Introduction à la programmation et à l'algorithmique en VBA pour Excel

Cécile Le Pape

cecile.lepape@lip6.fr

http://webia.lip6.fr/~lepape/ens/infogen1

Références

- *http://aigespc57.cicrp.jussieu.fr/algo/index.htm* Introduction à l'algorithmique. Christophe Darmangeat
- *Programmation et Algorithmique en VBA pour Excel*. Anne Brygoo et al. (Ed. Dunod.)
- *Excel 2003, Programmation VBA*. Daniel-Jean David (Ed. Eyrolles)
- *Les cahiers d'exercices – Excel 2007 : Macros et programmation en VBA*. Pierre Rigollet (Ed. ENI)

Préambule

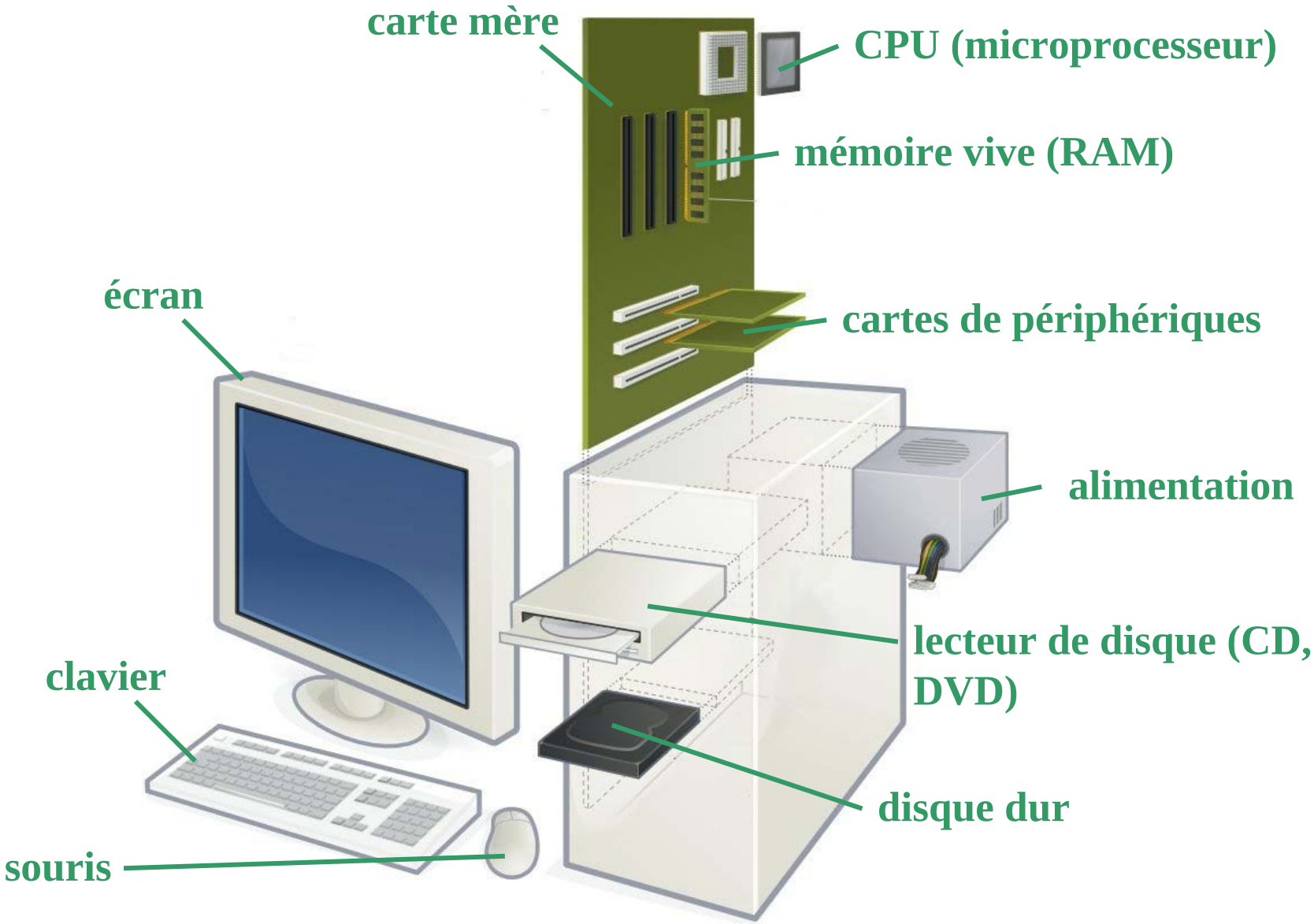
- Ce cours est une introduction à l'algorithmique et la programmation.
  - Programmer vous permet d'effectuer des calculs automatiques complexes
  - Vous apprendrez simultanément à réfléchir (algorithmique) et à programmer dans un langage concret (VBA)
  - L'année prochaine, nous poursuivrons ce cours par un cours de programmation VBA/Excel avancé (objets, formulaires, activeX, …)

- Il est illustré par l'apprentissage du langage VBA pour Excel.
  - VBA pour Excel a été choisi (au lieu de C) pour sa simplicité d'apprentissage par les débutants.
  - Il permet d'écrire des programmes interactifs graphiques exécutés dans le tableur Excel, compétence ludique et utile en entreprises.

Plan du cours 1

1. Information et ordinateurs
    1. Représentation de l'information dans un ordinateur
    2. Codage binaire et hexadécimal
2. Introduction à l'algorithmique
    1. Définition d'un algorithme
    2. De l'algorithme au programme
3. Programmer en VBA avec Excel
    1. Présentation d'Excel
    2. L'éditeur de programmes VBE

Architecture des ordinateurs



Les ordinateurs ne pensent pas, ils calculent !

- Les ordinateurs sont capables de traiter du texte, de jouer de la musique, de projeter des vidéos, de faire tourner des jeux, etc.
- Pourtant, ils ne sont capables que d'une seule chose : **faire des calculs**. C'est le rôle du microprocesseur.
- Lorsqu'un ordinateur traite du texte, de l'image, du son ou de la vidéo, il ne traite qu'une seule chose : **des nombres**.
- Plus simple, il ne connaît pas le nombre 3 ou -10, encore moins 2,51 ou π.
- Un ordinateur manipule exclusivement des **informations binaires.**

Qu'est-ce qu'une information binaire ?

- C'est une information qui ne peut avoir que **deux états**
  - ouvert ou fermé, libre ou occupé, militaire ou civil, assis ou couché, blanc ou noir, vrai ou faux, etc.
- L'information binaire est présente dans le **monde courant**
  - pile chargée ou déchargée, secrétaire absente ou présente, carté perforée ou non perforée, circuit électrique ouvert ou fermé, etc.
- Un **ordinateur** possède différents supports physiques capables de stocker de l'information binaire
  - La mémoire vive (la « RAM ») est formée de millions de composants électroniques qui peuvent retenir ou relâcher une charge électrique.
  - La surface d'un disque dur, d'une bande ou d'une disquette est recouverte de particules métalliques qui peuvent, grâce à un aimant, être orientées dans un sens ou dans l'autre.
  - Sur un CD-ROM, on trouve un long sillon étroit irrégulièrement percé de trous.

Représentation d'une information binaire

- La coutume veut qu'on symbolise une information binaire, quel que soit son support physique, **sous la forme de 1 et de 0**.
- Il faut bien comprendre que ce n'est là qu'une représentation, une image commode, que l'on utilise pour parler de toute information binaire : c'est une **abstraction** utile pour représenter l'information.
- On utilise des abstractions (pas seulement binaires) tous les jours :
  - une mappemonde permet de se représenter la terre
  - la carte du métro permet de se représenter le réseau des stations
  - le chiffre 4 peut avoir différentes représentations : 4 IV 100
  - 4+3=7 est une abstraction mathématique qui évite d'utiliser des morceaux de bois ou des cailloux pour compter

La numérotation de position

- Pour représenter un nombre, aussi grand soit-il, nous disposons d'un alphabet spécialisé : une série de 10 signes qui s'appellent **les chiffres** : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9.

- Lorsque nous écrivons un nombre en mettant certains de ces chiffres les uns derrière les autres, **l'ordre** dans lequel nous mettons les chiffres est capital.
  - 2034 est différent de 4203.

- C'est la **position** des chiffres dans le nombre qui donne sa valeur.
  - $2034 = \mathbf{2}*1000+\mathbf{0}*100+\mathbf{3}*10+\mathbf{4}*1.$
    $= \mathbf{2}*10^3+\mathbf{0}*10^2+\mathbf{3}*10^1+\mathbf{4}*10^0.$

La base décimale

En **base décimale**, deux règles s'appliquent pour l'écriture d'un nombre :

- Règle 1 : nous utilisons un **alphabet numérique de dix symboles**.
  - Nous nous servons de dix chiffres, pas un de plus, pas un de moins.

- Règle 2 : la **position** d'un de ces dix chiffres dans un nombre désigne la **puissance de dix** par laquelle ce chiffre doit être multiplié pour reconstituer le nombre.
  - Si je trouve un 7 en cinquième position à partir de la droite, ce 7 ne représente pas 7 mais $7*10^4$, soit 70 000.
  - Attention au décalage de 1 entre la position et la puissance de 10 ! Si le 7 est en première position, il représente $7*10^0$, soit 7. Ce sont les unités.

La représentation binaire

- La base binaire
  - Les ordinateurs ont des dispositifs physiques faits pour stocker des informations binaires. Lorsqu'on représente une information stockée par un ordinateur, le plus simple est donc d'utiliser un système de représentation à deux chiffres : les fameux 0 et 1. C'est la **base binaire**.

- Les octets
  - Avec un emplacement d'information d'ordinateur, on est limité à deux choses différentes seulement. Avec une telle information binaire, on ne va pas loin. Voilà pourquoi, dès leur invention, les ordinateurs ont été conçus pour manier ces informations **par paquets** de 0 et de 1. Et la taille de ces paquets a été fixée à 8 informations binaires. Ce sont les **octets**.

La représentation binaire

- Une information binaire, symbolisée couramment par 0 ou 1, s'appelle un **bit** (en anglais... bit, pour binary digit).

- Un groupe de huit bits s'appelle un **octet** (en anglais, byte)

- Donc, méfiance avec le byte (en abrégé, **B** majuscule), qui vaut un octet, c'est-à-dire huit bits (en abrégé, **b** minuscule).

La base binaire : décodage

En base binaire, les règles sont les suivantes:

- Règle 1 : nous utilisons un **alphabet numérique de deux chiffres, 0 et 1**.
- Règle 2 : **la position** d'un de ces deux chiffres dans un nombre désigne la **puissance de deux** par laquelle ce chiffre doit être multiplié pour reconstituer le nombre.

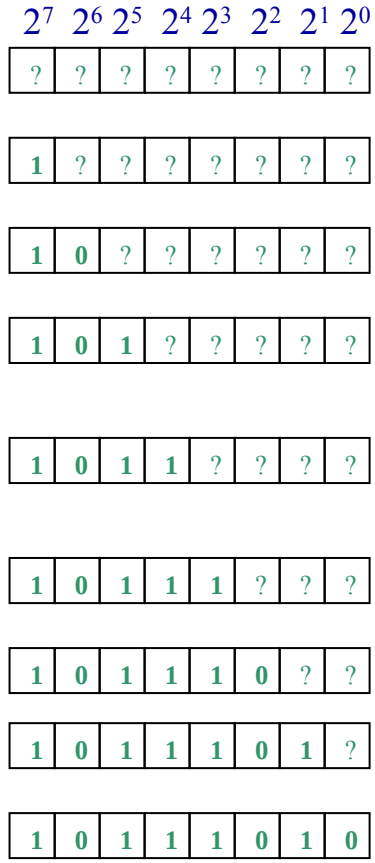
- Décodage de l'octet 10110101
  - $10110101 = \mathbf{1}*2^7+\mathbf{0}*2^6+\mathbf{1}*2^5+\mathbf{1}*2^4+\mathbf{0}*2^3+\mathbf{1}*2^2+\mathbf{0}*2^1+\mathbf{1}*2^0$
    $= 128+32+16+4+1$
    $= 181$

en base 2 **en base 10**

La base binaire : codage (1)

Il suffit de rechercher dans notre nombre les puissances successives de 2.

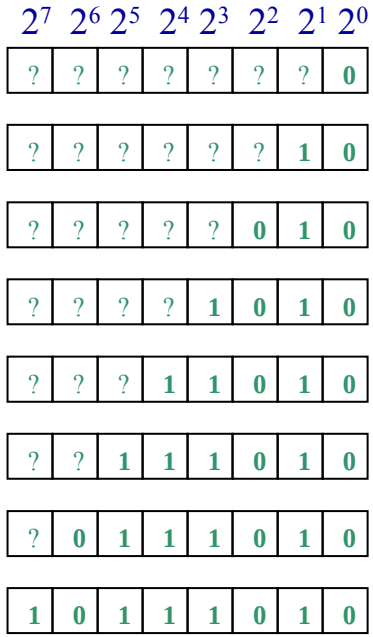
- Codage de 186.
  - 186 est compris entre 0 et 255 donc 186 se code sur 8 bits
  - Dans 186, on trouve 1 x 128, soit 1 x 2^7.
  - Je retranche 128 de 186 et j'obtiens 58. Dans 58, on trouve 0 x 64, soit 0 x 2^6.
  - Je ne retranche donc rien. Dans 58, on trouve 1 x 32, soit 1 x 2^5.
  - Je retranche 32 de 58 et j'obtiens 26. Dans 26, on trouve 1 x 16, soit 1 x 2^4.
  - Je retranche 16 de 26 et j'obtiens 10.Dans 10, on trouve 1 x 8, soit 1 x 2^3. Je retranche 8 de 10 et j'obtiens 2.
  - Dans 2, on trouve 0 x 4, soit 0 x 2^2.
  - Je ne retranche donc rien. Dans 2, on trouve 1 x 2, soit 1 x 2^1.
  - Je retranche 2 de 2 et j'obtiens 0.Dans 0, on trouve 0 x 1, soit 0 x 2^0.
  - Il ne me reste plus qu'à reporter ces différents résultats (dans l'ordre !) pour reconstituer l'octet : en binaire, 186 est représenté par : 10111010



La base binaire : codage (2)

L'autre méthode pour convertir un nombre décimal en base 2 est d'utiliser des successions de divisions entières par le nombre 2, jusqu'à obtenir un quotient de 0. Le résultat est obtenu en lisant les restes obtenus en sens inverse.

- Codage de 186
  - 186/2=93 reste 0
  - 93/2=46 reste 1
  - 46/2=23 reste 0
  - 23/2=11 reste 1
  - 11/2=5 reste 1
  - 5/2=2 reste 1
  - 2/2=1 reste 0
  - 1/2=**0** reste 1
  - fin de la division car quotient nul
  - Soit (**en lisant les restes obtenus en sens inverse**) : 10111010



Octets et choix de codage d'un nombre

- Un octet peut servir à coder 256 nombres différents car chaque bit de l'octet peut occuper deux états
  - 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 = 28 = 256 possibilités
- Différents choix sont possibles pour interpréter un octet
  - la série des nombres entiers de 1 à 256
  - la série des nombres entiers de 0 à 255, comme vu précédemment
  - la série des nombres entiers relatifs de +127 à +128
- C'est une pure affaire de convention, de choix de codage.
- Si l'on veut coder des nombres plus grands que 256, ou des nombres négatifs, ou des nombres décimaux, on va donc être contraint de mobiliser plus d'un octet.
  - Avec deux octets, on a 256 x 256 = 65 536 possibilités.
  - Avec trois octets, on passe à 256 x 256 x 256 = 16 777 216 possibilités.

Octet et codage de texte

- Un octet peut coder un caractère au lieu d'un nombre
  - Il y a 26 lettres dans l'alphabet. Même en comptant différemment les minuscules et les majuscules, en y ajoutant les chiffres et les signes de ponctuation, on arrive à un total inférieur à 256.
- Quel caractère doit être représenté par quel état de l'octet?
  - Si ce choix était librement laissé à chaque informaticien, ou à chaque fabricant d'ordinateur, la communication entre deux ordinateurs serait un véritable casse-tête. L'octet 10001001 serait par exemple traduit par une machine comme un T majuscule, et par une autre comme une parenthèse fermante !
- L'ASCII (American Standard Code for Information Interchange) est un standard international de codage des caractères et des signes de ponctuation.
  - 01001110 est le codage du N (majuscule) et 01101110 du n (minuscule).

Le codage hexadécimal

- Le codage hexadécimal est un codage de position en **base 16**.
  - En base seize, 16 nombres différents se représentent avec un seul symbole, parmi {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}.
- Un octet est représenté par deux valeurs hexadécimales
  - L'octet est décomposé en **deux paquets de 4 bits** : les quatre de gauche et les quatre de droite.
  - Avec 4 bits, nous pouvons coder 2^4= 16 nombres différents.
- Le codage hexadécimal constitue une alternative pratique au codage binaire car il est plus **économique** et plus **lisible**
  - 10011110 en binaire <=> 9E en héxadécimal

Le codage hexadécimal

- Codage en base hexadécimale de 1001111
  1. décomposition en 2 paquets de 4 bits : 1001 1110
  2. $1001 = 2^3+2^0 = 8 + 1 = 9$
  3. $1110 = 2^3+2^2+2^1 = 8 + 4 + 2 = 14$ qui s'écrit E en base 16.
  4. Le nombre s'écrit donc en hexadécimal : 9E

- Décodage de 4A
  - $4A = 4*16^1 + A*16^0$, avec A valant 10 en base décimale
    $= 64 + 10$
    $= 74$

- Pour éviter toute ambiguité, les langages informatiques utilisent une notation préfixée par *0x*
  - $0x41 \neq 41$

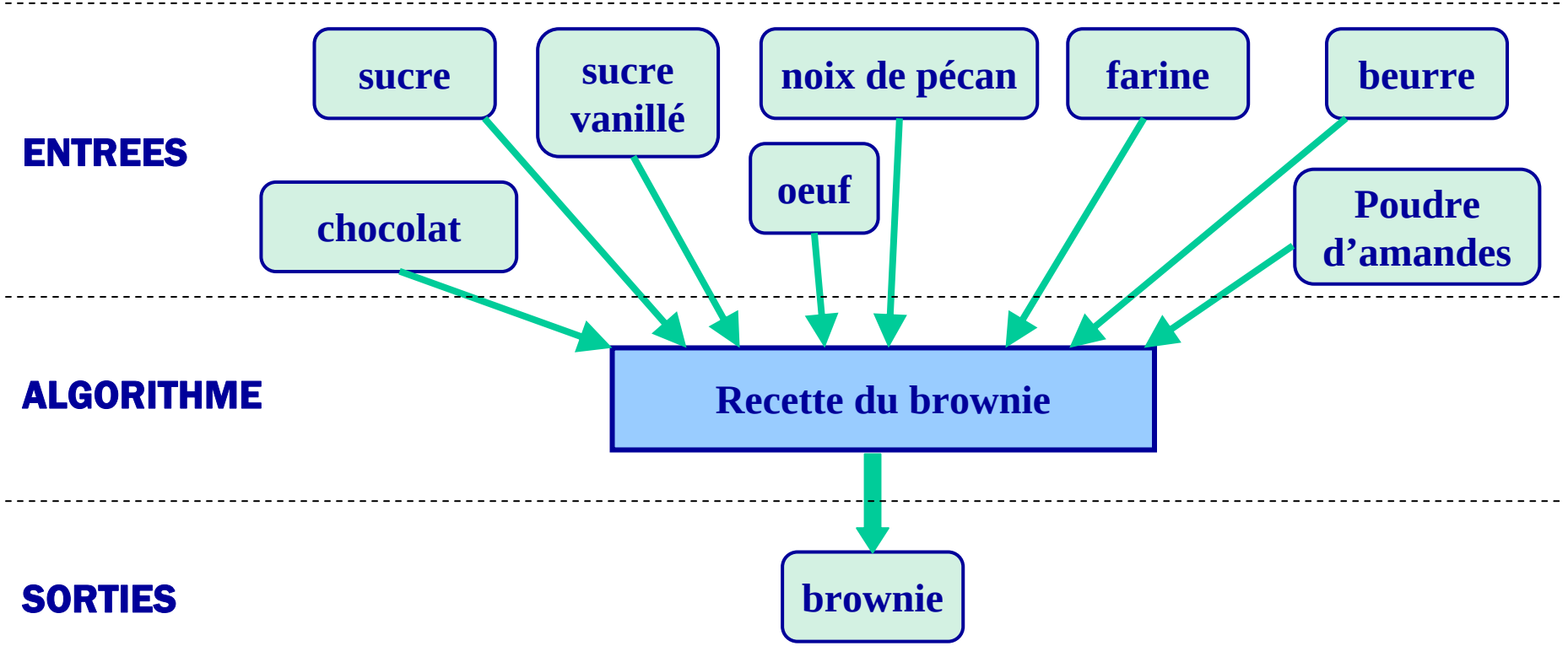
Introduction à l'algorithmique

Qu'est-ce que l'algorithmique ?

- Vous avez déjà **exécuté des algorithmes**
  - Si vous avez-vous déjà ouvert un livre de recettes de cuisine
  - Si vous avez déjà déchiffré un mode d'emploi traduit pour faire fonctionner un magnétoscope ou un répondeur téléphonique réticent
- Vous avez déjà **fabriqué – et fait exécuter – des algorithmes**
  - Si vous avez déjà indiqué un chemin à un touriste égaré
  - Si vous avez fait chercher un objet à quelqu'un par téléphone
- L'algorithmique est une aptitude partagée par la totalité de l'humanité et qui ne s'applique pas seulement à l'informatique

Définition d'un algorithme

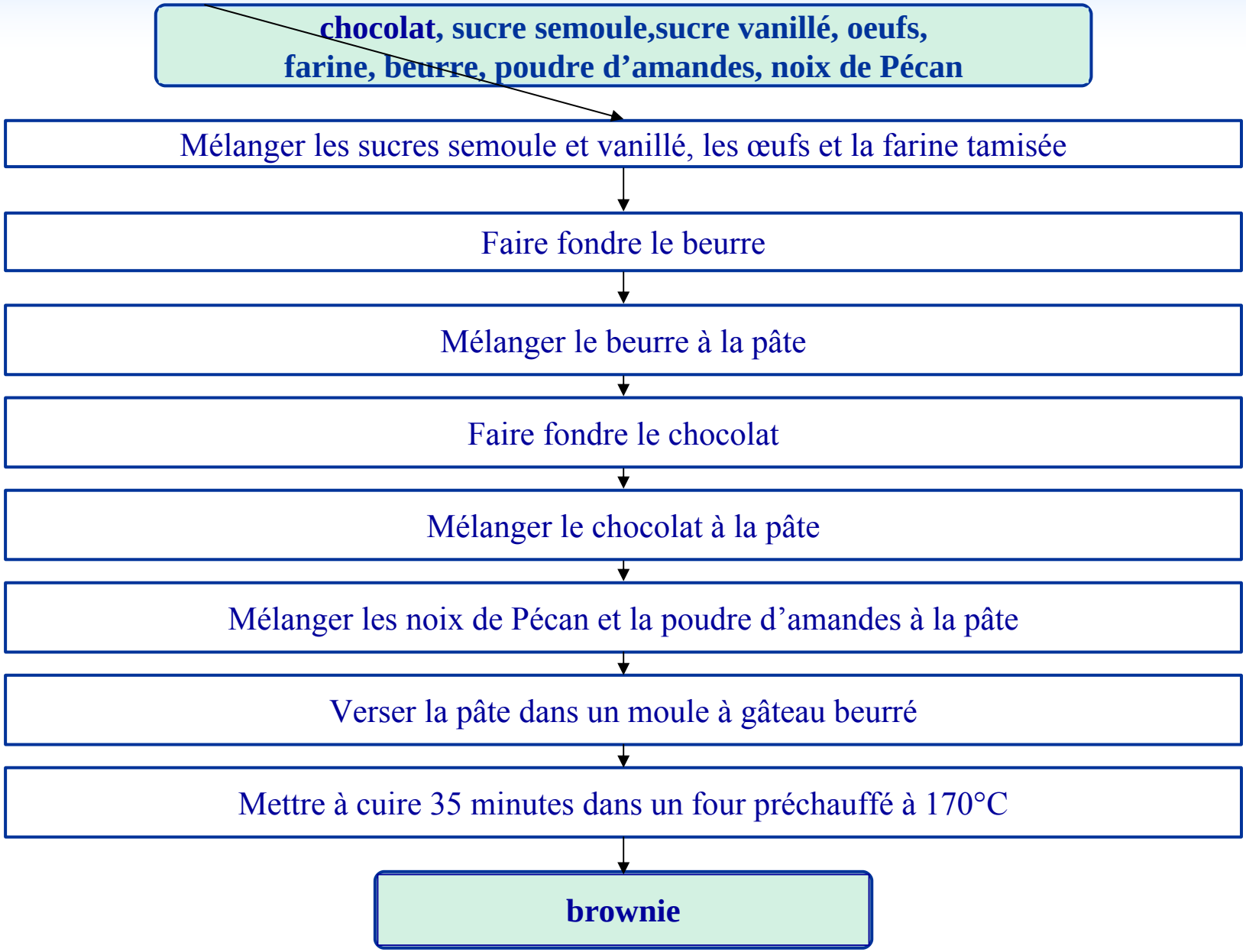
- On peut comparer un algorithme à une recette de cuisine. Les ingrédients nécessaires à la réalisation du gâteau sont les données ou **entrées**. L'**algorithme** est une **suite d'actions** qui produira des résultats ou **sorties**.
- La recette du brownie



Suite d'actions d'un algorithme

- Les actions (ou **instructions**) qui composent un algorithme sont **séquentielles**
  - Les actions se suivent et doivent être réalisées l'une après l'autre,
  - On ne peut pas faire cuire le brownie avant d'avoir mis la pâte dans le moule.
- L'algorithme du brownie :
  1. Mélanger les sucres semoule et vanillé, les oeufs et la farine tamisée
  2. Faire fondre le beurre
  3. Mélanger le beurre à la pâte
  4. Faire fondre le chocolat
  5. Mélanger le chocolat à la pâte
  6. Mélanger les noix de Pécan et la poudre d'amande à la pâte
  7. Versez la pâte dans un moule à gâteau beurré
  8. Mettre à cuire 35 minutes dans le four préchauffé à 170°C.

Représentation sous forme d'organigramme



Faut-il être matheux pour être bon en algo ?

- Cette opinion sert régulièrement d'excuse
  - "Moi, de toute façon, je suis mauvais(e) en algo, j'ai jamais rien pigé aux maths ". Mais faut-il être " bon en maths " pour expliquer correctement son chemin à quelqu'un ?

- La maîtrise de l'algorithmique requiert deux qualités :
  - il faut être **rigoureux**. Chaque fois qu'on écrit une série d'instructions, il faut se mettre à la place de la machine qui va les exécuter pour vérifier si le résultat obtenu est bien celui que l'on voulait. Mais cette opération ne requiert pas la moindre once d'intelligence, uniquement d'être méthodique.
  - il faut avoir **une certaine intuition.** Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. C'est là, si l'on y tient, qu'intervient la forme "d'intelligence" requise pour l'algorithmique. Alors, c'est certain, il y a des gens qui ont au départ davantage cette intuition que les autres. Mais… mais d'une part, les réflexes, cela s'acquiert, et l'expérience finit par compenser largement bien des intuitions.

De l’algorithme au programme

Définition d'un programme informatique

- Ecrire un programme revient à écrire un algorithme dans un langage compréhensible par un ordinateur.
  - Un **programme informatique** est une liste d'ordres indiquant à un ordinateur ce qu'il doit faire. Il se présente sous la forme d'une ou plusieurs séquences d'**instructions** devant être exécutées dans un certain ordre par un **processeur**, et comportant souvent des **données** d'entrées chargées dans la **mémoire vive**.
- Un programme informatique est écrit dans un langage de programmation.
  - Le langage permet à la personne qui rédige un programme de **faire abstraction de certains mécanismes internes**, généralement des activations et désactivations de commutateurs électroniques, qui aboutissent au résultat désiré.
  - Contrairement à un algorithme qui s'écrit dans un langage de pseudo-code, un programme doit être **correct syntaxiquement** (pas de fautes d'orthographe !).
  - Un même algorithme peut être écrit dans des dizaines de langages de programmations différents.

Langages de programmation

- Le **langage machine**
  - C'est la suite de bits qui est interprétée par le processeur de l'ordinateur lors de l'exécution d'un programme, chaque instruction étant caractérisée par un code d'opération.
  - Exemple en MIPS : `000000 00001 00010 00110 00000 100000` signifie " ajouter les registres* 1 et 2 et placer le résultat dans le registre 6"

- Le **langage d'assemblage** (ou simplement **assembleur**)
  - Langage proche du langage machine qui peut être directement interprété par le processeur de l'ordinateur tout en restant lisible par un humain, créé pour faciliter le travail des programmeurs. Il consiste à représenter les combinaisons de bits employées en langage binaire par des symboles.
  - Exemple : `mov 1,$0x61` signifie "mettre la valeur hexadécimale 61 dans le registre 1 ".

*registre = case mémoire propre au processeur.

Langages de programmation

- **Le langage C**
  - Langage d'assemblage de haut niveau qui offre une syntaxe plus lisible que l'assembleur tout en permettant de gardant la maîtrise des ressources utilisées (ex : l'usage de la mémoire). Utilisé pour programmer les systèmes d'exploitation (ex : Linux), les systèmes embarqués (ex : lecteur mp3), les calculs intensifs, …
  - Exemple : `int i = 3;` signifie "réserver un emplacement mémoire de la taille d'un entier, nommer-le i, et y mettre la valeur 3"

- **Le Basic**
  - Le BASIC (***B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode*) a été conçu à la base en 1963 pour permettre aux étudiants ne travaillant pas dans des filières scientifiques d'utiliser les ordinateurs (et même de programmer sa calculatrice), ne nécessitant pas la compréhension du matériel de l'ordinateur.
  - Exemple : `INPUT "Quel est votre nom ?"; NOM$` signifie "afficher le message *Quel est votre nom ?* dans le terminal et sauvegarder la réponse dans un emplacement mémoire qu'on appellera NOM."

Langages de programmation

- **Le Visual Basic (VB)**
  - Visual Basic est un environnement de développement intégré propriétaire pour le langage BASIC sous Windows, édité par Microsoft.
  - Le Visual Basic est une évolution des Basics précédents de Microsoft (Basica, GW Basic, Qbasic) qui permet de créer des applications fenêtrées/ Bénéficiant de la simplicité du Basic originel, il permet de créer des programmes relativement rapidement. Le programmeur en Visual Basic manipule des éléments visuels à l'écran auxquels il ne reste plus qu'à associer du code.

- **Le Visual Basic for Application (VBA)**
  - Le langage de programmation des applications de Microsoft Office (Excel, Word, Access, PowerPoint, FrontPage, Outlook, …)
  - VBA permet d'automatiser les tâches, de créer des applications complètes, de sécuriser vos saisies et vos documents, de créer de nouveaux menus et de nouvelles fonctions pour personnaliser et « booster » votre logiciel.

De l'algorithme au programme

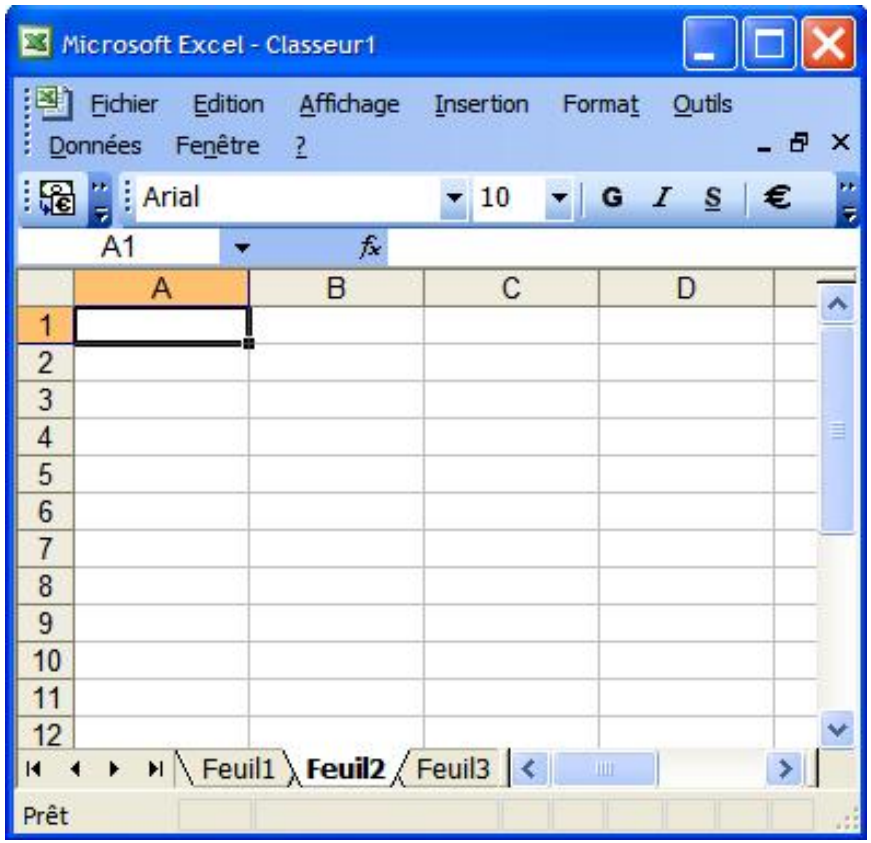
- Pourquoi apprendre l'algorithmique pour apprendre à programmer ? En quoi a-t-on besoin d'un langage spécial, distinct des langages de programmation compréhensibles par les ordinateurs ?
  - L'algorithmique exprime les instructions résolvant un problème donné **indépendamment des particularités de tel ou tel langage**.
  - Pour prendre une image, si un programme était une dissertation, l'algorithmique serait le plan, une fois mis de côté la rédaction et l'orthographe. Or, vous savez qu'il vaut mieux faire d'abord le plan et rédiger ensuite que l'inverse…

- Dans ce cours, nous utiliserons le langage **VBA pour Excel** pour programmer (on dit aussi **coder**) nos algorithmes.
  - Langage simple avec environnement graphique
  - Aide à la saisie et à la correction (débogage) intégrée

Programmer en VBA avec Excel

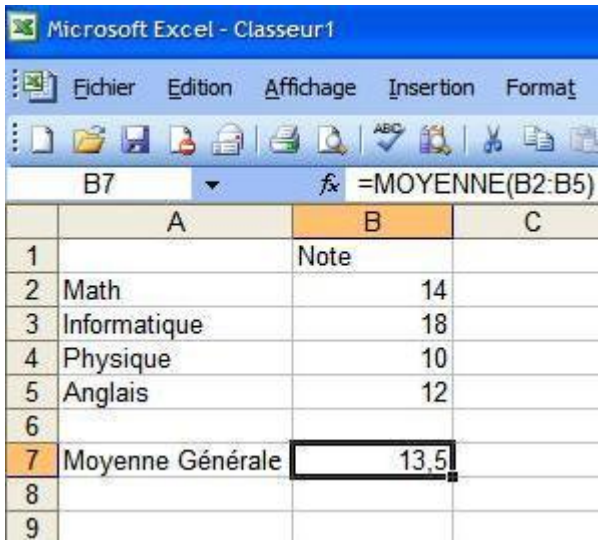
Présentation d'Excel

- Excel = Tableur
  - Collection de feuilles (*sheet*) composées de cellules (*cells*).
  - Un cellule est l'intersection entre une lignes et une colonne. Ex : A1, B15
  - Les feuilles sont regroupées en classeurs (*workbook*).

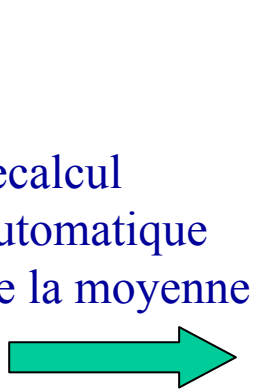


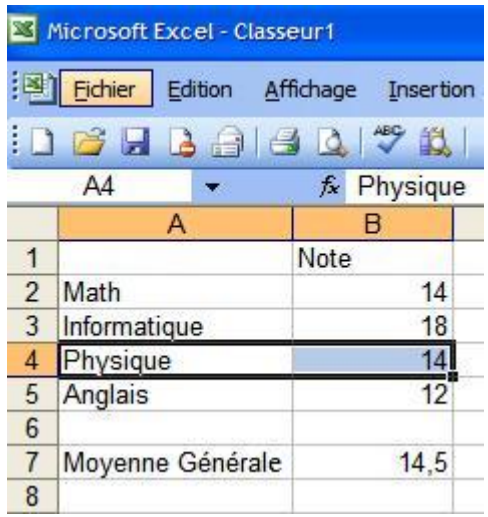
Présentation d'Excel

- Utilisation d'Excel
  - Organisation et stockage des données. *Exemple : les notes des étudiants.*
  - Calculs automatiques. *Exemple : la note moyenne d'un groupe d'étudiants.*
  - Production automatique de graphiques. *Exemple : dessiner un histogramme de répartition des notes.*



recalcul automatique de la moyenne



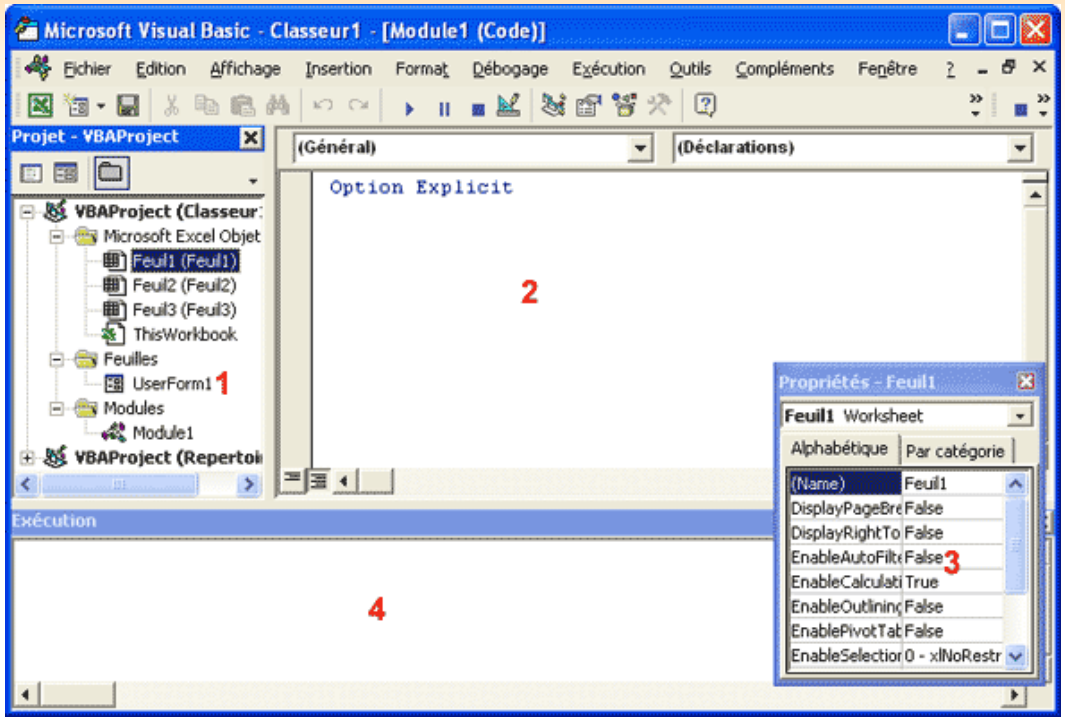


Editeur de macro Excel

- Macro = programme VBA
  - Enregistrement de toutes les commandes effectuées par l'utilisateur pour éviter la répétition d'une tâche
  - Mais ce n'est rien d'autre qu'un programme écrit en VBA.
  - Avant qu'Excel n'utilise ce langage de programmation, le logiciel utilisait son propre langage de programmation et une application était appelée « macro », pour macro-commande. Ce terme est resté.

- Editeur de code ou VBE (Visual Basic Editor)
  - C'est l'environnement de programmation de VBA. Il se lance par le menu « Outils > Macro > Visual-Basic-Editor » ou par le raccourci clavier « Alt+F11 » .
  - **Une fois dans VBE, sélectionner un texte et obtenez l'aide avec F1.**

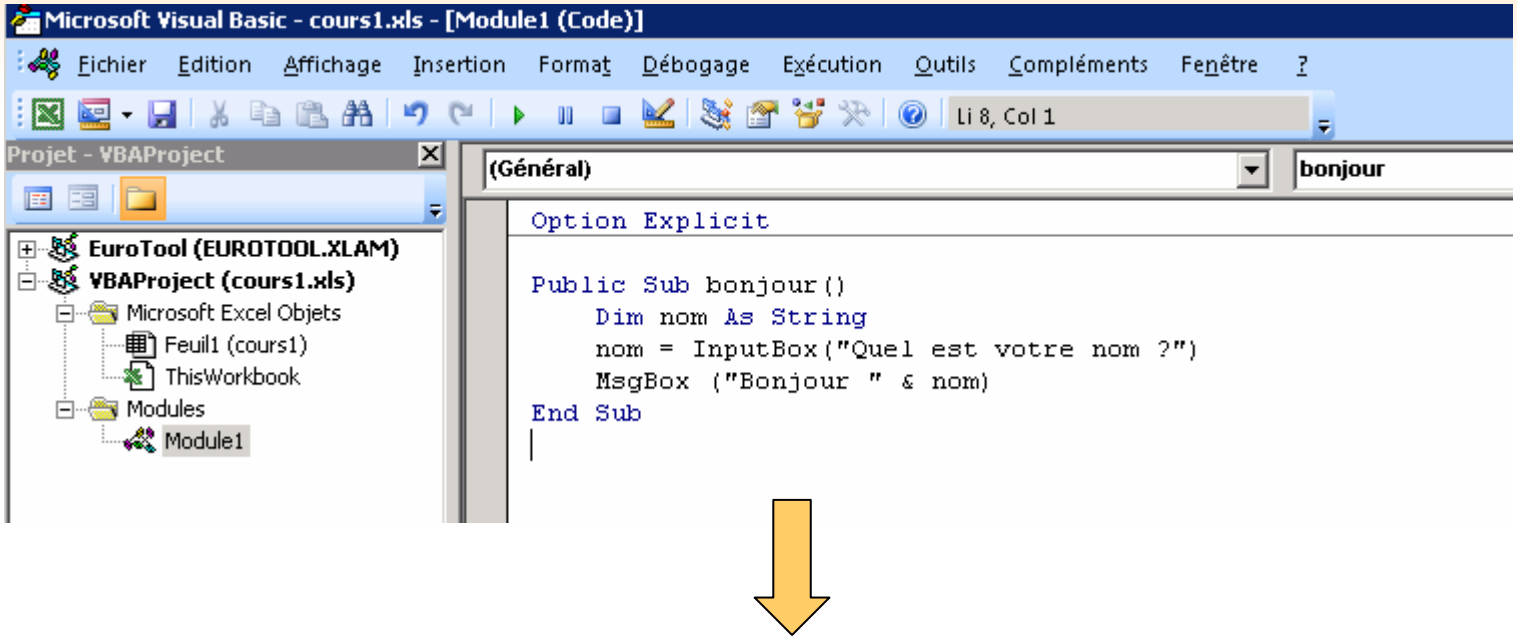
Editeur de code VBE



- **1 - Fenêtre VBAProject.** Elle présente les différents projets ouverts et permet de naviguer facilement entre vos différentes feuilles de codes VBA.
- **2 - Fenêtre Code**. C'est l'endroit ou vous allez saisir votre code VBA.
- **3 - Fenêtre Propriétés.** Propriétés de l'objet sélectionné.
- **4 - Fenêtre Exécution.** Elle permet de tester une partie du code. Elle peut s'avérer très utile pour voir comment s'exécutent certaines lignes de code.

Exemple de programme VBA

Ecriture du programme dans VBE



Exécution du programme dans Excel