à la programmation 24

2 Enregistrement de macro 26

3 Exécution d'un macro 29

4 Création d'une fonction personnalisée 30

CHAPITRE 3: Logiciel de gestion de la production 32

1 Feuilles boite de dialogue Visual 33

1.1 Création de feuille boite de dialogue Visual Basic 33

1.2 Exécution d'une feuille boite de dialogue Visual Basic 34

2 Bouton de commande personnalisée 37

2.1 Définition des propriétés d'un contrôle *bouton de commande* 38

2.2 Affectation de l'ordre de tabulation des boutons 41

3 Gestion des erreurs et des valeurs d'erreur 41

CHAPITRE 4 : Avancement du projet 44

1 Les objectifs du projet 45

2 Avancement du projet 46

CONCLUSION ET RECOMMMANDATIONS 47

Annexes 49

LISTES DES FIGURES

Figure 1 :Macro d'exemple 12

Figure 2 :Exemple de fonction personnalisée 18

Figure 3 : Fonction personnalisée défectueuse 25

Figure 4 : Macro pour le calcul d'une *valeur future* 26

Figure 5 :Enregistrement d'un macro 29

Figure 6 :Mode création d'une feuille *boite de dialogue*. 34

Figure 7 :Classeur de calcul du logiciel 35

Figure 8 :Exécution du logiciel. 37

Figure 9 :Message d'erreur pour une mauvaise entrée de données 43

SOMMAIRE

Ce **logiciel de Gestion de la Production** que nous avons à présenter, est la suite logique d'un travail commencé depuis l'année dernière. En effet, depuis que nous avons fait le cours de gestion de la production, nous nous sommes sentis incapable de résister à un désir grandissant de créer un outil de gestion. Ainsi, notre Projet de Fin d'Etudes (PFE), aura sans doute était la concrétisation d'un rêve qui, tout au moins partiellement, méritait sa réalisation.

Le chapitre premier de ce rapport de PFE, présentera donc Visual Basic Application (logiciel de développement de notre travail), tout en insistant sur les macros et fonctions personnalisées.

Le chapitre deux parlera de la création de macros et fonctions personnalisées tout en décrivant le passage du cours de gestion à l'état logiciel.

Le chapitre trois décrira les boîtes de dialogue et leur création dans le logiciel et ainsi, introduire le chapitre quatre qui fera le bilan des objectifs par rapport à l'avancement du projet.

INTRODUCTION GENERALE

Le cours de gestion de la production que nous avons eu à suivre en deuxième DIC (Diplôme d'ingénieur de Conception) nous a beaucoup marqués. En effet depuis qu' on nous l'a dispensé nous n'avons cessé d'avoir l'œil critique en matière d'affaires. Dés lors, face à des problèmes de ruptures de stocks chez nos boutiquiers et pharmaciens, on a voulu savoir comment réagissait un commerçant sénégalais, face à un problème de gestion de la production

Ainsi, nous avons fait une enquête. Elle n'aura pas satisfait nos attentes. En effet, nos commerçants semblent ne pas maîtrisés la Gestion, bien qu'il méritent beaucoup de respect. Globalement, le commerçant sénégalais semble ne pas gérer avec des méthodes précises qui privilégient le profit maximum ; il est passé totalement à côté d'un problème qui consistait à calculer une Q_{opt} (quantité économique a commander).

Le schéma de la première page de l'annexe A décrit l'enjeu de la productivité dans un pays. Selon ce schéma, la baisse de la productivité des entreprises d'un pays pourrait impliquer une baisse du niveau de vie, une hausse de l'inflation, une baisse des exportations et donc une chute économique qui sera marquer par un taux de chômage grandissant.

Ainsi pour participer au développement du Sénégal nous avons pensé créer un ***logiciel de Gestion de la production*** pour améliorer la productivité des petites entreprises. En effet, il existe déjà un logiciel de Gestion de la productivité sur le marché. Cependant son coût reste élevé en même temps que le coût de l'ordinateur qui doit l'exploiter.

Ainsi, dans notre logiciel créer avec Microsoft Excel et Visual Basic, nous avons proposé deux grands chapitres qui sont :

- Calcul financier et calcul de productivité
- Prévision et gestion des stocks.

1 INTRODUCTION A VISUAL BASIC APPLICATION

Microsoft Visual Basic est un langage de programmation puissant et convivial de Microsoft Excel. Cependant par Visual Basic application, il faut entendre l'adaptation à Microsoft Excel de l'environnement de développement Microsoft Visual Basic, Edition Application plutard dans ce rapport vous allez voir que Visual Basic Application vous permet d'automatiser les tâches répétitives, d'ajouter certaines caractéristiques et fonctions personnalisées répondant à vos besoins spécifiques et même de créer des applications complètes.

Dans Microsoft Excel, vous automatisez l'exécution de tâches au moyen de Macros.

Une macro est constituée d'une série d'instructions qui commandent à Microsoft Excel d'exécuter certaines tâches bien définies. Ces instructions sont écrites en Visual Basic mais cela ne signifie pas que la création des macros soit réservée aux programmeurs. En réalité, vous n'avez même pas besoin de connaître Visual Basic pour utiliser les macros.

Le programme Microsoft est livré avec un Enregistreur de macros qui écrit les macros à votre place. Il enregistre les actions que vous exécutez

et les commandes que vous choisissez pendant que vous utilisez Microsoft Excel, ensuite, il vous permet d'exécuter la macro et de reproduire ainsi automatiquement les actions enregistrées. Une économie d'efforts et un gain de temps appréciables.

Ainsi quand vous savez comment enregistrer et exécuter, nous pouvons affirmer que vous ne manquerez pas d'apprécier leur utilité et vous souhaiterez probablement accroître d'avantage leur puissance en leur ajoutant votre propre code Visual Basic. Et dès lors, que vous soyez un utilisateur débutant ou que vous possédiez une grande expérience de l'écriture des macros, Visual Basic ne manquera pas d'améliorer votre productivité. C'est justement à cause de cela que nous avons préféré faire notre logiciel avec VBA.

Avec Visual Basic Application, nous avons créé des commandes, des menus, des boîtes de dialogue, [des messages]et des boutons personnifiés. Nous avons aussi utilisé son aide très bien faite et qui, avec Internet, nous a permis d'avoir des informations exhaustives pour son utilisation rapide. Et ainsi, nous avons transformé Microsoft Excel en une application totalement différente. De l'automatisation des tâches fastidieuses au développement d'applications puissantes et complètes.

2 Automatisation des tâches répétitives

2.1 Introduction aux macros :

Plus vous travaillez avec Microsoft Excel, plus vous constaterez que vous exécuter certaines tâches de manière routinière. Par exemple la mise à jour régulière des chiffres de ventes, des demandes espérez par rapport aux demandes réelles actuelles exige parfois que vous répétiez une même séquence d'actions et de commandes. Ainsi, nous avons automatisé la plupart des tâches de la gestion de la production afin de permettre aux utilisateurs de gagner du temps et d'épargner des efforts, en utilisant le puissant langage de programmation Visual Basic dans Microsoft Excel.

Dans ce sous chapitre nous allons essayer de vous éclairer sur l'enregistrement des tâches. Cependant, signalons d'abord qu'il n'est pas indispensable de comprendre la programmation pour se lancer dans l'utilisation de VBA.. Microsoft Excel comprend un Enregistreur de macros, outil intégré qui crée le code Visual Basic à votre place, en fait l'enregistrement de macros n'est qu'un début. Ensuite, vous devez généralement modifier et personnaliser ces macros en fonction de vos besoins. Pour visualiser et modifier du code visual Basic reportez-vous au chapitre 3 ou nous avons expliqué comment réaliser notre travail.

répéter automatiquement les actions ainsi enregistrées. Dès que vous avez enregistré une macro, vous pouvez aussi l'affecter à une commande ou à un bouton. Pour exécuter la macro, il vous suffit par la suite de choisir la commande ou de cliquer sur le bouton.

2.3 Quand faut-il enregistrer une macro ?

Vous devez envisager l'enregistrement d'une macro chaque fois que vous tapez régulièrement une même séquence de touches, que vous choisissez les mêmes commandes ou que vous exécutez la même séquence d'action. Les tâches journalières que vous pouvez automatiser à l'aide de macros comprennent notamment :

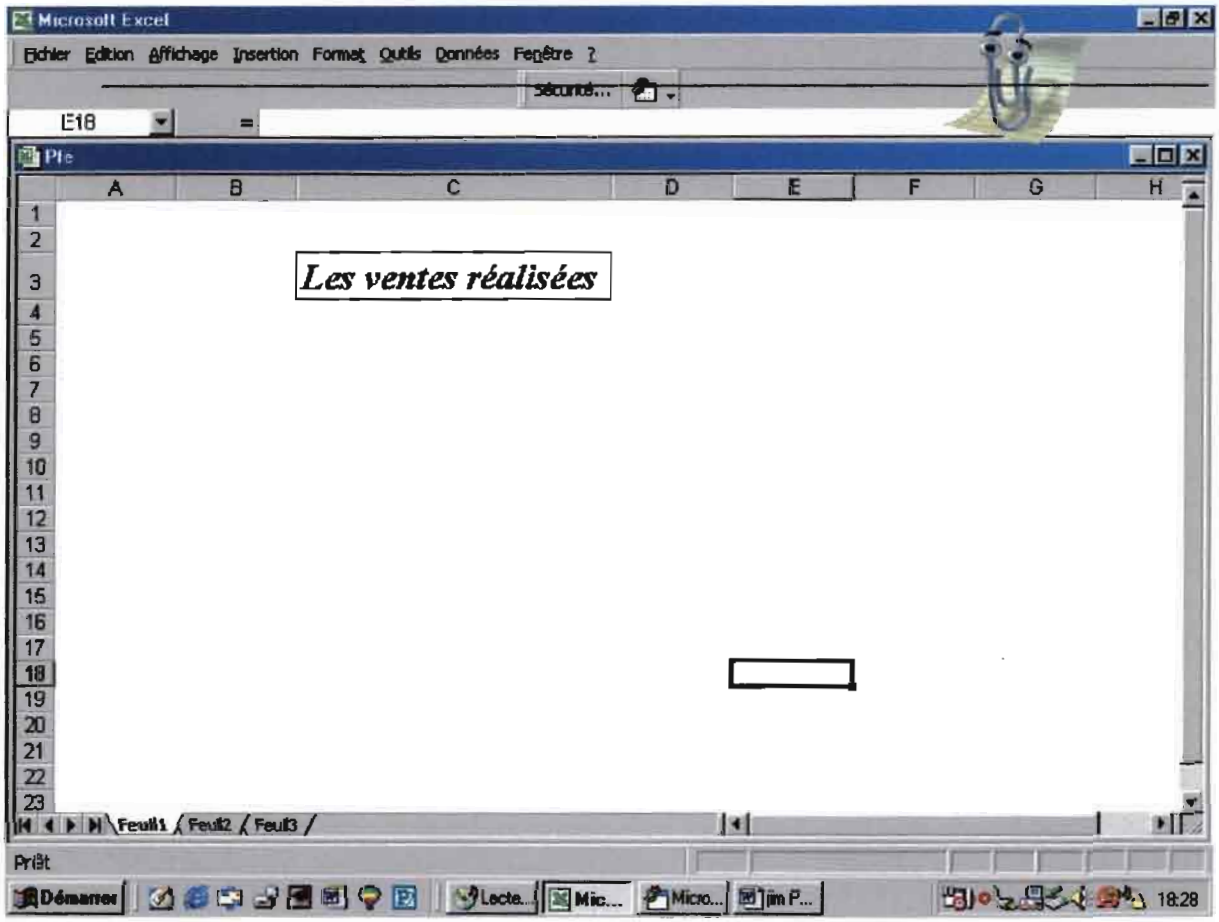
- L'ouverture d'un groupe de classeurs et la lecture des informations qu'ils contiennent.
- L'impression de plusieurs plages de cellules.
- L'ouverture de données, son tri, la création d'un rapport et sa fermeture
- La définition d'une nouvelle feuille de calcul en introduisant des titres, en ajustant la largeur des colonnes et en appliquant des mises en forme etc…

Et en exemple, nous supposons qu'on doit régulièrement définir une nouvelle feuille de calcul dans laquelle on introduit des données de

ventes. Après avoir basculé vers une nouvelle feuille de calcul Excel, on exécute les actions suivantes :

- Désactiver le quadrillage
- Sélectionner la cellule C3.
- Introduire le titre : les ventes réalisées
- Mise en forme du titre en Times New Roman 18 points
- Mise en gras et en italique du titre
- Application d'une bordure de couleur autour de la cellule.
- Elargissement de la colonne C afin qu'elle puisse accueillir le titre.

Figure 1 :Macro d'exemple : la feuille de calcul complétée est montrée dans l'illustration suivante.



Pour accélérer la procédure de définition de cette feuille de calcul, vous pouvez enregistrer une macro qui exécute la totalité de la tâche à votre place. Ensuite, lorsque vous exécutez la macro, Microsoft Excel définit automatiquement votre feuille de calcul en utilisant une séquence de procédures identiques à celle que vous avez exécutée.

Remarque : Bien que vous puissiez aussi définir les titres et les mises en forme de vos feuilles de calcul à l'aide d'un modèle, il est préférable d'utiliser une macro, car vous pourriez toujours l'améliorer et la personnaliser ultérieurement en vue d'automatiser des tâches encore plus complexes. Vous pouvez notamment créer une macro qui affiche une zone dans laquelle vous introduisez le texte de titre, définissez son emplacement dans la feuille de calcul.

Procédure d'enregistrement

Dans Microsoft Excel, l'enregistreur de macro enregistre des actions que vous exécutez ou des commandes que vous sélectionnez pendant la session en cours. L'enregistreur de macros fonctionne de la même manière qu'un magnétophone. Alors que ce dernier enregistre votre voix, l'enregistreur de macros enregistre vos actions.

Les actions que vous exécutez…

Les mots que vous prononcez…

Enregistreur
De macros

…sont enregistrées dans des macros.

…sont enregistrées sur une cassette

Et quand on exécute une macro ou qu'on lise la cassette, on aura ressemblance.

Enregistreur
De macros

Quand vous exécutez une macro enregistrée…

Quand vous lisez une cassette…

Microsoft Excel exécute les actions

… le son enregistré est émis Automatiquement.

3 Création de fonctions personnalisées :

3.1 Introduction :

Dans ce sous chapitre nous allons essayè de vous faire une fonction personnalisée. Ainsi, progressivemen dans le rapport nous tenterons de décrire comment concevoir des fonctions personnalisées, les créer et les entrer dans une feuille de calcul. Une fonction personnalisée est semblable à n'importe quelle fonction de feuille de calcul intégré dans Microsoft Excel, par exemple SOMME ou MOYENNE. Toutefois, dans le cas des fonctions personnalisées c'est à nous de déterminer exactement la tâche qu'elle aura à exécuter.

Une simple fonction personnalisée peut parfois remplacer une longue formule ou une formule indiquée dans une feuille de calcul, voire un ensemble de formules. En remplaçant plusieurs formules par une seule dans une feuille de calcul, les fonctions personnalisées s'avèrent plus faciles à mémoriser et plus facile à utiliser.

Toutes les fonctions personnalisées reposent sur du code Visual Basic. Cela n'implique cependant pas que leur mise en œuvre nécessite une connaissance approfondie de la programmation. Ce sous chapitre ne constitue qu'une première approche. Il vous présente des fonctions personnalisées, tout en évitant de mentionner de nombreux termes, sujets et éléments d'informations importants dans Visual Basic.

Par ailleurs, mieux on connaît Visual Basic, plus on est à même d'écrire des fonctions qui répondent vraiment à nos besoins. Et ceci, nous allons le voir dans les chapitres [3]et[4]

3.2 Rôle d'une fonction personnalisée :

On crée une fonction personnalisée dans un module Visual Basic en combinant des expressions mathématiques, des fonctions Microsoft Excel intégrées et du code Visual Basic. Par la suite, on lui fournit un ensemble de valeurs, afin qu'elle puisse effectuer des calculs et renvoyer une nouvelle valeur.

Remarque : une fonction personnalisée s'applique également à du texte, des dates et des valeurs, sans se limiter aux nombres et aux expressions mathématiques. Cependant, vous n'aurez pas location de le voir dans ce rapport parce que justement, on n'en a pas utilisé dans notre travail.

Une fonction personnalisée est semblable à une macro. Il existe toutefois des différences entre une telle fonction et le type de macro qu'on a en présenter un peu plus haut. Voici certaines de ces différences :

Macros enregistrées	Fonctions personnalisées
Exécutent une action, qu'il s'agisse de créer un graphique ou de déplacer des céllules, par exemple.	Renvoient une valeur, mais ne peuvent exécuter aucune action.
Peuvent être enregistrées.	Doivent être crées dans un module Visual Basic.
Sont encadrées par les mots **Sub** et **End Sub**	Sont encadrées par les mots clés **Function** et **End Function**

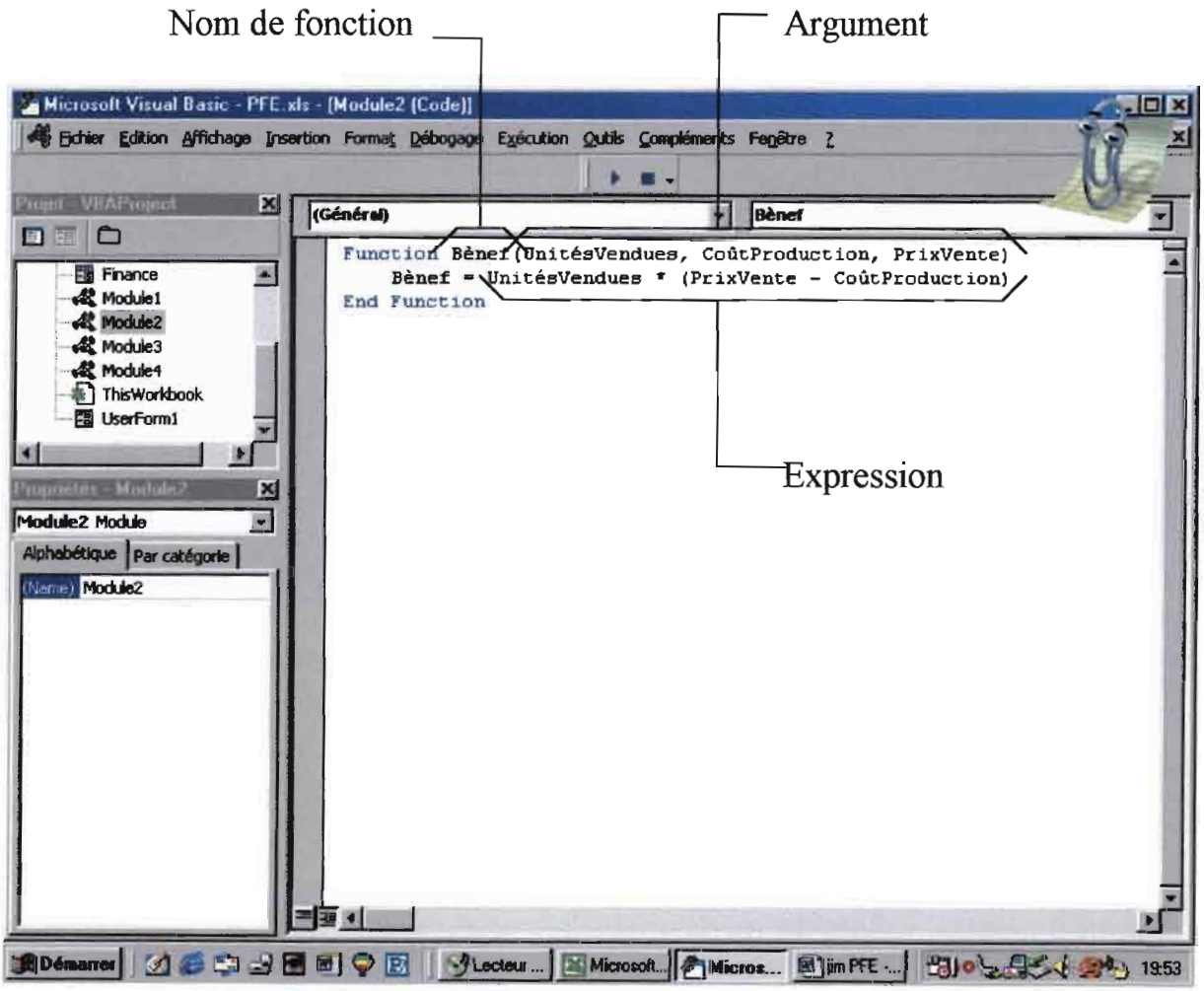
La différence fondamentalement réside dans le fait que les macros enregistrées exécutent des actions qui modifient une feuille, d'une manière ou d'une autre, tandis que les fonctions personnalisées renvoient des valeurs.

3.3 Les éléments d'une fonction personnalisée :

Les différents éléments d'une fonction très simple définie par l'utilisateur de la fonction sont illustrés ci-dessous. La fonction Bénef calcule le

bénéfice brut sur la base du nombre d'unités vendues d'un produit déterminé du coût total de la fabrication de chaque produit et du prix obtenue pour chaque unité. Ainsi, la fonction possède trois arguments (Unités vendues, coût de production et prix de vente) ainsi qu'une expression mathématique.

Figure 2 : Exemple de fonction personnalisée



Comme une fonction personnalisée accepte des valeurs, exécute des calculs et renvoie une valeur, on doit prévoir les éléments suivant, afin que les fonctions personnalisées puissent effectuer ces opérations :

- Les instructions **Function** et **End Function**. Ces mots clés de Visual Basic marquent le début et la fin de fonction.
- Un nom. Il s'agit de l'identificateur unique de la fonction.
- Arguments. Il s'agit des valeurs qu'on fournit. C'est à partir de celles-ci que la fonction effectue ses calculs. Pour définir les arguments d'une fonction personnalisée, on doit taper les noms des arguments entre parenthèse à la suite du nom de la fonction, en les séparant de liste. Le type de séparateur de liste dépend des paramètres du pays de l'utilisateur. En français le séparateur est un point virgule. Pour plus d'informations l'écriture de code pour une utilisation internationale, l'annexe A pourra être utile.
- Code et expression Visual Basic. Il s'agit des instructions qui déterminent le calcul que la fonction personnalisée va exécuter. Une expression produit une valeur. Elles est constituée de nombres, de variables et d'opérateur mathématiques.
- La valeur de renvoi. Il s'agit de la valeur renvoyée par la fonction personnalisée une fois qu'elle a terminé ses calculs.

3.4 Comprendre les éléments d'une fonction personnalisée :

Certains éléments d'une fonction personnalisée sont semblables à ceux qu'on a déjà rencontrés dans les feuilles de calcul, tandis que d'autre peuvent souvent nous être inconnus. Ce qui suit pourra servir à expliquer chacun de ces éléments en partants des connaissances déjà acquises les feuilles de calcul Excel et leurs formules.

Arguments

Les arguments des fonctions personnalisées semblable au nom qu'on définit pour les cellules d'une feuille de calcul. Un argument est type de variable. Une variable est un nom qui représente une valeur. Il peut également s'agir d'une référence à un objet tel qu'une cellule ou une feuille de calcul. Si les noms facilitent la lecture et la compréhension des formules des feuilles de calculs, les arguments et les variables permettent e rendre les expressions de nos fonctions personnalisées compréhensibles.

Exemple :

La fonction suivante cale un bénéfice après impôt (bénéfice net). Elle a pour arguments unités vendues, coûtProduction, prix vente et taux Imposition. Les autres variables sont BénefBrut et BénefNet est une variable spéciale, car elle renferme la valeur de renvoi.

```
Function Bénefnet (UnitésVendues ; CoûtProduction ; Prix-
                                    Vente ; TauxImposition)
    BénefBrut = UnitéVendues x (PrixVente - CoûtProduction)
    BénefNet = BénefBrut x (1 - TauxImposition)
End Function
```

Cet exemple illustre l'utilisation conjointe d'arguments et de variables dans des expressions. L'expression qui calcule le bénéfice brut utilise trois arguments de la fonction tandis que le résultat de l'expression est enregistré dans la variable Bénef Brut, tandis que le résultat est enregistré dans Bénef Net.

Il est aussi possible de créer une fonction personnalisée qui ne nécessite aucun argument. Une telle fonction peut, par exemple effectuer des calculs à partir de la valeur contenue dans une cellule ou à partir de l'heure qu'il est, plutôt que d'utiliser les valeurs que vous fournissez dans les arguments.

Expression Visual Basic

Les expressions sont semblables aux formules que l'on tape dans les feuilles de calcul. La plupart des fonctions et opérateurs mathématiques qu'on dans les formules d'une feuille de calcul sont accessibles aux

fonctions personnalisée il existe toute fois plusieurs différence entre les formule feuille de calcul et les expressions Visual Basic :

- Les formules de feuilles de calculs sont tapées dans de cellules alors qu'on tape les expression Visual Basic dans le module Visual Basic.
- Les formule de feuille de calcul commencent par un signe égal et le résultat est placé dans la cellule qui renferme la formule. Comme Visual Basic a besoin de savoir ou placer le résultat d'une fonction personnalisée, les expressions Visual Basic sont précédées par une variable et signe égal. La variable située sur la gauche de l'équation correspond à la destination ou au lieu d'enregistrement de la valeur qui est calculée sur la droite. Dans la ligne ci-dessous Bénef Brut est une variable contenant le résultat de l'expression mathématique située sur la droite.

```
BénefBrut = UnitésVendues x (prixVente - CoûtProduction)
```

La ligne de code complète – une variable suivie d'un signe égal lui-même suivi d'une expression est appelé introduction d'affection, car elle effectue une valeur à une variable.

- Les fonctions personnalisées peuvent inclure des instructions Visual Basic telle que si, if, For et Do (voir chapitre 3 : code du bouton CLASSIFICATION ABC).

CHAPITRE 2

Fn =(Fn / A, i%, n)

De la Gestion A la programmation

Comme il a été dit plus haut dans ce rapport, plus on maîtrise Visual Basic mieux on réussit à créer des fonctions plus adaptées à nos besoins. Aussi, pour modifier un macro, Visual Basic devient encore indispensable. Dès lors, il devient aisé de comprendre, dans ce chapitre introductif de notre travail, que le langage V.B. sera de premier ordre pour expliquer ce qui nous attendait. Ainsi, pour introduire cette section qui parle de gestions financières comme la plupart des programmes basés sur des formules, notre logiciel n'est pas là seulement pour faciliter des tâches répétitives et parfois longues. Mais avant tout, un outil qui veut se faire de la place dans l'économie sénégalaise.

Les différentes parties du chapitre

- Création de macros
- Exécution de macros
- Création de fonctions personnalisées
- Exécution de fonctions personnalisées
- De la gestion de la production (cours de gestion) à la programmation

1 De la gestion de la Production à la Programmation

Comme l'indique le nom du chapitre, il fallait partir de plusieurs formules pour faire notre travail. Ainsi, les macros et fonctions personnalisées développées largement plus haut ont été de premier ordre. Toutes les équations utilisées ont été introduites dans notre programme sous forme de macro ou de fonction personnalisée. Cependant, remarquons que, par rapport à leur forme générale elles étaient plus attendues en tant que fonctions personnalisées, que macros. Pour mieux comprendre notre choix nous vous invitons à suivre le code de cette fonction personnalisée qui devait calculer la valeur future d'un montant à l'année ***n***, lorsque ***i*** représente le taux d'actualisation et P_o représente la valeur présente du même montant

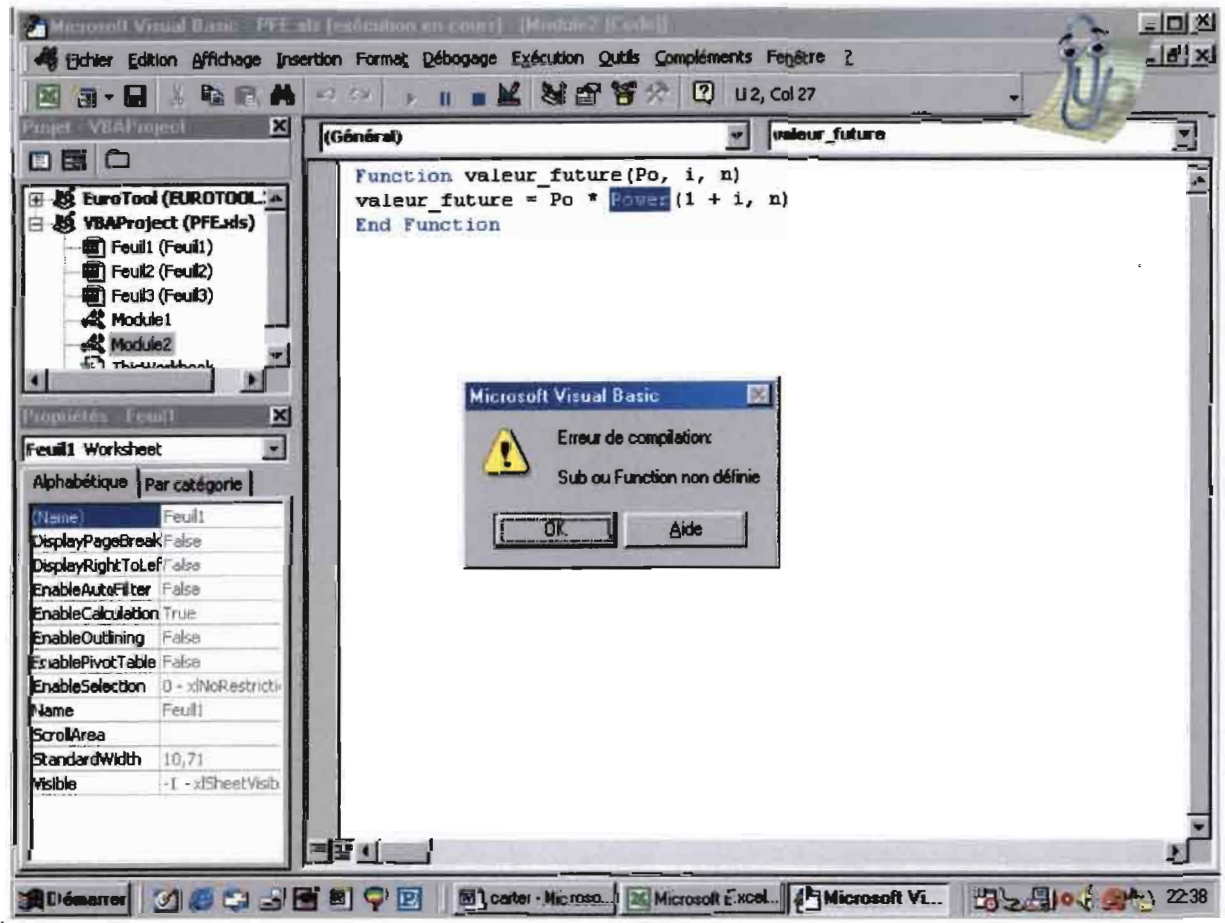
$$F_n = P_o.(1+i)^n$$

Le code correspondant à cette formule devait être

```
Function valeur_future (Po, i,n )
    valeur_future = Po*Power(1+i,n)
End Function
```

Il a était écrit sans faute (ce code) dans un module créer à cette effet. Ainsi, comme nous allons le voire dans ce schéma suivant, nos attentes n'ont pas été satisfaites.

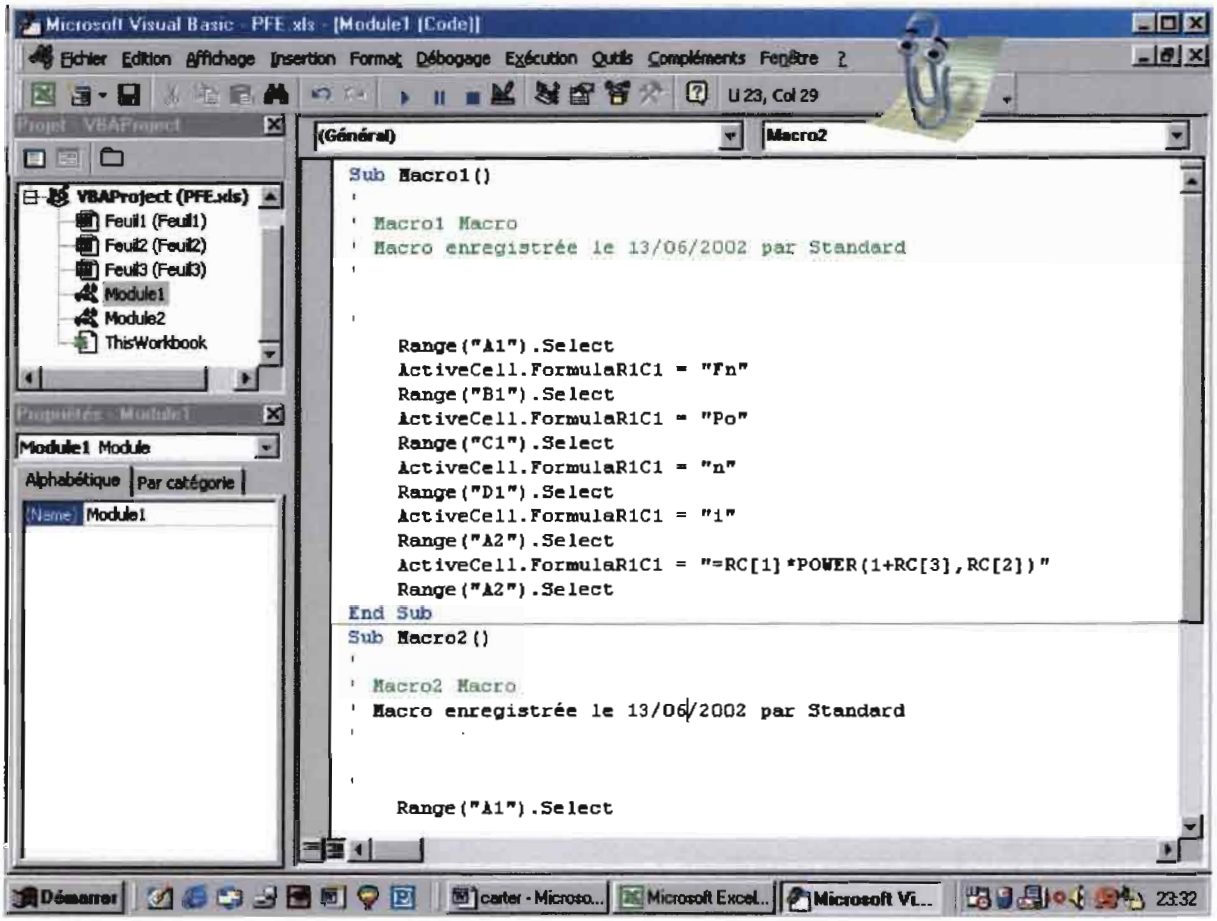
Figure 3 : Fonction personnalisée défectueuse



En effet VBA (Visual basic Application) ne connaît pas la fonction ***Power*** (Puissance). Et ceci, lorsque les macros la reconnaissent parfaitement (voir schéma ci-dessous). Dés lors un problème c'est posé : comment insérer dans le programme, la fonction **puissance** de Microsoft Excel ? Il restera sans suite, parce que justement, nous avons préféré utiliser les macros au lieu de s'engager dans une recherche qui s'annonçait difficiles

dés le début. Ainsi, pour beaucoup de fonctions nous avons choisi les macros à la place des fonctions personnalisées.

Figure 4 : Macro pour le calcul d'une valeur future



2 Enregistrement de macro

Pour automatiser une tâche dans Microsoft Excel, nous avions d'abord enregistrer une macro. Vous trouverez ci-dessous la procédure générale pour enregistrer une macro ainsi que la démarche détaillée qui illustre

l'exécution de cette procédure, par rapport à l'enregistrement de notre fonction *valeur future.*

☆ **Pour enregistrer une macro**

- Choisissez dans le menu **Outil, macro** puis **Nouvelle macro.**
- Dans la zone << Nom de la macro >>, tapez le nom de la macro.

Le nom peut contenir des lettres, des chiffres et des traits de soulignement, mais pas d'espaces ni de signes de ponctuation. Il doit commencer par une lettre.

- Dans la zone << Description >>, tapez la description de la macro.
- Pour définir des options pour la macro, choisissez le bouton <<Options>>, puis définissez les options souhaitées.

Pour obtenir une description de ces options, choisissez le bouton <<Aide>>.

- Choisissez << OK >>.

Pendant que l'enregistreur de macros fonctionne, le bouton <<Arrêter l 'enregistrement >> apparaît à l'écran dans sa propre barre d'outils.

■

- Exécuter les actions que vous voulez enregistrer.
- Cliquez sur le bouton<< Arrêtez l'enregistrement >>.

Vous pouvez aussi choisir, dans le menu **Outils,** la commande **Macro,** puis **Arrêter l'enregistrement.**

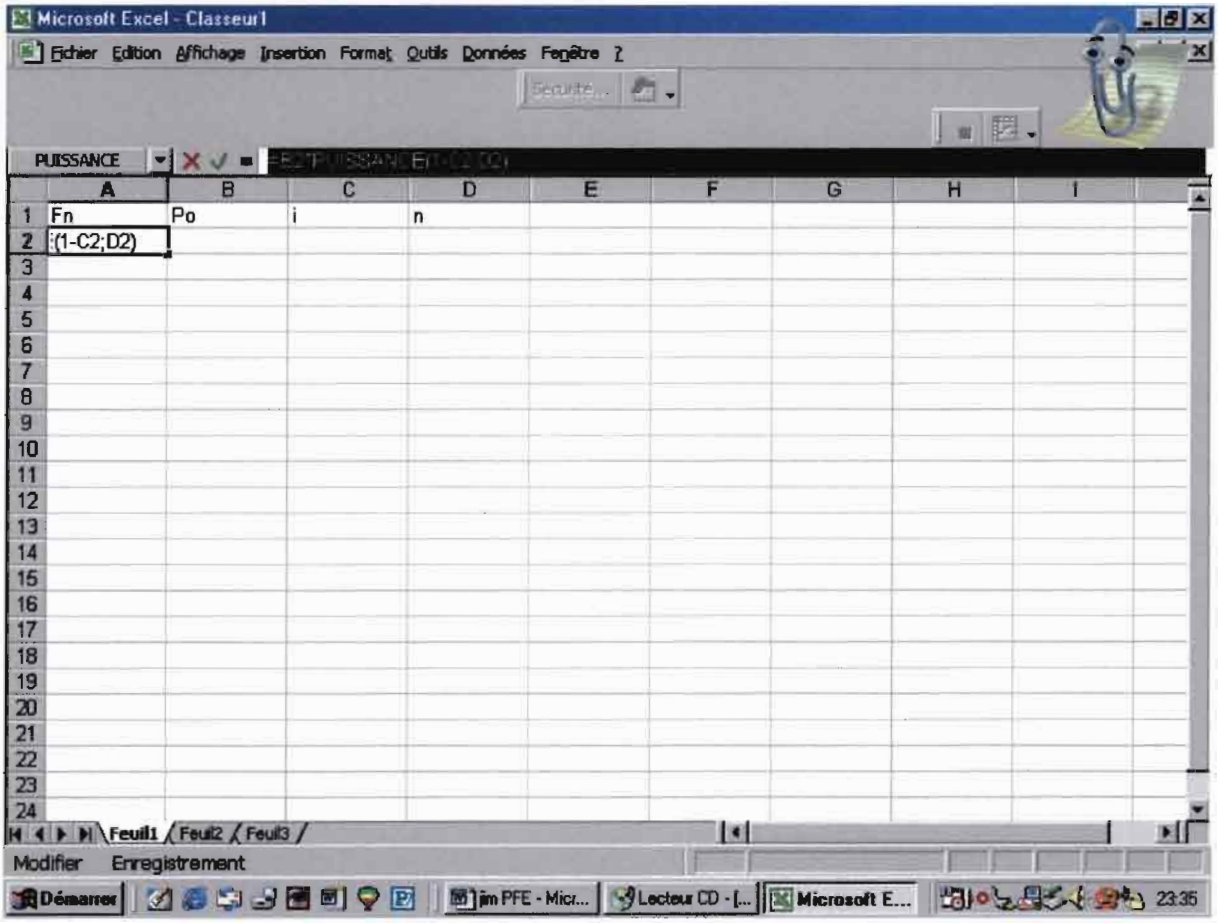
Conseil : Pour accélérer la procédure d'enregistrement, vous pouvez sauter les étapes 2 et 3 de la procédure détaillée ci-dessus et laisser Microsoft Excel nommer automatiquement votre macro. Celle-ci reçoit le nom << Macron >> où n représente le premier chiffre qui confère à la macro un nom unique.

Exemple

La procédure ci-dessous décrit comment enregistrer une macro qui calcule une *valeur future* définit par la formule suivante

- Choisir **Outils Macro**
- Choisir **Nouvelle macro** et définir les propriétés de votre nouvelle macro (Nom, Raccourci, etc…)
- Comme le montre le schéma suivant, remplir les cellules A1, B1, C1 et D1 (respectivement Fn, Po, i, n).
- Calculer dans A2 la formule de la *valeur future* en lui donnant des valeurs vides des B2, C2 et D2
- Vous avez maintenant terminé votre enregistrement. Il ne vous reste qu'à appuyer sur le bouton **STOP** pour arrêter votre macro.

Figure 5 : Enregistrement de macro



3 Exécution d'un macro

Après avoir enregistré une macro, vous pouvez la lire ou l'exécuter à tout moment ; Microsoft Excel exécute toutes les commandes enregistrées dans la macro.

☆ **Pour exécuter une macro**

- Choisissez **Outils Macro**
- Dans la zone << Nom de macro >>,tapez ou sélectionnez un nom.

- Choisissez le bouton << Exécuter>>.

Remarque : Vous pouvez interrompre l'exécution d'une macro en appuyant sur la touche ECHAP. Cette fonction est très utile lorsque vous n'exécutez pas la macro adéquate ou que vous décidez d'interrompre l'exécution de la macro. Quand vous interrompez une macro, Microsoft Excel affiche la boîte de dialogue **Erreur Macro.** Pour fermer la boîte de dialogue le bouton ECHAP suffira encore.

---

4 Création d'une fonction personnalisée

Dans le chapitre précédent nous avons définit le rôle et les éléments d'une fonction personnalisée dans un module Visual Basic. Aussi, comme nous l'avions dit dans le même chapitre, la connaissance du code VB était indispensable pour créer une fonction. Ainsi, avec Visual Basic nous allons tenter d'expliquer la création de fonctions personnalisées.

☆ **Pour créer une fonction personnalisée**

- Pour passer dans un module Visual Basic dans le classeur actif, sélectionnez l'onglet d'un module Visual Basic.

-Ou -

Pour créer un nouveau module VB, choisissez dans le menu **Outils** la commande **Macro,** puis choisissez **Module.**

- Tapez **Function** suivi du nom de votre fonction personnalisée.
- Tapez la liste des d'arguments, entourée de parenthèses, en séparant les arguments par une virgule.
- Appuyez sur ENTREE pour passer à une nouvelle ligne.

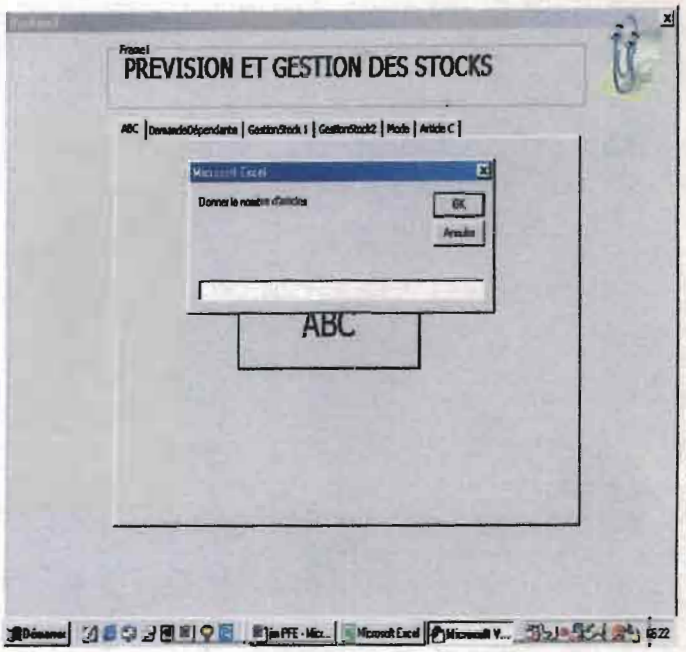
Excel Vérifie la syntaxe de la ligne que vous venez de taper. Les mots clés Visual Basic apparaissent maintenant en bleu.

- Appuyez sur TAB, tapez votre première ligne de code puis appuyez sur ENTREE.

En appuyant sur TAB, vous mettez votre code en retrait afin de faciliter la lecture.

- Tapez les autres lignes de code.
- Tapez **End Function**, puis appuyez sur ENTREE.

CHAPITRE 3



Logiciel de Gestion de La production

Ce chapitre décrit comment le logiciel a pu permettre à nos connaissances en Gestion de la Production d'être plus accessibles à toutes les personnes qui participent à la progression économique de notre pays. Ainsi, il s'agissait avant tout de fournir le maximum d'efforts pour adapter notre logiciel, par rapport aux capacités de compréhension de l'utilisateur et surtout de faciliter son utilisation.

Les différentes parties du chapitre

- Feuilles *boite de dialogue* Visual Basic
- Bouton de commandes personnalisées
- Gestion des erreurs et des valeurs d'erreurs

1 Feuilles boîte de dialogue Visual

1.1 Création de feuille boite de dialogue Visual Basic

Dans le souci de rendre notre logiciel plus facile d'utilisation et surtout plus présentable nous avons créé des feuilles boite de dialogue. Pour mieux vous informer par rapport à ce que l'on a pu faire en choisissant de créer des feuilles boîte de dialogue VB nous vous présentons ci-dessous la procédure générale permettant de les créer.

☆ **Pour créer une feuille boîte de dialogue Visual Basic**

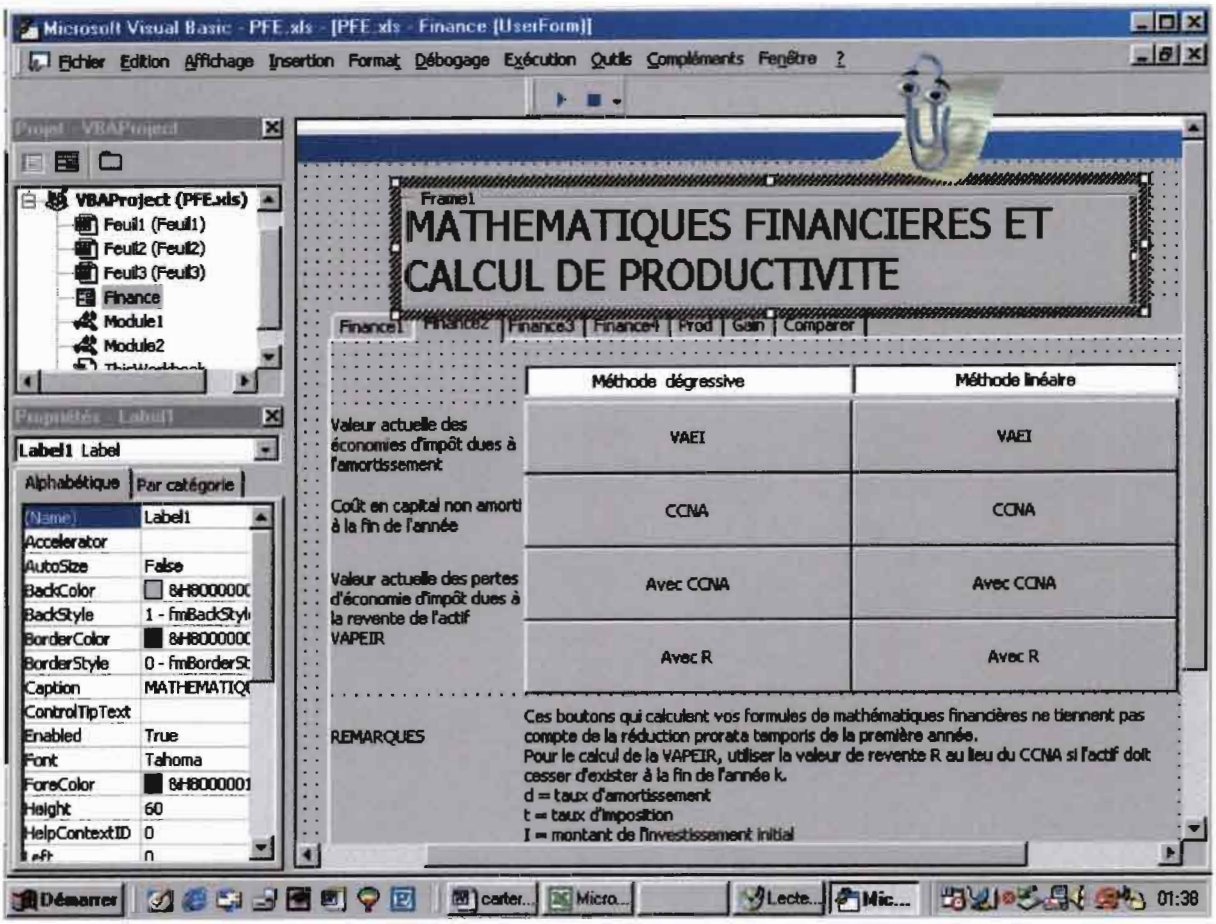
- Avant tout ouvrir la fenêtre VB correspondant à votre feuille Excel en suivant la même procédure pour créer une fonction personnalisée (voir chapitre précédent).
- Dans le menu **Insertion** choisissez **UserForm.**

Ainsi, une boîte à outils apparaît sur votre écran et permet de créer votre feuille boîte de dialogue.

- Agrandir votre feuille boîte de dialogue pour qu'elle puisse contenir tout ce qu'elle doit recevoir.

Ainsi, il sera possible de créer une feuille boîte de dialogue à l'image du schéma suivant.

Figure 6: Mode création d'une feuille boite de dialogue VB



1.2 Exécution d'une feuille boite de dialogue Visual Basic

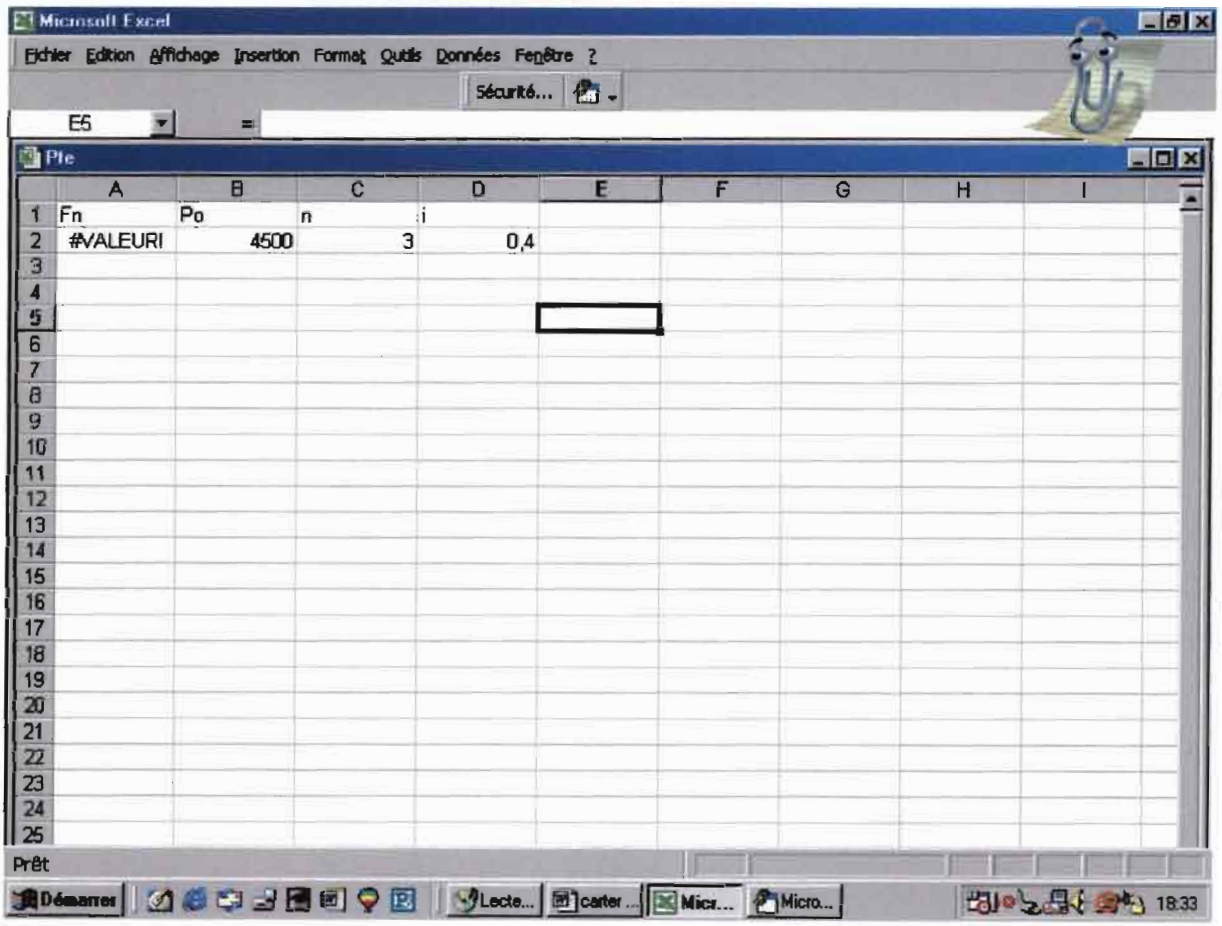
Après avoir créer notre feuille boîte de dialogue, il nous reste maintenant à l'exécuter. Elle servira ainsi, à l'utilisateur de saisir rapidement les possibilités de calcul qui lui sont offertes. D'ailleurs, par rapport à notre logiciel, l'utilisateur sera dispensé de la création de cette feuille boîte de dialogue. En effet, on l'a déjà créée pour lui, et l'exécution de la procédure qui suit n'exige pas beaucoup de connaissances en informatique notamment en Visual Basic et en matière de feuille boîte de dialogue.

Pour exécuter les feuilles boite de dialogue du logiciel

- Ouvrir le classeur Excel qui comprend le développement VBA..

Il est possible de l'ouvrir à partir du bureau en choisissant l'icône nommée ***PROD.*** Ainsi, la feuille Excel prendra la configuration suivante.

Figure 7:Classeur de calcul du logiciel de *Gestion de la production*

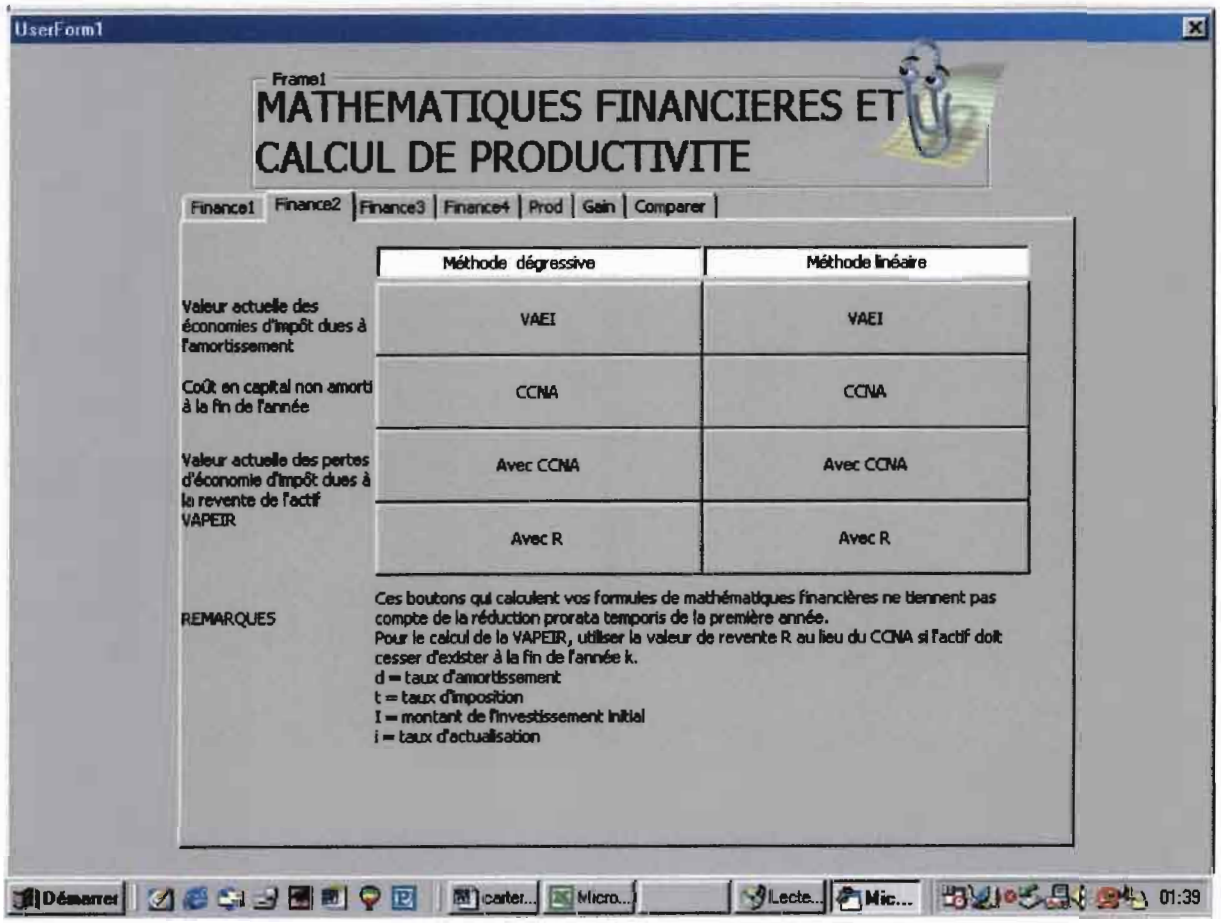


- Choisissez directement le bouton de commande Visual Basic Editor.

- Choisissez **Finances** ou **Stocks** selon que vous voulez calculer respectivement :
- Dans la feuille boite de dialogue ***MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE***
- Ou dans celle nommée ***PREVISION ET GESTION DES STOCKS***
- Assurer vous que Visual Basic n'exécute aucun programme en appuyant sur le même bouton qui arrête une macro (voir création de macro).
- Choisissez à présent le bouton EXECUTER pour démarrer vos calculs.

Ainsi, votre écran vous proposera une feuille similaire à la suivante. Il ne vous restera, dés lors, qu'a commencer vos calculs par rapport à ce que vous voulez chercher (voir *figure 8*).

Figure 8: Exécution du logiciel



2 Bouton de commande personnalisée :

Pour rendre notre boîte de dialogue utile, nous avons eu recours à la *boîte à outils* pour placer les nouveaux contrôles dans le cadre de la boîte de dialogue.

☆ Pour placer un contrôle dans une feuille boîte de dialogue

- Dans la boite à outils, cliquez sur le bouton du contrôle que vous souhaitez ajouter.

- Dans la feuille boîte de dialogue, faîtes glisser le contrôle jusqu'à ce qu'il atteigne la taille et la forme désirées.

Remarque : Dans cette section, nous ne parlerons que du contrôle *bouton de commande.* En effet, dans ce rapport notre objectif n'est pas de vous fournir toutes les informations pour comprendre parfaitement Visual Basic, mais essentiellement : comment on a pu se servir de VB pour construire notre logiciel.

2.1 Définition des propriétés d'un contrôle *bouton de commande*

Quand un contrôle *bouton de commande est crée*, Microsoft Excel lui affecte un jeu de propriétés par défaut, exactement comme s'il s'agissait d'une boîte de dialogue intégrée. Il est possible de modifier certaines de ces propriétés en double cliquant sur le contrôle.

En réalité, le processus d'utilisation d'une boîte de dialogue personnalisée commence par l'affichage de cette boîte à l'écran afin de permettre à l'utilisateur de modifier les contrôles qu'elle renferme. Ainsi le code VB va nous permettre d'accéder à l'état des *boutons de commande* de la boîte de dialogue par le biais de leurs propriétés, puis de modifier la manière

dont l'application réagit aux changements apportés à l'intérieur de la boîte de dialogue.

Ainsi, pour le bouton **CLASSIFICATION ABC** (voir annexe) il a fallu tout cet ensemble code qui suit :

```
Private Sub CommandButton1_Click()
Dim a, i, h, f, j As Integer
Dim e, c, d As Double, b As String
a = InputBox("Donner le nombre d'articles")
For i = 1 To a
    b = InputBox("Donner le nom de l'article", i)
    Sheets(1).Cells(i, 1).Value = b
    c = InputBox("Donner sa demande annuelle")
    Sheets(1).Cells(i, 2).Value = c
    d = InputBox("Donner son prix unitaire en CFA")
    Sheets(1).Cells(i, 3).Value = d
    Sheets(1).Cells(i, 4).Value = c * d
Next i
For h = 1 To a
```

```
f = 0
e = Cells(h, 4).Value
For i = 2 To a
    If Cells(i, 4).Value > e Then
      e = Cells(i, 4).Value
      f = i
   Else
     f = i - 1
Next i
Rows(f).Select
Selection.Cut
j = 2
While Sheets(2).Cells(j, 2).Value <> 0
   j = j + 1
Wend
Sheets(2).Rows(j).Paste
Application.CutCopyMode = False
```

```
Next h

Macro1

End Sub
```

2.2 Affectation de l'ordre de tabulation des boutons

Dans une boîte de dialogue, il existe plusieurs manières de sélectionner un contrôle (de le rendre *actif*). On peut simplement cliquer sur un contrôle, qu'il s'agisse d'un bouton, d'une zone de modification ou d'une zone de liste. On peut également appuyer sur la touche ALT dans Microsoft Excel.

Il est aussi possible de définir l'ordre de tabulation des boutons dans une feuille de calcul. Quand l'utilisateur appuie sur la touche TAB, les boutons deviennent actifs dans l'ordre de tabulation qui a été défini.

☆ **Pour modifier l'ordre de tabulation des boutons**

- Passez à la feuille boîte de dialogue.
- Choisissez **Affichage** puis **Ordre de tabulation.**
- Sélectionnez l'élément dont vous souhaitez modifier la place dans l'ordre de tabulation.

3 Gestion des erreurs et des valeurs d'erreur

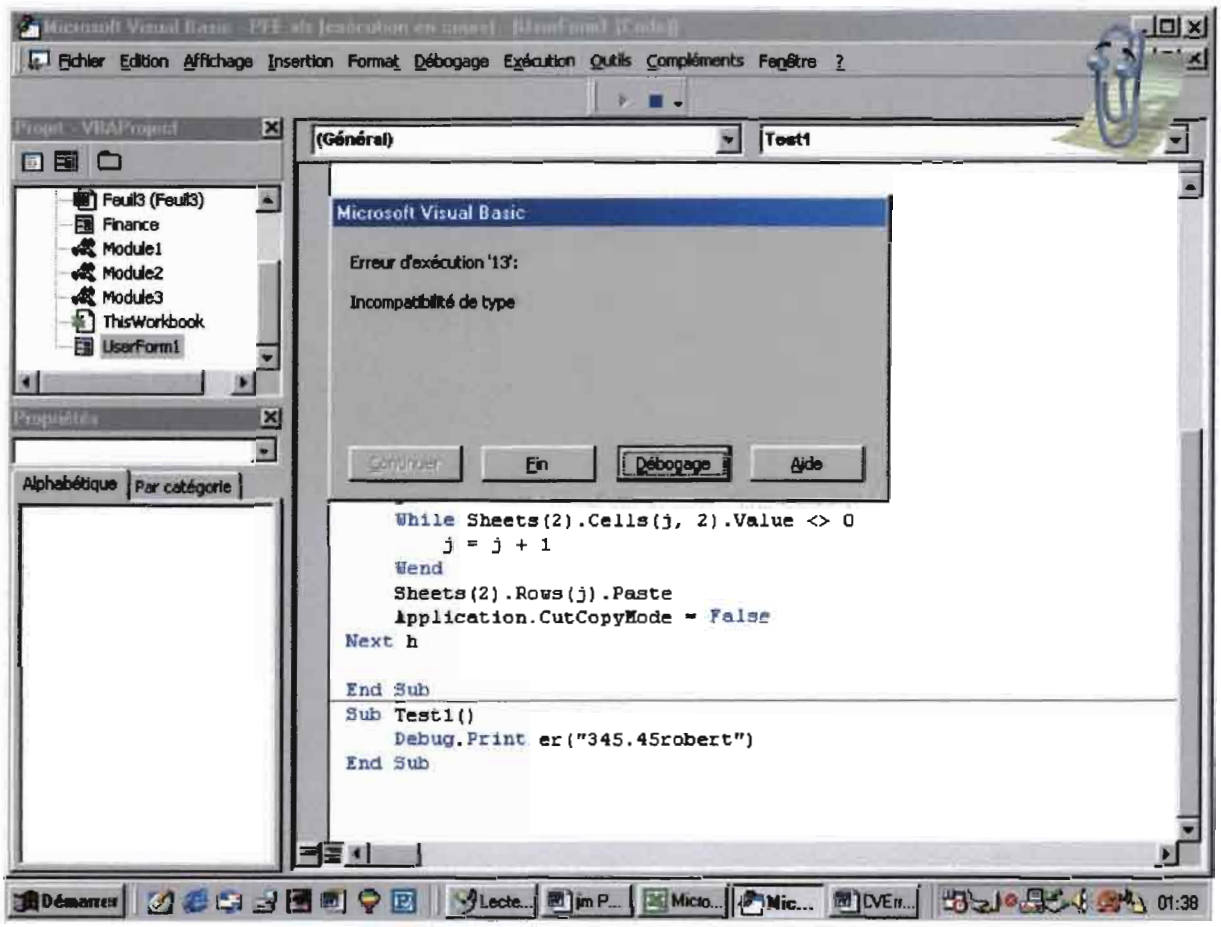
Si tout était parfait, les procédures Visual Basic Application n'auraient pas besoin de code de gestion des erreurs. Malheureusement, des fichiers

peuvent être supprimer par inadvertance, nous pouvons aussi entrer des réponses inadéquates pour des questions posées par les boîtes de dialogue. Ainsi, pour gérer de telles erreurs nous avons modifié tout le logiciel.

En effet, si les erreurs surviennent dans notre code et si nous ne les interceptons pas, Visual Basic génère souvent une erreur d'exécution qui interrompt l'application que l'utilisateur est généralement incapable de relancer. D'autres erreurs n'interrompent peut-être pas l'exécution de notre code, mais l'amènent à se comporter de manière imprévisible. Ainsi, pour le code du bouton **CLASSIFICATION ABC** qu'on a eu à lire plus haut dans ce chapitre, il fallait insérer un ensemble de code VB pour avoir un message qui facilitera sa progression. Cependant, puisque l'utilisateur ne fera qu'entrer des valeurs et chiffres lorsqu'il utilise le logiciel, la gestion des erreurs ne pouvait pas poser de problème. En effet, chacune de ses erreurs correspondra à une incompatibilité de la donnée par rapport aux attentes.

Ainsi, à chaque erreur correspondra une réaction de votre ordinateur décrite par le schéma ci-dessous. Et il lui suffira d'appuyer sur le bouton **FIN** de la boite de message d'erreur, pour pouvoir reprendre l'exécution de la boite de dialogue.

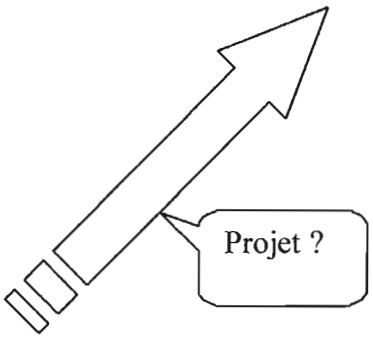
Figure 9 : Message d'erreur pour une mauvaise entrée de données



CHAPITRE 4

Avancement

Du projet



Dans la vie, les objectifs que l'on se fixe et leur réalisation font deux. On n'y peut rien. Ils peuvent être satisfaits entièrement, à moitié et même être dépassés. Par rapport à notre projet, nous avons été dopés par notre amour de la gestion de la production. Ainsi, un engagement fortement mobilisateur de nos compétences nous a guidé tout au long de notre travail. Cependant, à la question : où en sommes nous par rapport à notre projet ? Nous espérons répondre, dans la suite, de manière adéquate.

Les différentes parties du chapitre

- Les objectifs du projet
- Avancement du projet

1 Les objectifs du projet

Comme on l'a dit dans l'introduction de ce rapport, le logiciel a été créé pour participer au développement du SENEGAL. Il devait donc, au moins pour les petites entreprises, veiller à la bonne productivité.

Pour un restaurant, par exemple, le propriétaire devrait vendre des plats de *qualité* et à un *prix raisonnable* pour maintenir et accroître sa clientèle. Cependant, il faudra aussi qu'il *gagne* assez *d'argent* pour pouvoir progresser.

Dès lors, il faudra qu'il *gère la qualité* dans le service aux clients et dans la préparation des repas. Il faudra aussi, pour vendre à un prix raisonnable et avoir des bénéfices en même temps, produire ses plats à moindres coûts. Ainsi, il devra alors acheter ses condiments en *minimisant tous les coûts* que ces achats occasionnent. Pendant la préparation, il aura intérêt à ce que cette dernière mobilise le moins possible ses employés et qu'elle se fasse avec le minimum de ressources. Et, si en même temps, il réussit à optimiser le confort des clients, selon nous, il aura réussi à bien gérer son entreprise. En effet, il aura donc bien *gérer la qualité* dans son entreprise ; il aura aussi bien *gérer son stock* de condiments (de l'achat à la

conservation en passant par le transport), et ceci sans oublier d'accroître la productivité de ses ressources.

Ainsi, pour aider ce propriétaire à bien gérer son entreprise nous avions pensé créer un **LOGICIEL DE GESTION DE LA PRODUCTION**. Ce logiciel devait comprendre trois parties qui sont :

- Calcul financier et calcul de productivité
- Prévision et gestion des stocks
- Gestion de la qualité

2 Avancement du projet

Dans cette partie de notre rapport nous allons vous demander de vous rapporter aux annexes A, B et C. En effet, pour tous les thèmes que vous pourriez voir dans les annexes A et B, nous avons fait de notre mieux pour les développer entièrement. Ainsi, l'annexe C vous servira d'exemple pour saisir de manière précise, comment le développement des thèmes déjà crées a pu être fait.

CONCLUSION ET RECOMMANDATIONS

Les formules de Gestion de la Production sont nombreuses et chacune d'entre elles est utilisée pour obtenir des résultats bien précises. Il nécessite une bonne compréhension des mathématiques allant des calculs les plus simples à la probabilité. Dès lors l'utilisateur doit nécessairement faire des études très poussées. Cependant, le logiciel que nous venons de présenter dans ce rapport, pose des questions à celui qui l'utilise pour que ce dernier puisse obtenir ce qu'il veut calculer. Ainsi, il (notre logiciel) devient un assistant au calcul de gestion.

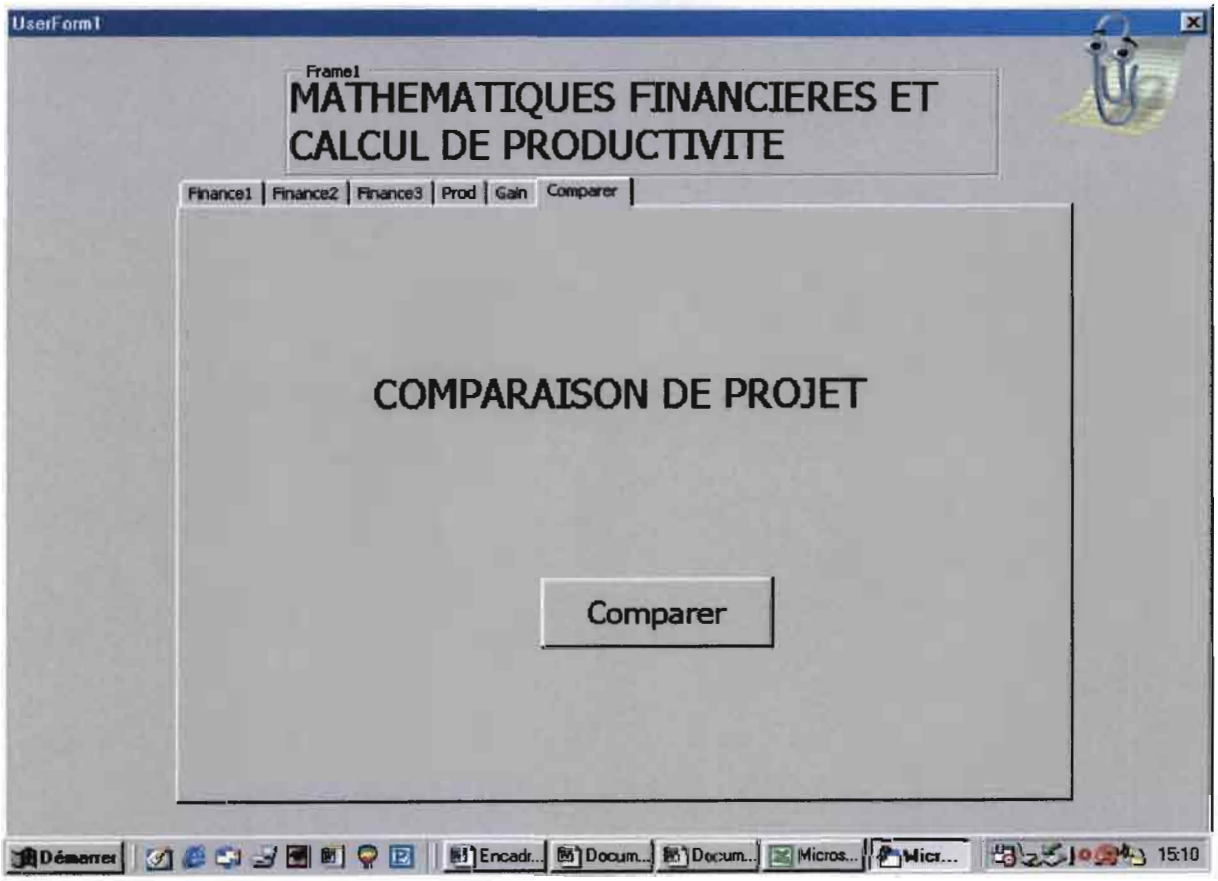
En effet, il a été créé pour satisfaire un public très large incluant les gestionnaires les moins avertis qui n'auront besoin que de la connaissance de Microsoft Excel. Ainsi, nous avons utilisé des macros et fonctions personnalisées pour introduire les équations. Ceux-ci devaient les automatiser et elles pourront à tout moment être appelé. Par la suite, nous avons utilisé des boîtes de dialogues dans lesquelles nous avons installé des boutons, des zones de

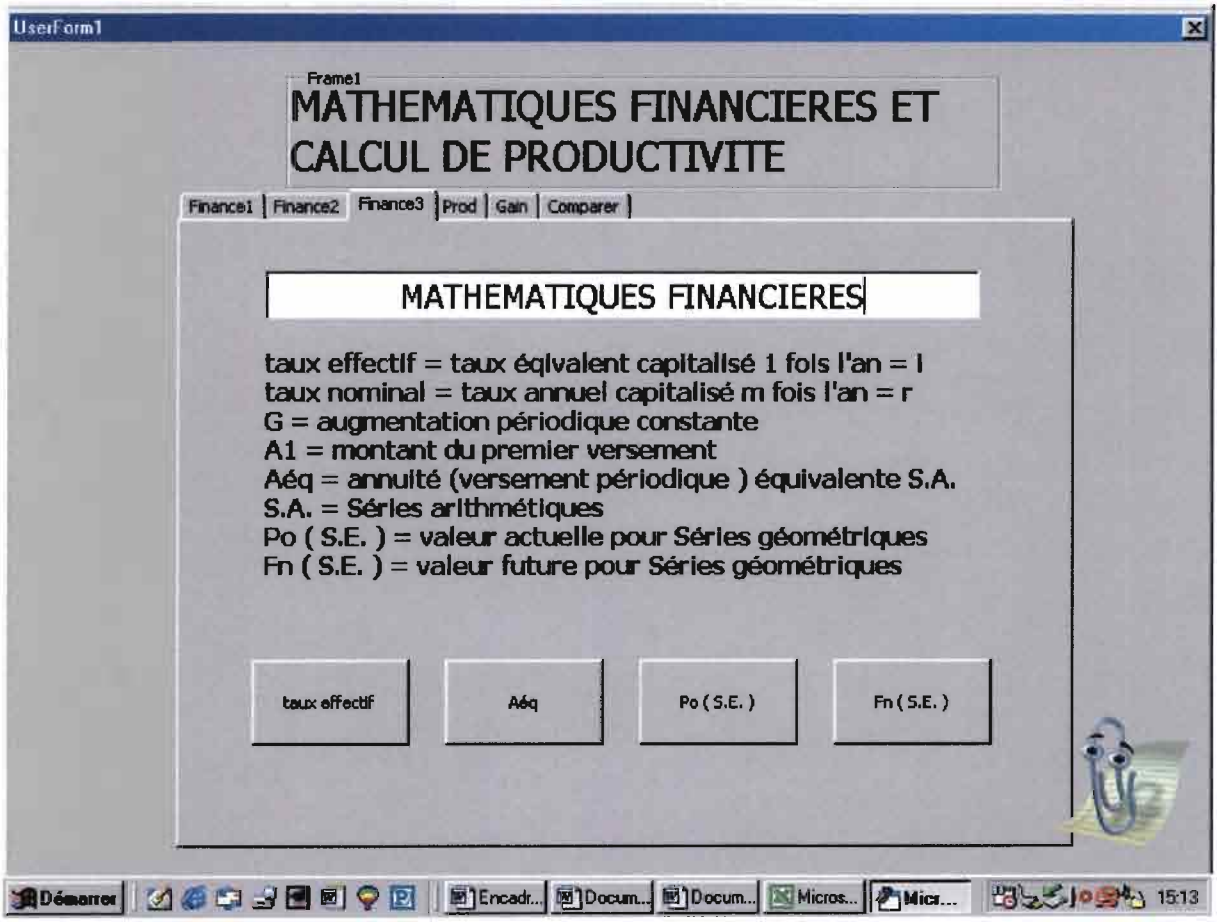
texte et intitulé pour faciliter le choix du thème dans lequel l'utilisateur voudra travailler. Aussi signalons que l'utilisation de boîte de message et boîte de recueil de données servira de guide tout au long de l'utilisation du logiciel.

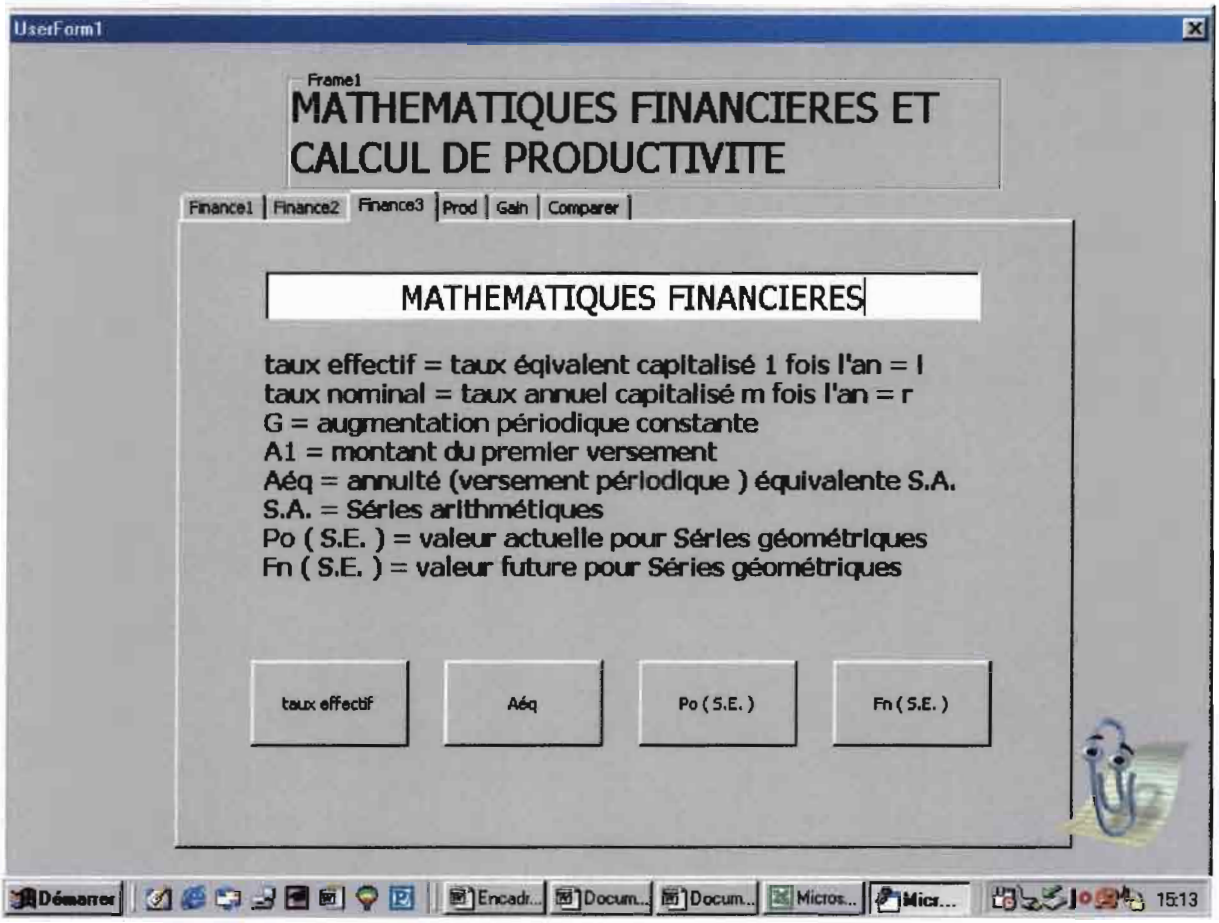
Comme l'autre disait : C'est dans la difficulté que l'on reconnaît les grandes personnes ; nous reconnaissons avoir eu beaucoup de problèmes surtout liés à l'utilisation de Visual Basic dont la documentation n'était disponible qu'en anglais. Ce qui rendait les choses plus difficiles puisque justement, un terme comme MsgBox n'a pas d'équivalent français dans le dictionnaire. Ainsi, nous accueillerons bien une documentation de développement de logiciel, qui serait de version anglaise, parce que justement l'informatique et la mécanique ont tendance à s'unir pour faire mieux.

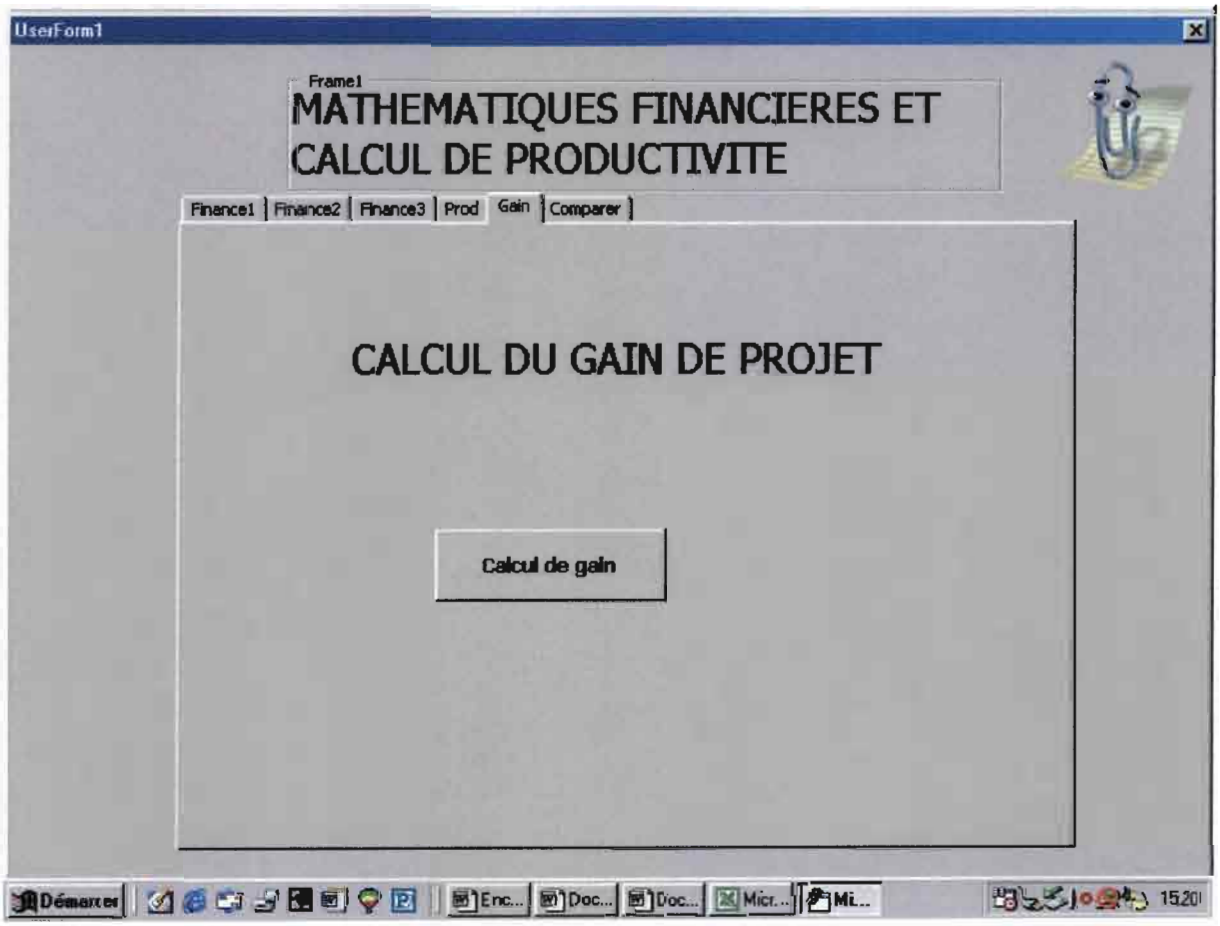
Annexes

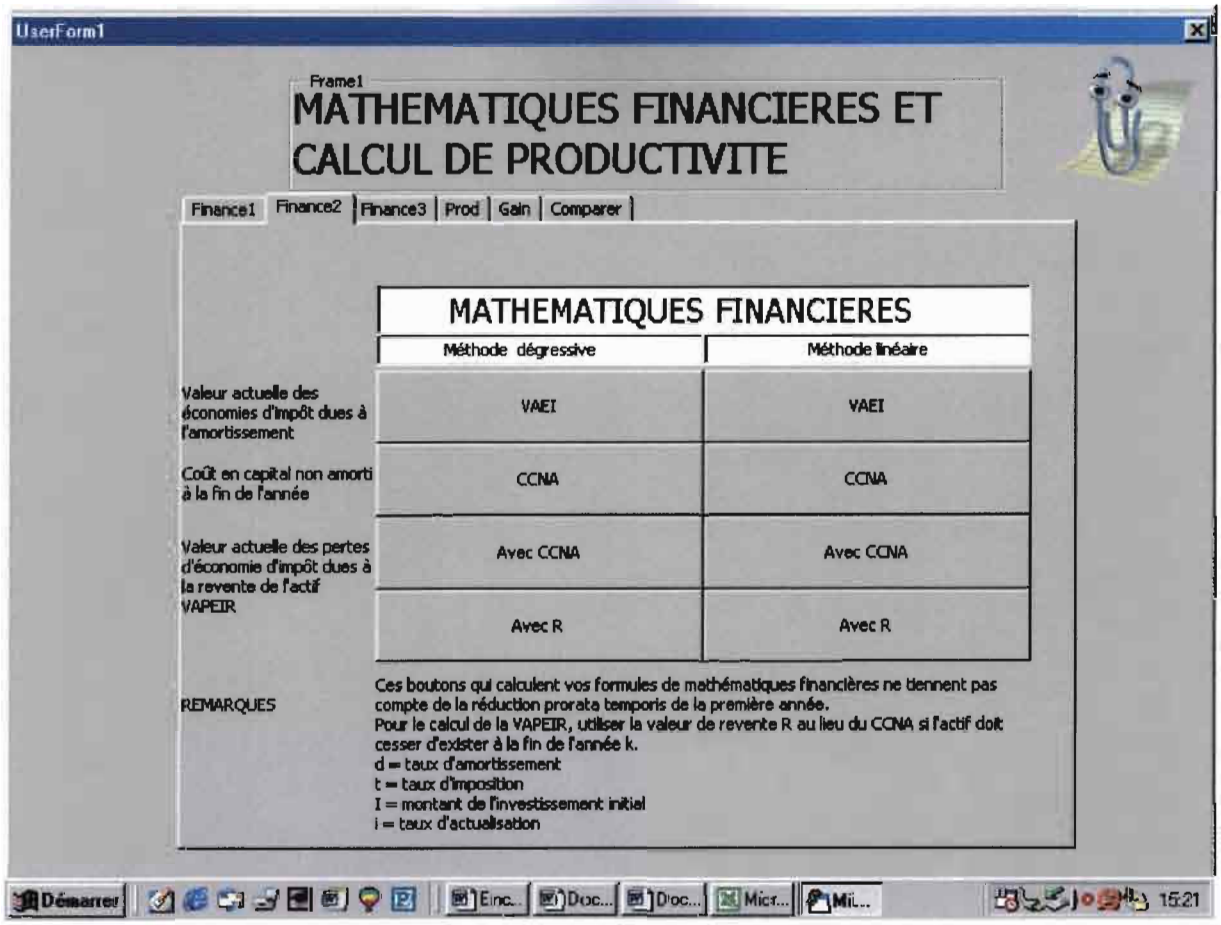
Cercle vicieux << baisse de productivité-inflation chômage- pauvreté>>

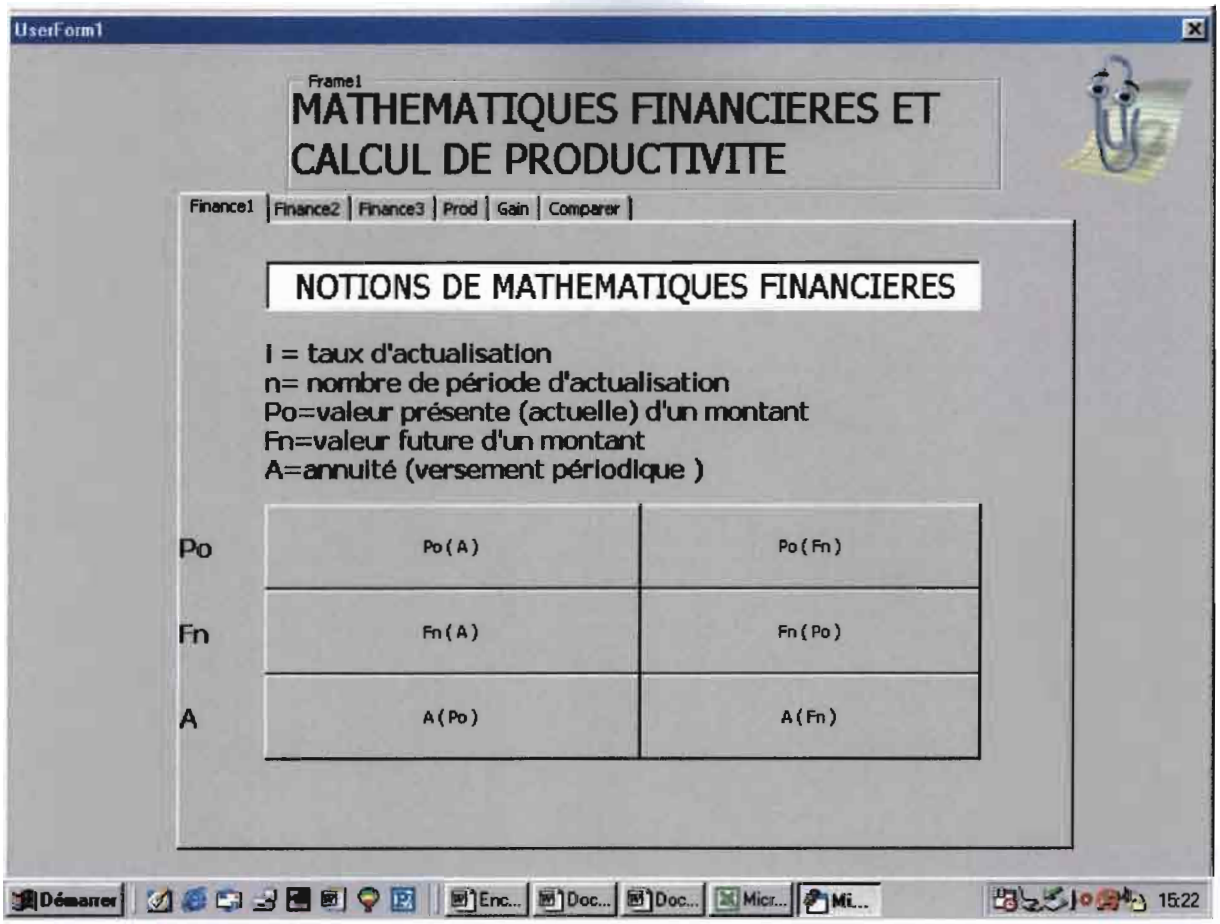

UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
COMPARAISON DE PROJET
Comparer
Démarrer
Encadr...
Docum...
Docum...
Micros...
Micr...
15:10


UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
MATHEMATIQUES FINANCIERES
taux effectif = taux équivalent capitalisé 1 fois l'an = i
taux nominal = taux annuel capitalisé m fois l'an = r
G = augmentation périodique constante
A1 = montant du premier versement
Aéq = annuité (versement périodique) équivalente S.A.
S.A. = Séries arithmétiques
Po (S.E.) = valeur actuelle pour Séries géométriques
Fn (S.E.) = valeur future pour Séries géométriques
taux effectif
Aéq
Po (S.E.)
Fn (S.E.)
Démarrer
Encadr...
Docum...
Docum...
Micros...
Micr...
15:13


UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
MATHEMATIQUES FINANCIERES
taux effectif = taux équivalent capitalisé 1 fois l'an = I
taux nominal = taux annuel capitalisé m fois l'an = r
G = augmentation périodique constante
A1 = montant du premier versement
Aéq = annuité (versement périodique) équivalente S.A.
S.A. = Séries arithmétiques
Po (S.E.) = valeur actuelle pour Séries géométriques
Fn (S.E.) = valeur future pour Séries géométriques
taux effectif
Aéq
Po (S.E.)
Fn (S.E.)
Démarrer
Encadr...
Docum...
Docum...
Micros...
Micr...
15:13


UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
CALCUL DU GAIN DE PROJET
Calcul de gain
Démarrer
Enc...
Doc...
Doc...
Micr...
Mi...
15:20


UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
MATHEMATIQUES FINANCIERES
Méthode dégressive
Méthode linéaire
Valeur actuelle des économies d'impôt dues à l'amortissement
VAEI
VAEI
Coût en capital non amorti à la fin de l'année
CCNA
CCNA
Valeur actuelle des pertes d'économie d'impôt dues à la revente de l'actif VAPEIR
Avec CCNA
Avec CCNA
Avec R
Avec R
REMARQUES
Ces boutons qui calculent vos formules de mathématiques financières ne tiennent pas compte de la réduction prorata temporis de la première année.
Pour le calcul de la VAPEIR, utiliser la valeur de revente R au lieu du CCNA si l'actif doit cesser d'exister à la fin de l'année k.
d = taux d'amortissement
t = taux d'imposition
I = montant de l'investissement initial
i = taux d'actualisation
Démarrer
15:21


UserForm1
Frame1
MATHEMATIQUES FINANCIERES ET CALCUL DE PRODUCTIVITE
Finance1
Finance2
Finance3
Prod
Gain
Comparer
NOTIONS DE MATHEMATIQUES FINANCIERES
I = taux d'actualisation
n= nombre de période d'actualisation
Po=valeur présente (actuelle) d'un montant
Fn=valeur future d'un montant
A=annuité (versement périodique)
Po
Fn
A
Po(A)
Po(Fn)
Fn(A)
Fn(Po)
A(Po)
A(Fn)
Démarrer
15:22