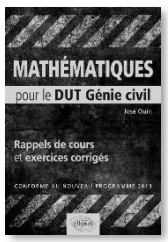
José OUIN
Ingénieur INSA Toulouse
Ancien élève de l'ENS Cachan
Professeur Agrégé de Génie civil
Professeur Agrégé de Mathématiques

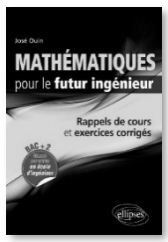
Excel en Jeux : 10 jeux interactifs codés en VBA

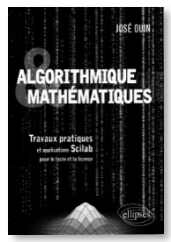
Maîtriser VBA en jouant et en programmant

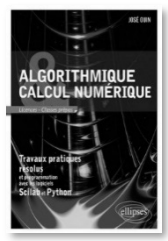
www.joseouin.fr

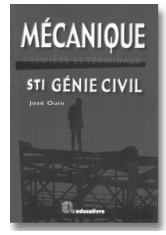
Du même auteur aux Editions Ellipses et Educalivre

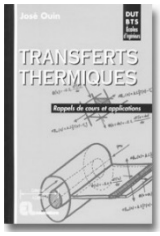


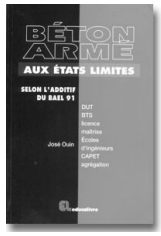












ISBN : 978-2-9592760-9-5

Avant-Propos

Cet ouvrage, ainsi que les classeurs Excel disponibles en téléchargement sur `joseouin.fr`, ont été conçus avant tout pour vous offrir une expérience ludique et divertissante. Que vous soyez un amateur de jeux de stratégie ou simplement à la recherche d'une nouvelle façon de passer le temps, vous trouverez ici des jeux captivants à jouer directement dans Excel.

Pour ceux d'entre vous qui souhaitent aller au-delà du simple plaisir de jouer, cet ouvrage offre une opportunité unique de découvrir le langage VBA (Visual Basic for Applications) à travers des cas concrets de jeux de stratégie. Chaque chapitre vous guide pas à pas dans la création de jeux, illustrant les concepts de programmation VBA de manière claire et pratique. Vous apprendrez comment manipuler des feuilles de calcul, créer des interfaces utilisateur interactives et développer des algorithmes pour des jeux complexes.

Pour les lecteurs déjà familiers avec le langage VBA, cet ouvrage représente une mine d'or d'exemples pratiques et de code source bien documenté. Vous aurez l'occasion d'examiner en détail le code de chaque jeu, de comprendre les mécanismes qui les sous-tendent et, si vous le souhaitez, de modifier certains paramètres pour personnaliser les jeux selon vos préférences. Que vous vouliez ajuster les règles d'un jeu, changer son apparence ou ajouter de nouvelles fonctionnalités, les possibilités sont infinies.

Mon objectif avec ce livre est de vous faire découvrir les immenses possibilités offertes par le langage VBA pour Excel. Trop souvent perçu comme un simple outil de gestion de données et de rapports, Excel, combiné à VBA, se révèle être une plateforme puissante pour la création de jeux interactifs et stimulants. En explorant les exemples de jeux de stratégie présentés ici, j'espère que vous serez inspirés à expérimenter et à créer vos propres applications innovantes.

Je vous invite à plonger dans les pages qui suivent avec curiosité et enthousiasme. Que vous soyez ici pour jouer, apprendre ou personnaliser, j'espère que cet ouvrage vous apportera autant de plaisir que j'en ai eu à l'écrire.

Bonne lecture et bon jeu !

José Ouin

Tous les classeurs Excel des jeux étudiés dans cet ouvrage sont disponibles en téléchargement sur le site Internet : `joseouin.fr`
Merci de consulter les instructions de téléchargement à la fin de cet ouvrage.

Table des matières

Première partie

Les prérequis essentiels

Deuxième partie

Les jeux de stratégie

Le jeu : Puissance 4 29

Le jeu : Morpions ou Tic-tac-toe 39

Le jeu : Le compte est bon 49

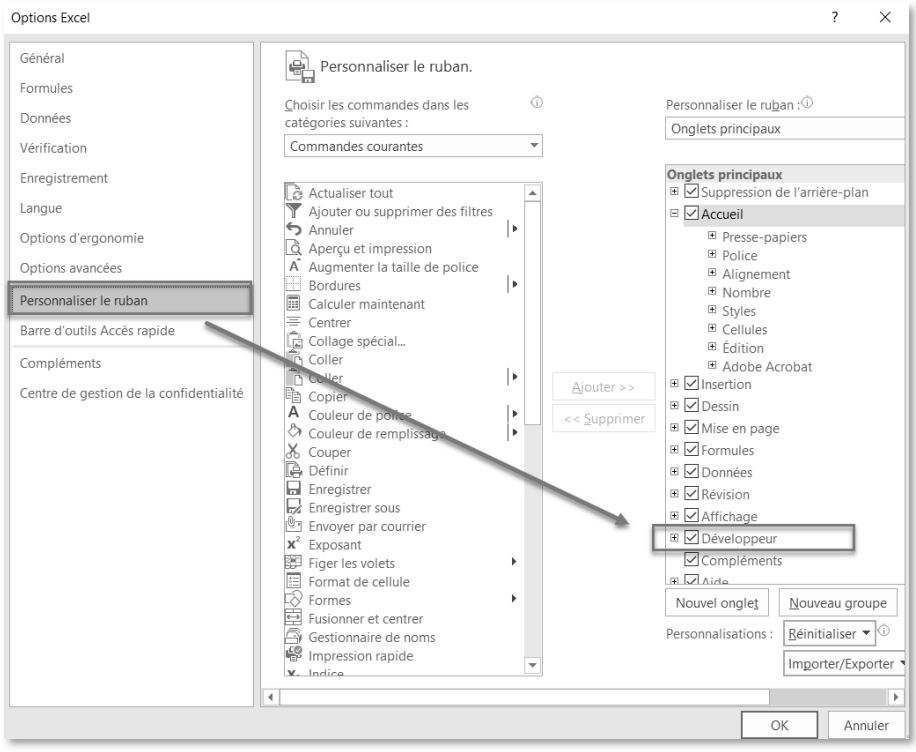
Première partie

Les prérequis essentiels

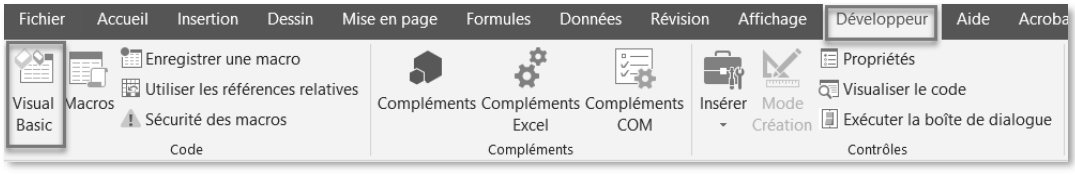
Excel et l'éditeur Visual Basic

1- Accès à l'éditeur Visual Basic

Affichage du ruban « Développeur » : Cliquez sur « **Fichier/Options/Personnaliser le ruban** » puis cocher « **Développeur** ».



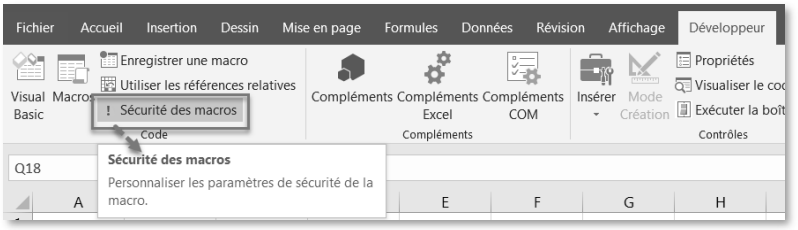
- **Utilisation du ruban Excel :**
  - **Onglet "Développeur" :** L'onglet "Développeur" dans le ruban Excel donne accès à divers outils de développement, y compris l'éditeur VBA.
  - **Bouton "Visual Basic" :** En cliquant sur le bouton "Visual Basic", l'éditeur VBA s'ouvre, offrant un environnement dédié à la programmation.

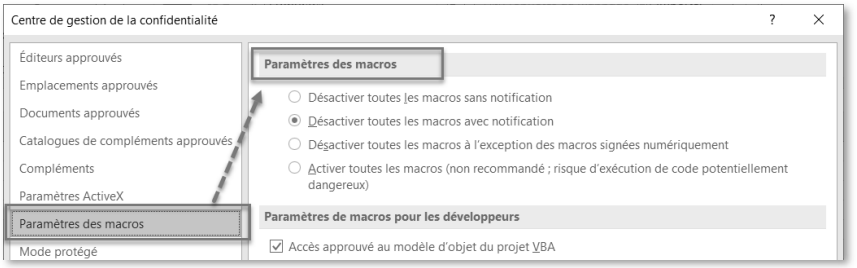


- **Raccourcis clavier :**
  - **Alt + F11 :** Une autre méthode d'accès rapide à l'éditeur VBA consiste à utiliser le raccourci clavier "Alt + F11". Ce raccourci bascule rapidement entre Excel et l'éditeur VBA.

2- Sécurité des macros

Le bouton de menu « Sécurité des macros » du ruban « Développeur » permet de régler les paramètres de sécurité des macros :





Désactiver toutes les macros sans notification :

- Description : Avec cette option activée, Excel désactive automatiquement toutes les macros sans vous avertir.
- Impact sur la programmation VBA : Les macros, même celles provenant de sources de confiance, ne seront pas exécutées sans notification explicite.

Désactiver toutes les macros avec notification :

- Description : Excel demande à l'utilisateur si les macros doivent être exécutées ou non dans le classeur qui a été ouvert.
- Impact sur la programmation VBA : Dès l'ouverture du classeur Excel, les utilisateurs seront avertis que des macros existent. Ils peuvent choisir de les exécuter ou non.

Désactiver toutes les macros à l'exception des macros signées numériquement :

- Description : Cette option empêche l'exécution de toutes les macros, sauf celles qui ont été signées numériquement par un éditeur de confiance.
- Impact sur la programmation VBA : Si cette option est activée, seules les macros signées numériquement pourront être exécutées, ce qui peut ajouter une couche de sécurité supplémentaire.

Activer toutes les macros (non recommandé, risque de sécurité potentiel) :

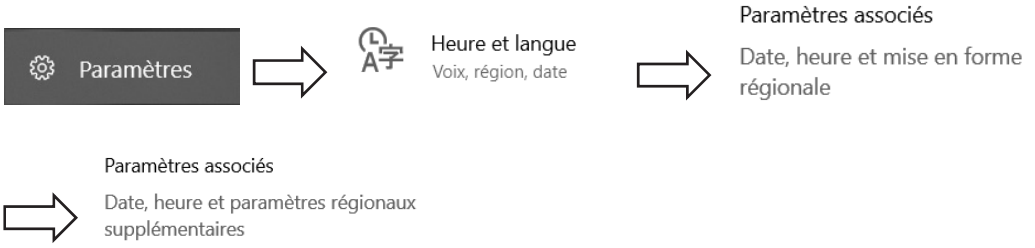
- Description : Cette option permet l'exécution de toutes les macros sans aucune restriction ni notification.
- Impact sur la programmation VBA : Toutes les macros, y compris celles potentiellement dangereuses, seront exécutées **sans avertissement**.

Il est important de noter que la dernière option ("Activer toutes les macros") est considérée comme non recommandée en raison des risques potentiels pour la sécurité.

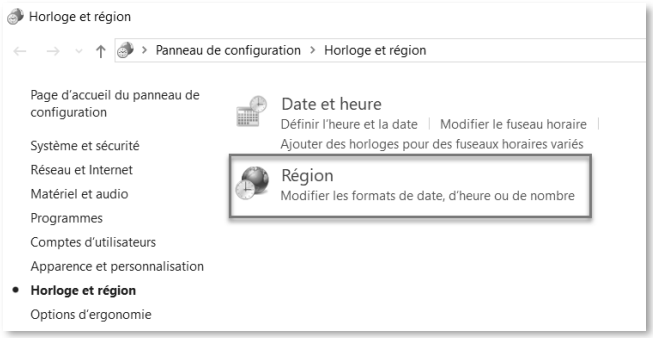
3- Symbole décimal : réglage des paramètres régionaux

Le symbole décimal est fonction du réglage des paramètres régionaux qui ont été définis dans l'environnement Windows : le point « **.** » ou la virgule « **,** ».

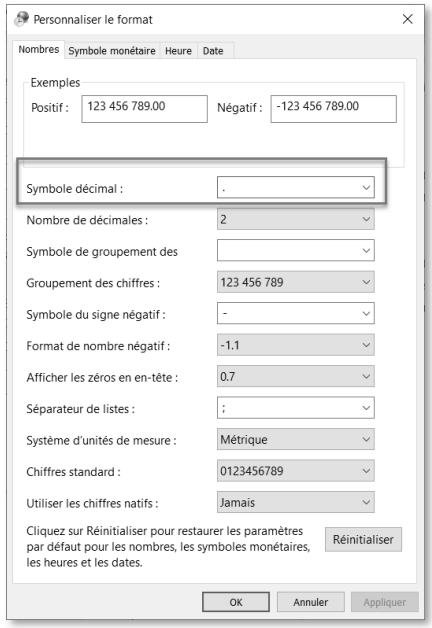
Cliquez sur « Paramètres »



On sélectionne alors « Région » :



On choisit le symbole décimal « . » ou « , » :



Deuxième partie

Les jeux de stratégie

Premier jeu

Puissance 4

Le jeu Puissance 4 a une histoire intéressante ! Créé en 1974 par Howard Wexler, un ingénieur américain, et Ned Strongin, un écrivain, Puissance 4 était à l'origine appelé "Connect Four" aux États-Unis. Le jeu a été commercialisé par la société Milton Bradley (qui fait maintenant partie de Hasbro) en 1974.

L'idée du jeu est née d'une observation simple : Wexler a remarqué que les enfants utilisaient souvent des crayons pour aligner quatre pions dans une rangée lorsqu'ils jouaient au "Morris", un jeu de stratégie plus ancien. En s'inspirant de cela, lui et Strongin ont développé le concept de Puissance 4, un jeu de stratégie simple mais captivant.

Le jeu s'est rapidement répandu dans le monde entier, devenant un classique des jeux de société. Il a été décliné sous différentes formes et a inspiré de nombreuses variantes et adaptations. En raison de sa simplicité et de son attrait universel, Puissance 4 reste populaire même des décennies après sa création.

Le jeu : Puissance 4

1- Présentation et règles du jeu

1-1. Présentation

Le jeu Excel-Puissance 4 est un jeu de stratégie classique qui se joue sur une grille verticale de 6 lignes sur 7 colonnes. Le but du jeu est d'aligner quatre jetons de votre couleur (horizontalement, verticalement ou en diagonale) avant votre adversaire.

1-2. Déroulement du jeu

- Grille de jeu : La grille de jeu est constituée de 6 lignes et 7 colonnes. Chaque joueur choisit une couleur de jeton, généralement rouge ou jaune.
- Placement des jetons : Les joueurs alternent pour placer un jeton de leur couleur dans une colonne. Le jeton tombe au bas de la colonne disponible.
- Objectif : Le premier joueur qui parvient à aligner quatre jetons de sa couleur dans n'importe quelle direction (horizontale, verticale ou diagonale) remporte la partie.
- Victoire : Le jeu détecte automatiquement lorsque quatre jetons de la même couleur sont alignés et affiche un message indiquant le joueur gagnant.
- Match nul : Si la grille est remplie sans qu'aucun joueur n'aligne quatre jetons, la partie est déclarée nulle.

Le jeu Excel-Puissance 4 est un jeu simple à comprendre mais qui demande une stratégie réfléchie pour anticiper les mouvements de l'adversaire et créer des opportunités de victoire. Il peut être joué entre deux joueurs ou contre un ordinateur avec une implémentation VBA dans Excel.

1-3. Déroulement du jeu avec Excel

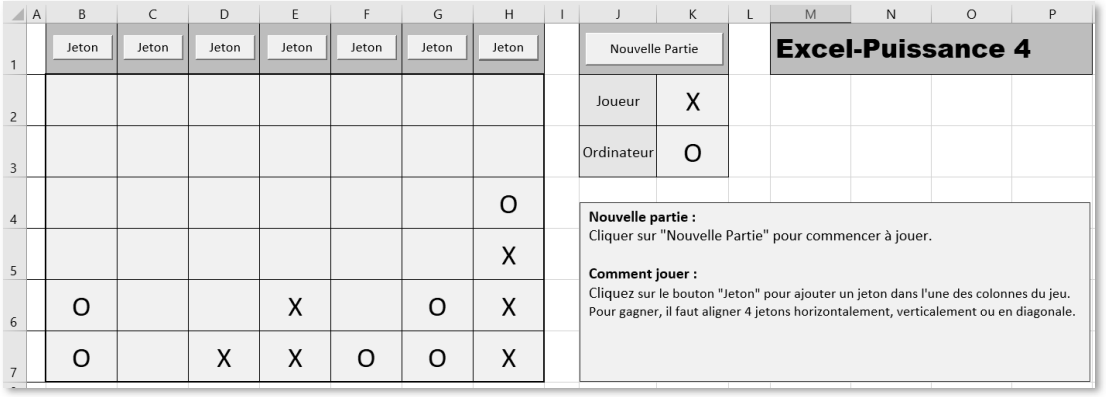
Dans la version Excel de « Puissance 4 », la grille de jeu est représentée par une feuille de calcul avec des boutons qui représentent les colonnes. Chaque colonne contient un bouton « **Jeton** » qui permet de faire tomber un jeton dans la colonne correspondante.

Lorsque c'est au tour du joueur, il clique sur le bouton « **Jeton** » de la colonne où il souhaite placer son jeton. Le jeton du joueur est symbolisé par un « **X** ». Une fois que le joueur a cliqué sur un bouton « **Jeton** », le jeton « **X** » tombe dans la colonne choisie jusqu'à ce qu'il atteigne la première case vide.

De même, lorsque c'est au tour de l'ordinateur, il effectue automatiquement son coup en choisissant une colonne de manière aléatoire. Le jeton de l'ordinateur est symbolisé par un « **O** ». Une fois que l'ordinateur a effectué son coup, le jeton « **O** » tombe dans la colonne choisie jusqu'à ce qu'il atteigne la première case vide.

Le jeu se poursuit ainsi jusqu'à ce qu'un joueur parvienne à aligner quatre de ses jetons horizontalement, verticalement ou en diagonale, ou jusqu'à ce que la grille soit entièrement remplie, auquel cas la partie est déclarée nulle.

Capture d'écran de la feuille Excel



2- Le code VBA pour Excel

Le code VBA ci-après présente les différentes procédures (Sub) et fonctions (Function) nécessaires à la mise en œuvre de ce jeu. Le code est commenté afin de permettre aux lecteurs de comprendre le fonctionnement des différentes macros VBA.
Chaque jeu présenté est accompagné d'un classeur Excel téléchargeable depuis le site Internet de l'auteur : joseouin.fr. À la fin de cet ouvrage, une page explique la démarche à suivre pour obtenir l'ensemble des classeurs Excel correspondant à tous les jeux étudiés. Vous pouvez ainsi approfondir votre compréhension des stratégies et des mécaniques de jeu grâce à ces ressources pratiques.

2-1. Présentation du code VBA

Stratégie de l'ordinateur :
L'IA de l'ordinateur utilise une approche basée sur l'analyse des coups possibles pour déterminer la meilleure colonne où placer son jeton. Elle cherche d'abord à gagner la partie en vérifiant s'il existe une colonne où un alignement de quatre jetons est possible dans le prochain coup. Si ce n'est pas le cas, elle essaie de bloquer les tentatives de victoire de l'adversaire en identifiant les colonnes où celui-ci pourrait gagner dans le prochain coup. Enfin, si aucune victoire n'est imminente pour l'un ou l'autre joueur, l'ordinateur choisit une colonne de manière aléatoire parmi celles qui sont disponibles.

Liste des procédures et des fonctions

- **InitialiserGrille :**

Cette procédure initialise la grille de jeu en attribuant la valeur 0 à chaque case (représentant une case vide) et efface le contenu des cellules dans la feuille Excel. Elle désigne également le joueur humain comme le joueur actuel pour commencer la partie.

- **PlacerJeton :**

Cette procédure place un jeton dans une colonne spécifiée. Elle trouve d'abord la première case vide dans la colonne et place le jeton du joueur actuel. Ensuite, elle vérifie s'il y a une victoire et change de joueur si nécessaire.

- **JouerOrdinateur :**

Cette procédure permet à l'ordinateur de jouer. Elle essaie d'abord de trouver la meilleure colonne pour gagner. Si cela n'est pas possible, elle essaie de bloquer le joueur humain. Enfin, si aucune des deux options n'est disponible, l'ordinateur joue aléatoirement.

- **VerifierVictoire :**

Cette fonction vérifie s'il y a une victoire en parcourant la grille et en vérifiant les lignes, les colonnes et les diagonales.

- **ColonnePleine :**

Cette fonction vérifie si une colonne spécifiée est pleine en regardant si la première case de la colonne est vide ou non.

- **TrouverMeilleureColonne :**

Cette fonction trouve la meilleure colonne pour le joueur spécifié. Elle simule les coups possibles pour le joueur et retourne la colonne qui entraînera une victoire immédiate si elle existe.

- **CliquerColonne :**

Cette procédure est exécutée lorsqu'un joueur clique sur une colonne de la grille dans l'interface Excel. Elle détermine la colonne cliquée et place un jeton dans cette colonne.

- **EffacerGrille :**

Cette procédure efface la grille de jeu et réinitialise le jeu pour une nouvelle partie en appelant la procédure d'initialisation de la grille.

2-2. Code VBA du jeu

```
Option Explicit
```

' Définition des constantes pour les joueurs

```
Const JoueurHumain As Integer = 1
Const JoueurOrdinateur As Integer = 2

' Déclaration de la grille de jeu
Dim grille(1 To 6, 1 To 7) As Integer

' Variable pour suivre le joueur actuel
Dim joueurActuel As Integer
```

' Procédure pour initialiser la grille de jeu

```
Sub InitialiserGrille()
    Dim i As Integer, j As Integer
    For i = 1 To 6
        For j = 1 To 7
            grille(i, j) = 0 ' 0 représente une case vide
           Cells(i + 1, j + 1).Value = "" ' Efface le contenu des cellules
        Next j
    Next i
    joueurActuel = JoueurHumain ' Le joueur humain commence la partie
End Sub
```

' Procédure pour placer un jeton dans une colonne

```
Sub PlacerJeton(colonne As Integer)
    Dim ligne As Integer
   ' Trouver la première case vide dans la colonne
    For ligne = 6 To 1 Step -1
        If grille(ligne, colonne) = 0 Then
              grille(ligne, colonne) = joueurActuel ' Placer le jeton du
joueur actuel
            If joueurActuel = JoueurHumain Then
                 Cells(ligne + 1, colonne + 1).Value = "X" ' Affiche un
"X" pour le joueur humain
```

Cinquième jeu

Sudoku

Qu'est-ce que le Sudoku ?
Le Sudoku est un jeu de réflexion et de logique inventé au Japon dans les années 1980. Son nom, **Sudoku**, signifie littéralement "**chiffres seuls**" en japonais. Le jeu consiste à remplir une grille de 9x9 cases divisée en neuf régions plus petites de 3x3 cases avec des chiffres de 1 à 9, de telle sorte que chaque ligne, chaque colonne et chaque région contienne tous les chiffres de 1 à 9 sans répétition. Ce défi apporte une satisfaction intellectuelle sans pareille à chaque grille résolue.

Pourquoi le Sudoku sur Excel ?
La version du Sudoku sur Excel combine la simplicité de la feuille de calcul Excel avec la puissance des macros VBA (Visual Basic for Applications), offrant une expérience de jeu fluide et interactive. Grâce à cette version, vous pouvez désormais profiter du Sudoku directement depuis votre ordinateur, sans avoir besoin de crayons ni de papier.

1					7		9	
	3			2				8
		9	6			5		
		5	3			9		
	1			8				2
6					4			
3							1	
	4							7
		7				3		

Le jeu : Sudoku

1- Présentation et règles du jeu

1-1. Présentation

Le Sudoku est un jeu de logique qui se joue sur une grille de 9x9 cases, divisée en neuf régions de 3x3 cases appelées également blocs ou sous-grilles. L'objectif est de remplir toutes les cases de la grille avec des chiffres de 1 à 9, en veillant à ce que chaque chiffre n'apparaisse qu'une seule fois dans chaque ligne, chaque colonne et chaque région de 3x3 cases.

Jouez au Sudoku avec Excel : Résolvez, créez et défiez-vous !
Le Sudoku, ce casse-tête logique addictif, trouve maintenant sa place dans Excel pour vous offrir une expérience de jeu riche et personnalisable.

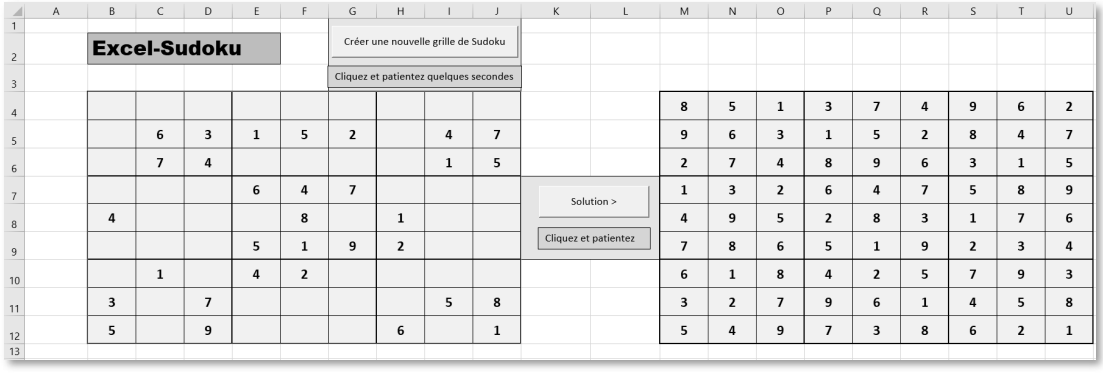
1-2. Déroulement et règles du jeu avec Excel

Résolvez vos propres grilles : Importez votre grille Sudoku préférée depuis un magazine ou tout autre support et mettez-vous au défi de la résoudre directement dans Excel. Testez vos compétences et voyez si vous avez ce qu'il faut pour venir à bout des grilles les plus corsées.

Créez des grilles personnalisées : Laissez Excel générer des grilles Sudoku uniques et personnalisées pour vous. Avec Excel comme votre générateur de Sudoku, l'aventure est sans fin !

Laissez Excel résoudre pour vous : Besoin d'un coup de pouce ? Laissez Excel prendre les commandes et résoudre une grille que vous trouvez particulièrement difficile. Obtenez instantanément la solution et découvrez les techniques de résolution utilisées par Excel pour affiner vos propres compétences.

Capture d'écran de la feuille Excel



2- Code VBA pour Excel

Le code VBA ci-après présente les différentes procédures (Sub) et fonctions (Function) nécessaires à la mise en œuvre de ce jeu. Le code est commenté afin de permettre aux lecteurs de comprendre le fonctionnement des différentes macros VBA.
Chaque jeu présenté est accompagné d'un classeur Excel téléchargeable depuis le site Internet de l'auteur : joseouin.fr. À la fin de cet ouvrage, une page explique la démarche à suivre pour obtenir l'ensemble des classeurs Excel correspondant à tous les jeux étudiés. Vous pouvez ainsi approfondir votre compréhension des stratégies et des mécaniques de jeu grâce à ces ressources pratiques.

2-1. Présentation du code VBA

Définition du « backtracking »

Dans le contexte de la résolution d'une grille de Sudoku (ou de tout autre problème de combinaison ou de recherche), le « backtracking » est une méthode algorithmique utilisée pour explorer toutes les possibilités d'une manière systématique et efficace.

Le processus de « backtracking » fonctionne de la manière suivante :

1. **Exploration récursive des choix** : À chaque étape, l'algorithme fait un choix parmi les options disponibles pour résoudre le problème. Par exemple, dans le Sudoku, à chaque étape de résolution d'une case vide, l'algorithme choisit un chiffre parmi les chiffres disponibles.

2. **Validation de la solution partielle** : Après avoir fait un choix, l'algorithme vérifie si la solution partielle jusqu'à présent est valide ou non. Dans le cas du Sudoku, cela signifie vérifier si le chiffre choisi est valide dans la ligne, la colonne et le carré 3x3 de la case.

3. **Récursion ou retour en arrière** : Si la solution partielle est valide, l'algorithme poursuivra l'exploration en explorant davantage les ramifications de la solution. Cependant, s'il n'y a pas de solution possible avec le choix actuel, l'algorithme "revient en arrière" (d'où le terme "backtracking"). Il annule le choix précédent et explore une autre branche de la solution.

4. **Exploration complète de toutes les possibilités** : L'algorithme continue ce processus d'exploration récursive et de retour en arrière jusqu'à ce qu'il ait exploré toutes les possibilités ou qu'il ait trouvé une solution valide.

Dans le contexte du Sudoku, le « backtracking » est utilisé pour essayer différentes combinaisons de chiffres dans les cases vides de la grille jusqu'à ce que la grille soit complètement remplie avec une solution valide. Si une erreur est détectée à un moment donné, l'algorithme revient en arrière pour essayer une autre combinaison, explorant ainsi toutes les possibilités jusqu'à ce qu'une solution soit trouvée.

Définition d'une fonction récursive

Une fonction récursive est une fonction qui s'appelle elle-même dans son corps. C'est un concept fondamental en programmation qui permet de résoudre des problèmes en les divisant en sous-problèmes plus petits et identiques, puis en combinant les résultats de ces sous-problèmes pour obtenir la solution globale. Dans le cas de la résolution d'un Sudoku, la fonction récursive est utilisée pour explorer toutes les possibilités de remplissage des cases vides de la grille.

La fonction récursive « `ResolverSudokuRecursive` » est utilisée pour résoudre une grille de Sudoku en utilisant une approche de backtracking.

Voici comment elle fonctionne :

1. La fonction prend en paramètre un tableau à deux dimensions représentant la grille de Sudoku à résoudre.

2. Elle recherche la première case vide dans la grille.

3. Si une case vide est trouvée, la fonction essaie différentes valeurs pour cette case, en commençant par 1 et en allant jusqu'à 9.

4. Pour chaque valeur testée, la fonction vérifie si elle est valide pour être placée dans la case, en vérifiant si elle est déjà présente dans la ligne, la colonne et le carré 3x3 de la case.

5. Si la valeur est valide, elle est placée dans la case, et la fonction est appelée récursivement pour résoudre le reste de la grille.

6. Si la fonction récursive renvoie vrai (ce qui signifie que la grille est entièrement remplie avec une solution valide), la fonction principale renvoie également vrai, indiquant que la résolution a réussi.

7. Si la fonction récursive renvoie faux (ce qui signifie qu'aucune solution n'a été trouvée avec la valeur actuelle), la fonction principale continue à essayer d'autres valeurs pour la case jusqu'à ce qu'une solution soit trouvée ou que toutes les possibilités aient été explorées.

Cette approche de résolution récursive permet d'explorer systématiquement toutes les possibilités de remplissage des cases vides de la grille de Sudoku jusqu'à ce qu'une solution valide soit trouvée.

Liste des procédures et des fonctions

Les procédures et les fonctions ci-après permettent de générer, effacer et résoudre des grilles de Sudoku dans Excel de manière interactive et efficace.

- **Sub CréerGrilleSudoku()** :

Cette procédure génère une grille de Sudoku valide dans la feuille de calcul "Sudoku". Elle initialise une grille vide, puis remplit les cellules avec des chiffres de sorte que chaque ligne, colonne et région 3x3 contienne tous les chiffres de 1 à 9 sans répétition. Après avoir créé la grille, elle efface aléatoirement 40 valeurs pour créer une grille de Sudoku partiellement complétée.

- **Sub CréerGrilleSudokuEtEffacer()** :

Cette procédure est une extension de la précédente. Elle affiche une boîte de dialogue d'attente pendant que la grille Sudoku est générée, puis cache la boîte de dialogue une fois la grille créée.

- **Sub EffacerAleatoire(grille As Range)** :

Cette procédure efface aléatoirement 40 valeurs de la grille de Sudoku fournie en paramètre. Elle est utilisée pour effacer certaines valeurs de la grille générée aléatoirement, créant ainsi une grille partiellement complétée pour le joueur.

- **Sub EffacerCasesAleatoirement()** :

Cette procédure efface aléatoirement 40 cases de la grille de Sudoku dans la feuille de calcul "Sudoku". Elle est une alternative à la procédure « `EffacerAleatoire` » et remplit le même objectif.

- **Sub ResolutionGrilleSudoku()** :

Cette procédure copie la grille de Sudoku de la plage principale vers une plage de solution dans la feuille de calcul « Sudoku ». Ensuite, elle résout la grille de Sudoku en appelant une fonction de résolution récursive.

- **Sub ResolverSudoku(ByRef solution As Range)** :

Cette procédure est appelée pour résoudre la grille de Sudoku. Elle copie d'abord la grille de solution dans un tableau à deux dimensions, puis appelle une fonction récursive pour résoudre le Sudoku.

- **Function ResolverSudokuRecursive(ByRef solutionArray As Variant) As Boolean** :

Cette fonction résout récursivement la grille de Sudoku en utilisant une approche de backtracking. Elle essaie différentes valeurs pour chaque case vide jusqu'à ce que la grille soit entièrement remplie et valide.

- **Function TrouverCelluleVide(ByRef solutionArray As Variant, ByRef rowIdx As Integer, ByRef colIdx As Integer) As Boolean** :

Cette fonction recherche la première cellule vide dans la grille de Sudoku. Elle retourne les indices de ligne et de colonne de la première cellule vide trouvée.

- **Function EstChiffreValide(ByRef solutionArray As Variant, ByVal rowIdx As Integer, ByVal colIdx As Integer, ByVal chiffre As Integer) As Boolean** :

Cette fonction vérifie si un chiffre est valide pour être placé dans une certaine case de la grille de Sudoku, en vérifiant s'il est déjà présent dans la ligne, la colonne et le carré 3x3 de cette case.

- **Function EstPresentDansLigne(ByRef solutionArray As Variant, ByVal rowIdx As Integer, ByVal chiffre As Integer) As Boolean** :

Cette fonction vérifie si un chiffre est déjà présent dans la ligne spécifiée de la grille de Sudoku.

- **Function EstPresentDansColonne(ByRef solutionArray As Variant, ByVal colIdx As Integer, ByVal chiffre As Integer) As Boolean** :

Cette fonction vérifie si un chiffre est déjà présent dans la colonne spécifiée de la grille de Sudoku.

- **Function EstPresentDansCarre(ByRef solutionArray As Variant, ByVal rowIdx As Integer, ByVal colIdx As Integer, ByVal chiffre As Integer) As Boolean** :

Cette fonction vérifie si un chiffre est déjà présent dans le carré 3x3 auquel appartient la case spécifiée de la grille de Sudoku.

2-2. Le code VBA du jeu

```
1   Sub CréerGrilleSudoku()
2       Dim grille As Range
3       Dim chiffres() As Integer
4       Dim i As Integer, j As Integer, k As Integer
5       Dim ligne As Integer, colonne As Integer
6       Dim chiffre As Integer, index As Integer
7       Dim count As Integer
8
9
10      ' Définit la plage de la grille Sudoku
11       Set grille = ThisWorkbook.Sheets("Sudoku").Range("B4:J12")
12
13      ' Efface la grille existante
14       grille.ClearContents
15
16      ' Initialisation du tableau de chiffres disponibles
17       ReDim chiffres(1 To 9)
18
19      ' Génère une grille Sudoku valide
20       For i = 1 To 9
21           For j = 1 To 9
22              ' Initialise les chiffres disponibles pour la case (i,j)
23               For k = 1 To 9
24                   chiffres(k) = k
25               Next k
26
27              ' Trouve la ligne et la colonne de la région 3x3
28               ligne = 3 * Int((i - 1) / 3) + 1
29               colonne = 3 * Int((j - 1) / 3) + 1
30
31              ' Élimine les chiffres déjà utilisés dans la ligne, la colonne et
la région 3x3
32               For k = 1 To 9
33                                If  Not  IsEmpty(grille.Cells(i,  k))  Then
chiffres(grille.Cells(i, k).Value) = 0
34                                If  Not  IsEmpty(grille.Cells(k,  j))  Then
chiffres(grille.Cells(k, j).Value) = 0
35               Next k
```

```
36              For k = ligne To ligne + 2
37                  For index = colonne To colonne + 2
38                          If Not IsEmpty(grille.Cells(k, index)) Then
chiffres(grille.Cells(k, index).Value) = 0
39                  Next index
40              Next k
41
42              ' Compte les chiffres disponibles
43              count = 0
44              For k = 1 To 9
45                  If chiffres(k) <> 0 Then count = count + 1
46              Next k
47
48              ' Vérifie s'il y a au moins une valeur disponible
49              If count = 0 Then
50                ' Retourne en arrière pour réinitialiser la grille
51                  grille.ClearContents
52                  i = 1
53                  j = 1
54                  Exit For
55              End If
56
57              ' Sélectionne un chiffre aléatoire parmi les chiffres disponibles
58              Do
59                  chiffre = Int((9 * Rnd) + 1)
60              Loop While chiffres(chiffre) = 0
61              grille.Cells(i, j).Value = chiffre
62          Next j
63      Next i
64
65      ' Efface 40 valeurs de manière aléatoire
66      EffacerAleatoire grille
67
68  End Sub
```

```
1   Sub CréerGrilleSudokuEtEffacer()
2       ' Affiche la UserForm d'attente
3       UserFormAttente.Show vbModeless
4       UserFormAttente.Repaint
5       ' Appelle la procédure pour créer la grille Sudoku
6       CréerGrilleSudoku
7
8       ' Cache la UserForm d'attente
9       UserFormAttente.Hide
10  End Sub
```

Neuvième jeu

Calcul mental

Le jeu de calcul mental sur Excel offre une expérience divertissante et éducative, accessible à tous, des plus jeunes aux adultes. En combinant plaisir et exercice cérébral, ce jeu stimulant permet de renforcer les compétences en mathématiques tout en offrant une expérience ludique.

- **Un Jeu pour Tous :**

Que vous soyez un enfant en phase d'apprentissage des mathématiques de base ou un adulte cherchant à entretenir ses capacités mentales, ce jeu est conçu pour vous. Il propose différents niveaux de difficulté, adaptés à chaque niveau de compétence, permettant ainsi à chacun de défier et d'améliorer ses compétences en calcul mental.

- **Rappels Historiques :**

Les jeux de calcul mental ont une longue histoire, remontant à l'époque des anciennes civilisations où les défis mathématiques étaient utilisés comme moyen d'entraînement intellectuel. Au fil du temps, ces jeux ont évolué pour devenir des outils d'apprentissage populaires, aidant les individus à développer leur agilité mentale et leur précision dans les calculs.

Le jeu de calcul mental sur Excel est bien plus qu'un simple divertissement : il représente une opportunité d'apprentissage interactive et enrichissante. Que ce soit pour améliorer les compétences mathématiques, exercer son cerveau ou simplement s'amuser, ce jeu offre une expérience stimulante pour tous les passionnés de mathématiques, petits et grands.

Le jeu : Calcul Mental

1- Présentation et règles du jeu

1-1. Présentation

Dans ce jeu de calcul mental, votre objectif est de résoudre une série de calculs mathématiques le plus rapidement possible.

1-2. Règles et déroulement du jeu avec Excel

Génération des opérations : À chaque tour, une série d'opérations mathématiques apparaîtra dans la colonne « B ». Ces opérations contiendront des nombres entiers aléatoires compris entre 1 et une valeur que vous pouvez spécifier dans la cellule « H2 ».

Entiers compris entre 1 et :	15

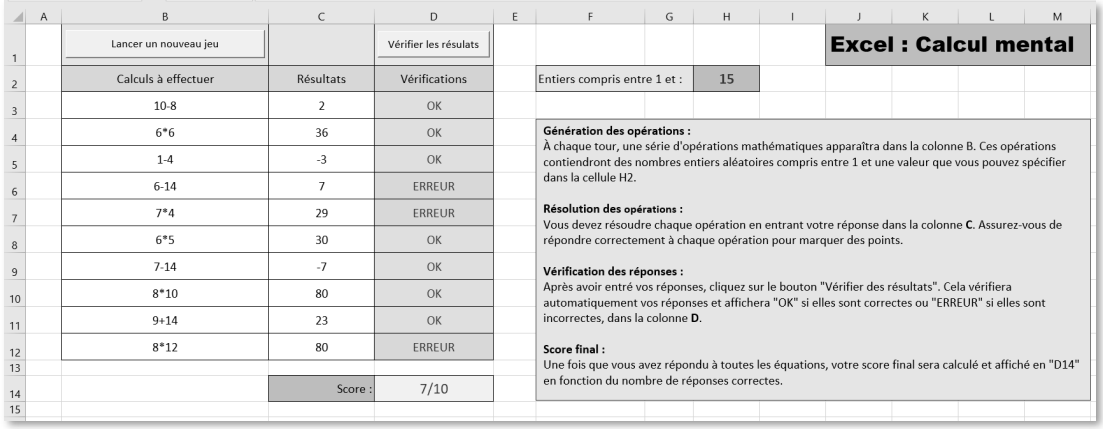
Résolution des opérations : Vous devez résoudre chaque opération en entrant votre réponse dans la colonne « C ». Assurez-vous de répondre correctement à chaque opération pour marquer des points.

Vérification des réponses : Après avoir entré vos réponses, cliquez sur le bouton « Vérifier les résultats » : la macro correspondante vérifiera automatiquement vos réponses et affichera "OK" si elles sont correctes ou "ERREUR" si elles sont incorrectes, dans la colonne « D ».

Score final : Une fois que vous avez répondu à toutes les opérations, votre score final sera calculé en fonction du nombre de réponses correctes en cellule « D14 ».

Défi personnel : Essayez de battre votre meilleur score à chaque partie et d'améliorer votre rapidité de calcul. Prêt à relever le défi du calcul mental ? Lancez-vous et montrez vos compétences mathématiques !

Capture d'écran de la feuille Excel : opérations avec de petits nombres pour les tous petits



2- Code VBA pour Excel

Le code VBA ci-après présente les différentes procédures (Sub) et fonctions (Function) nécessaires à la mise en œuvre de ce jeu. Le code est commenté afin de permettre aux lecteurs de comprendre le fonctionnement des différentes macros VBA.
Chaque jeu présenté est accompagné d'un classeur Excel téléchargeable depuis le site Internet de l'auteur : joseouin.fr. À la fin de cet ouvrage, une page explique la démarche à suivre pour obtenir l'ensemble des classeurs Excel correspondant à tous les jeux étudiés. Vous pouvez ainsi approfondir votre compréhension des stratégies et des mécaniques de jeu grâce à ces ressources pratiques.

2-1. Présentation du code VBA

Ces procédures et fonctions travaillent ensemble pour créer une expérience de jeu fluide et interactive, encourageant les utilisateurs à résoudre des équations mathématiques tout en s'amusant.

Liste des procédures et des fonctions

- **Sub JeuCalculMental()**

Cette procédure initialise le jeu de calcul mental en effaçant la plage des questions précédentes, génère de nouvelles opérations mathématiques à résoudre et les affiche dans la plage dédiée sur la feuille de calcul.

- **Sub VerificationResultats()**

Cette procédure vérifie les réponses fournies par l'utilisateur aux opérations mathématiques et affiche « OK » si la réponse est correcte et « ERREUR » sinon. Elle appelle également la fonction `AfficherScoreFinal` pour afficher le score final du joueur.

- **Function GenererEquation() As String**

Cette fonction génère une équation mathématique aléatoire en choisissant deux nombres aléatoires entre 1 et une valeur définie par l'utilisateur, puis en sélectionnant aléatoirement l'un des trois opérateurs : addition (+), soustraction (-) ou multiplication (*).

- **Sub AfficherScoreFinal()**

Cette procédure calcule le score final du joueur en comptant le nombre de bonnes réponses et en le divisant par le nombre total de questions. Elle affiche ensuite ce score final dans une cellule spécifiée sur la feuille de calcul.

- **Sub EffacerPlage()**

Cette procédure efface le contenu de la plage des questions et des réponses précédentes, ainsi que le score final affiché précédemment, afin de préparer l'espace pour un nouveau jeu.

2-2. Le code VBA du jeu

```
Sub JeuCalculMental()
    Dim nbQuestions As Integer
    Dim score As Integer
    Dim questionRange As Range
    Dim i As Integer

   'Effacement de la plage
    EffacerPlage

     ' Références à la plage des questions
     Set questionRange = ThisWorkbook.Sheets("Feuil1").Range("B3:B12")

     ' Définir le nombre de questions
     nbQuestions = questionRange.Rows.Count

     ' Initialiser le score
     score = 0
     ' Poser les questions
     For i = 1 To nbQuestions
        ' Afficher la question
        questionRange.Cells(i, 1).Formula = "=GenererEquation()"
     Next i
End Sub
```

```
Sub VerificationResultats()
    Dim questionRange As Range
    Dim reponseRange As Range
    Dim resultatRange As Range
    Dim i As Integer
    Dim bonneReponse As String

   ' Références aux plages de questions, réponses et résultats
    Set questionRange = ThisWorkbook.Sheets("Feuil1").Range("B3:B12")
     Set reponseRange = ThisWorkbook.Sheets("Feuil1").Range("C3:C12")
     Set resultatRange = ThisWorkbook.Sheets("Feuil1").Range("D3:D12")

     ' Vérifier les réponses et afficher les résultats
     For i = 1 To questionRange.Rows.Count
         bonneReponse = Evaluate(questionRange.Cells(i, 1).Value)
         If reponseRange.Cells(i, 1).Value = bonneReponse Then
             resultatRange.Cells(i, 1).Value = "OK"
         Else
             resultatRange.Cells(i, 1).Value = "ERREUR"
         End If
     Next i

    'Affichage du score final
     AfficherScoreFinal
End Sub
```

Cet ouvrage a été achevé en juin 2024

Dépôt légal : juin 2024

Déposé auprès de la BnF (Bibliothèque Nationale de France)