

Examen d'Informatique

Mercredi 11 mai 2016 - Durée : 2h

Documents, calculatrices et téléphones portables interdits

Remarques :

- vous trouverez à la fin du sujet quelques rappels VBA extraits de votre polycopié de cours qui peuvent vous être utiles pour répondre à certaines questions ;
- l'évaluation de vos réponses tient compte de la lisibilité et de l'efficacité de vos programmes ;
- à chaque question, vous pouvez supposer que les fonctions et procédures des questions précédentes sont définies correctement, vous pouvez donc les utiliser.

Exercice 1 : Le jeu de la vie

Le jeu de la vie est une modélisation de la sélection naturelle des espèces selon leur adaptabilité à l'environnement. On considère un monde dans lequel plusieurs individus se déplacent selon leur code génétique à la recherche de nourriture. Sur le long terme, la simulation permet de déterminer quelle population s'adapte le mieux à un environnement. Le *monde* est considéré comme une grille à 2 dimensions et se comporte comme un tore : si un animal quitte le monde par le bas, il apparaît par le haut. Il en est de même pour les côtés. Chaque cellule comporte donc 8 voisines, y compris celles situées "au bord" du monde (puisque c'est un tore). Une petite partie du monde, appelée la *Beauce*, est considérée comme *fertile*, avec plus de nourriture. On suppose le temps comme étant *tour par tour*. Les *animaux* se déplacent de cellule en cellule à chaque pas de temps. Un déplacement coûte 1 unité d'énergie à l'animal. Si un animal n'a plus d'énergie, il disparaît du monde. Si un animal arrive sur une cellule contenant un ou plusieurs morceau(x) de nourriture, il consomme tous les morceaux et gagne de l'énergie. Il ne peut y avoir qu'un seul animal par cellule, aussi si un animal doit se rendre sur une cellule qui en contient déjà un autre, alors il ne se déplace pas. Chaque animal possède un *chromosome* qui détermine la direction dans laquelle il se déplace. Un chromosome est constitué de 4 gènes *g, d, h* et *b*, pour *gauche, droite, haut* et *bas*, qui sont des valeurs binaires (0 ou 1) codant la direction de déplacement. Si le gène vaut 1 alors l'animal se déplace dans la direction codée par le gène. Ainsi, un animal dont les gènes *g* et *b* valent 1 et les gènes *d* et *h* valent 0 se déplacera dans la cellule située en bas à gauche de celle dans laquelle il se trouve. On peut noter que si les gènes *g* et *d* ont la même valeur, alors l'animal ne se déplacera pas horizontalement. On peut faire la même observation avec les gènes *h* et *b* et le déplacement vertical. Au cours de la simulation, un animal peut subir une *mutation* qui modifie son chromosome et, par conséquent, sa façon de se déplacer.

On s'intéresse dans cet exercice à l'écriture d'un programme VBA permettant une simulation du jeu de la vie. Le jeu de la vie est représenté dans une feuille Excel de la manière suivante :

- Le **monde** est la plage de cellules allant de la cellule L1C1 à la cellule L10C10.
- La **Beauce** est la plage de cellules allant de la cellule L4C4 à la cellule L6C6 ; l'intérieur de ses cellules est coloré en gris pour que l'utilisateur puisse visualiser cette zone.
- Les **animaux** créés pour la simulation sont listés à partir de la ligne 15 suivant le format : identifiant de l'animal (colonne 1), quantité d'énergie (colonne 2), et les 4 gènes de son chromosome dans l'ordre *g, d, h* et *b* (colonnes 3 à 6). Un animal est représenté dans le monde par son identifiant placé dans la cellule dans laquelle il se trouve.
- La présence de **nourriture** dans une cellule du monde est signalée par un **nombre en gras** qui représente la quantité de nourriture présente.

La figure 1 représente la feuille Excel obtenue après 2 itérations d’une simulation effectuée à partir de 5 animaux créés aléatoirement.

	1	2	3	4	5	6	7	8	9	10	11
1	1										
2								1			
3											
4				1		4				1	
5											
6					1					2	
7											
8		5									
9											
10											
11											
12											
13			0	0	1	0					
14											
15	1	17	1	0	1	0					
16	2	25	0	1	0	1					
17	3	0	0	1	0	0					
18	4	6	1	0	0	1					
19	5	27	0	0	0	1					
20											
21											

FIGURE 1 – Représentation du jeu de la vie sous Excel

On peut voir par exemple que l’animal 1 situé en L1C1 dispose encore de 17 unités d’énergie et que l’animal 2 situé en L6C10 dispose encore de 25 unités d’énergie. L’animal 3 en revanche n’a plus d’énergie, il n’apparaît donc plus dans le monde. L’animal 4 est le seul animal situé dans la Beauce à ce moment de la simulation. On peut aussi remarquer qu’au cours des 2 itérations, un morceau de nourriture a été placé dans les cellules L4C4, L6C5, L2C8 et L4C10, puisque le chiffre 1 de ces cellules est en gras. Le contenu de la ligne 13 (colonne 3 à 6) sera expliqué à la question 1.7.

La macro principale du programme est la macro `jeuVie` donnée ci-dessous :

```

Sub jeuVie()
Dim nIt As Integer, i As Integer
    'Création du monde
    beauce
    creeAnimaux

    'Simulation de l'évolution des espèces
    nIt = Application.InputBox("Nombre d'itérations ?")
    For i = 1 To nIt
        placeNourriture
        dplctAnimaux
    Next i

    'Analyse
    MsgBox "L'énergie moyenne des animaux dans la Beauce est " & moyB()
    triAnimaux
End Sub

```

La macro `jeuVie` se décompose en 3 parties. Une partie gérant la création du monde (macros `beauce` et `creerAnimaux`). Une partie gérant le déroulement de la simulation : à chaque itération, un morceau de nourriture est placé dans le monde et un dans la Beauce (procédure `placeNourriture`), puis tous les animaux encore vivants sont déplacés (macro `dplctAnimaux`). Enfin, la dernière partie analyse le résultat de la simulation en déterminant l'énergie moyenne des animaux dans la Beauce (fonction `moyB`) et en triant les animaux dans l'ordre décroissant de leur énergie (macro `triAnimaux`). Le but de l'exercice est d'écrire les procédures et fonctions utilisées par la macro `jeuVie`.

Dans ce jeu de la vie, il sera demandé à plusieurs reprises de générer un nombre aléatoirement (au hasard) entre deux valeurs données. Vous pourrez utiliser pour ceci la fonction `alea` donnée ci-dessous qui retourne un nombre compris entre `premVal` et `dernVal` (incluses).

```
Function alea(ByVal premVal As Long, ByVal dernVal As Long) As Integer
    alea = Int(Rnd() * (dernVal - premVal + 1)) + premVal
End Function
```

Ainsi `alea(2,5)` par exemple peut retourner 2, ou 3, ou 4, ou 5.

Partie 1. Création du monde [7 pts]

On s'intéresse dans cette première partie à la création du monde et des animaux.

Q 1.1 Écrire en VBA la macro `Beauce` qui colore en gris clair (RGB(220, 220, 220)) l'intérieur des cellules de la Beauce.

```
Sub beauce()
Dim i As Long, j As Long
For i = 4 To 6
    For j = 4 To 6
        Cells(i, j).Interior.Color = RGB(220, 220, 220)
    Next j
Next i
End Sub

OU

Sub beauce()
    Range(Cells(4, 4), Cells(6, 6)).Interior.Color = RGB(220, 220, 220)
End Sub
```

Sur 2pts

- boucles /0.5
- `Cells(i, j).Interior.Color = RGB(220, 220, 220) /1`
- **syntaxe (déclaration des variables, procédure,...) /0.5**

OU

- utilisation correcte de `Range` avec d'éventuelles déclarations de variables /1
- `Cells(i, j).Interior.Color = RGB(220, 220, 220) /1`

Q 1.2 Un chromosome étant constitué de 4 gènes binaires, il est considéré comme un nombre représenté en base 2 sur 4 bits, chaque bit représentant un gène.

Écrire en VBA la procédure `creerGene` qui prend en paramètre un numéro de ligne `li` et un entier positif `n` en base 10, et qui place la représentation en base 2 de `n` à la ligne `li` des colonnes 3 à 6 (colonnes des gènes des animaux).

Dans l'exemple de la figure 1, l'exécution de l'instruction `creerGene 16, 5` génère le chromosome de l'animal listé à la ligne 16 (puisque 5 vaut 0101 en base 2). Si la représentation en base 2 du nombre `n` passé en paramètre nécessite plus de 4 bits, la procédure `creerGene` affiche le message "Erreur".

```
Sub creeGene(ByVal li As Long, ByVal n As Long)
    Dim nb As Long, j As Long
    nb = n
    For j = 6 To 3 Step -1
        Cells(li, j).Value = nb Mod 2
        nb = nb \ 2
    Next j
    If nb > 0 Then MsgBox "Erreur"
End Sub
```

Sur 2 pts

- conversion en binaire (boucle, Mod, \) /1
- dépassement de capacité /0.5
- syntaxe (paramètres, variables, etc.) /0.5

Q 1.3 On considère maintenant la génération aléatoire des animaux. Écrire en VBA la macro `creerAnimaux` qui génère autant d'animaux que voulu par l'utilisateur. Pour cela, la macro génère un premier animal, d'identifiant 1, et le liste à la ligne 15. Puis la macro demande à l'utilisateur s'il souhaite créer un nouvel animal. Si la réponse de l'utilisateur est positive, l'animal d'identifiant 2 est ensuite initialisé et listé à la ligne 16, et ainsi de suite jusqu'à ce que l'utilisateur ne souhaite plus créer de nouvel animal. L'interaction avec l'utilisateur est gérée à l'aide de `MsgBox` et de ses boutons "Oui" et "Non". La génération d'un animal consiste à initialiser aléatoirement son énergie (par un nombre tiré aléatoirement entre 1 et 30), son chromosome et sa position dans le monde. Comme il ne peut y avoir qu'un seul animal par cellule, la position aléatoire d'un nouvel animal doit être à nouveau tirée aléatoirement si un autre animal est déjà présent dans la cellule.

```
Sub creeAnimaux()  
  Dim nb As Integer, rep As Integer, i As Long, j As Long  
  nb = 0  
  Do  
    nb = nb + 1  
    Cells(14 + nb, 1).Value = nb  
    Cells(14 + nb, 2).Value = alea(1, 30)  
    creeGene 14 + nb, alea(0, 15)  
    Do  
      i = alea(1, 10): j = alea(1, 10)  
    Loop Until Cells(i, j).Value = ""  
    Cells(i, j).Value = nb  
    rep = MsgBox("Nouvel animal ?", 4)  
  Loop Until rep = 7  
End Sub
```

Sur 3 pts

- MsgBox et boucle principale /1
- Calcul du i et j /1 (/0.5 appel de alea, /0.5 boucle)
- reste (autres instructions) /1

Deuxième partie. Déroulement de la simulation [10 pts]

On s'intéresse dans cette partie au déroulement de la simulation, c'est à dire au placement de la nourriture et au déplacement des animaux à chaque itération. Le placement de la nourriture est effectué par la macro `placeNourriture` qui ajoute un morceau de nourriture dans une cellule du monde ne contenant pas d'animal choisie aléatoirement et un morceau de nourriture dans une cellule de la Beauce ne contenant pas d'animal choisie aléatoirement. On suppose que cette macro est déjà définie et on ne demande pas de l'écrire. On s'intéresse ici à l'écriture de la partie du programme qui gère le déplacement des animaux.

Q 1.4 Écrire en VBA la procédure `cellAnimal` qui détermine la cellule dans laquelle se trouve l'animal `nb` passé en paramètre. Pour cela, la procédure donne le numéro de la ligne et le numéro de la colonne de la cellule dans laquelle se trouve l'animal. Précisons qu'il n'est **pas** demandé ici d'utiliser de boîtes de dialogues.

```

Sub cellAnimal(ByVal nb As Integer, ByRef li As Long, ByRef co As Long)
  Dim i As Long, j As Long
  For i = 1 To 10
    For j = 1 To 10
      If Cells(i, j).Value = nb And Cells(i, j).Font.Bold = False Then
        li = i: co = j
        Exit Sub
      End If
    Next j
  Next i
End Sub

```

Sur 2 pts

- Byref (ou variable globale) /0.5
- sortie de boucle /0.5
- parcours de la plage et test corrects /1

Pour gérer le déplacement des animaux, il est indispensable de connaître la cellule voisine dans laquelle devrait se déplacer un animal en fonction de son chromosome et de sa position courante. La détermination de la ligne et de la colonne suivantes à partir de la ligne et de la colonne courantes est gérée par les fonctions `lSvte` et `cSvte` données ci-dessous :

```

Function lSvte(ByVal liA As Long, ByVal nA As Integer) As Long
  lSvte = liA - Cells(nA + 14, 5).Value + Cells(nA + 14, 6).Value
  Select Case lSvte
    Case 0: lSvte = 10
    Case 11: lSvte = 1
  End Select
End Function
Function cSvte(ByVal colA As Long, ByVal nA As Integer) As Long
  cSvte = colA - Cells(nA + 14, 3).Value + Cells(nA + 14, 4).Value
  Select Case cSvte
    Case 0: cSvte = 10
    Case 11: cSvte = 1
  End Select
End Function

```

Dans ces deux fonctions, `liA` (respectivement `colA`) représente la ligne (respectivement colonne) initiale de l'animal dans le monde et `nA` l'identifiant de l'animal.

Q 1.5 Dans l'exemple de la figure 1, la prochaine cellule de l'animal 2 (actuellement en L6C10) est déterminée par l'exécution de `lSvte(6,2)` et `cSvte(10,2)`. Quelle est cette prochaine cellule ?

L7C1 (0.5pt)

Q 1.6 Écrire en VBA la procédure `dplct` qui prend un animal `nb` en paramètre et le déplace dans la cellule voisine appropriée. Si un animal y est déjà présent, l'animal `nb` ne se déplace pas. Si la cellule voisine contient de la nourriture, alors l'animal `nb` mange la nourriture, ce qui augmente son énergie de la quantité de nourriture présente, puis se déplace dans la cellule voisine (qui ne contient alors plus de nourriture). Le déplacement coûte une unité d'énergie à l'animal `nb`. Si, à l'issue de ce déplacement, l'animal `nb` n'a plus d'énergie, il disparaît du monde (mais pas de la liste des animaux) et le message "L'animal `nb` n'a plus d'énergie" est affiché.

```
Sub dplct(ByVal nb As Integer)
    Dim i As Long, j As Long, i_next As Long, j_next As Long

    cellAnimal nb, i, j
    i_next = lSvte(i, nb)
    j_next = cSvte(j, nb)

    If Cells(i_next, j_next).Font.Bold Then
        'nourriture
        Cells(14 + nb, 2).Value = Cells(14 + nb, 2).Value + Cells(i_next, j_next).Value - 1
        Cells(i_next, j_next).Font.Bold = False
        Cells(i_next, j_next).Value = nb
        Cells(i, j).Value = ""
    ElseIf Cells(i_next, j_next).Value = "" Then
        Cells(14 + nb, 2).Value = Cells(14 + nb, 2).Value - 1
        Cells(i, j).Value = ""
        If Cells(14 + nb, 2).Value <> 0 Then
            Cells(i_next, j_next).Value = nb
        Else
            MsgBox "l'animal" & nb & "n'a plus d'énergie"
        End If
    End If
End Sub
```

Sur 3 pts

- appel des fonctions et procédures /1 (/0.5 pour `cellAnimal`, /0.5 pour `lSvte` et `cSvte`)
- premier if /1
- deuxième if /1

Q 1.7 On s'intéresse dans cette question à la mutation du chromosome d'un animal. La mutation est simulée par une addition binaire du chromosome avec un nombre généré aléatoirement.

Écrire en VBA la procédure `mutation` qui prend en paramètre un identifiant d'animal `nb` et qui fait subir une mutation aux gènes de `nb`. Pour cela, la procédure `mutation` commence par générer un nombre aléatoire en base 10 représentable sur 4 bits, écrit la représentation binaire de ce nombre à la ligne 13 de la feuille Excel (colonnes 3 à 6), puis ajoute ce nombre au chromosome de l'animal `nb`. On demande ici d'effectuer une addition binaire des deux nombres binaires sur 4 bits (on ignore les éventuels dépassements de capacité après l'addition).

```
Sub mutation(ByVal nb As Integer)
  Dim r As Integer, i As Long, liA As Long
  creeGene 13, alea(0, 15)

  liA = 15
  Do Until Cells(liA, 1).Value = nb
    liA = liA + 1
  Loop

  'Autre version acceptable
  'liA = nb + 14

  r = 0
  For i = 6 To 3 Step -1
    Cells(liA, i).Value = Cells(liA, i).Value + Cells(13, i).Value + r
    If Cells(liA, i).Value >= 2 Then
      Cells(liA, i).Value = Cells(liA, i).Value - 2
      r = 1
    Else
      r = 0
    End If
  Next i
  Cells(liA, 8).Value = Cells(liA, 8).Value + 1
End Sub
```

Sur 2.5 pts

- calcul de liA /0.5
- écriture à la ligne 13 /0.5
- addition correcte /1.5

Q 1.8 Écrire en VBA la macro `dplctAnimaux` qui déplace à tour de rôle tous les animaux ayant encore de l'énergie (on n'impose pas l'ordre dans lequel les animaux sont déplacés). À l'issue de ces déplacements, la macro `dplctAnimaux` fait subir une mutation à l'un des animaux tiré aléatoirement dans la liste (qu'il ait encore de l'énergie ou non).


```
Sub dplctAnimaux()  
Dim nb As Long  
nb = 1  
Do While Cells(nb + 14, 1).Value <> ""  
    If Cells(nb + 14, 2).Value > 0 Then  
        dplct nb  
    End If  
    nb = nb + 1  
Loop  
nb = alea(1, nb - 1)  
mutation nb  
End Sub
```

Sur 2 pts

- boucle /1
- décompte des animaux /0.5
- mutation et alea /0.5

Troisième partie. Analyse [5 pts]

Q 1.9 Écrire en VBA la fonction moyB qui retourne l'énergie moyenne des animaux présents dans la Beauce. Si aucun animal n'est dans la Beauce, la fonction moyB retourne 0.

On accepte tout aussi bien un parcours de la liste des animaux pour déterminer ceux qui sont dans la beauce (version moyB) qu'un parcours de la Beauce pour rechercher l'énergie des animaux présents dans la Beauce (version moyB_bis).

```
Function moyB() As Single
Dim nbA As Integer, nrj As Integer, i As Long, j As Long, k As Long
nbA = 0: nrj = 0
k = 15
Do Until Cells(k, 1).Value = ""
    cellAnimal Cells(k, 1).Value, i, j
    'autre version acceptable
    'cellAnimal k-14, i, j

    If i > 3 And i < 7 And j > 3 And j < 7 Then
        nrj = nrj + Cells(k, 2).Value
        nbA = nbA + 1
    End If
    k = k + 1
Loop
If nbA > 0 Then
    moyB = nrj / nbA
Else
    moyB = 0
End If
End Function
```

```
Function moyB_bis() As Single
Dim nbA As Integer, nrj As Integer, i As Long, j As Long, k As Long
nbA = 0: nrj = 0
For i = 4 To 6
    For j = 4 To 6
        If Cells(i, j).Value > 0 And Not Cells(i, j).Font.Bold Then
            nbA = nbA + 1
            k = 15
            Do Until Cells(k, 1).Value = Cells(i, j).Value
                k = k + 1
            Loop
            nrj = nrj + Cells(k, 2).Value
        End If
    Next j
Next i
moyB_bis = 0
If nbA > 0 Then moyB_bis = nrj / nbA
End Function
```

Sur 2 pts

- recherche de l'énergie d'un animal de la Beauce /1
- calcul de la moyenne /1
- -0.5 pt si l'étudiant retourne un type entier ou fait une division par 0

Q 1.10 Écrire en VBA la macro `triAnimaux` qui trie les animaux de la liste dans l'ordre décroissant de leur énergie. Dans cette question, la notation ne tient pas compte du choix de l'algorithme de tri. Par exemple, l'application de la macro `triAnimaux` sur l'exemple de la figure 1 donnerait le résultat représenté dans la figure 2.

	1	2	3	4	5	6	7	8	9	10	11
1	1										
2								1			
3											
4				1		4				1	
5											
6					1					2	
7											
8		5									
9											
10											
11											
12											
13			0	0	1	0					
14											
15	5	27	0	0	0	1					
16	2	25	0	1	0	1					
17	1	17	1	0	1	0					
18	4	6	1	0	0	1					
19	3	0	0	1	0	0					
20											
21											

FIGURE 2 – Représentation du jeu de la vie après le tri des animaux

```
Sub triAnimaux()  
Dim i As Long, j As Long, lmax As Long, tmp As Integer, k As Long  
i = 15  
Do  
j = i + 1  
lmax = i  
Do While Cells(j, 1).Value <> ""  
If Cells(j, 2).Value > Cells(lmax, 2).Value Then  
    lmax = j  
End If  
j = j + 1  
Loop  
For k = 1 To 6  
tmp = Cells(lmax, k).Value  
Cells(lmax, k).Value = Cells(i, k).Value  
Cells(i, k).Value = tmp  
Next k  
i = i + 1  
Loop While Cells(i, 1).Value <> ""  
End Sub
```

Sur 3 pts

- tri /2
- échanges /1
- pas de pénalité si le tri est écrit pour une liste de 5 animaux

Rappels VBA

Ces éléments de cours sont donnés à titre indicatif

La méthode MsgBox

Function MsgBox (Prompt **As Variant** , -
 Optional Buttons **As Long** , -
 Optional Title **As String**) **As Integer**
 ' Affiche, sous forme de texte, la valeur Prompt à l'intérieur d'une
 ' boîte de dialogue ayant comme titre Title, la valeur de Buttons
 ' précisant les boutons qui sont affichés dans la boîte de dialogue.

Les tableaux suivants présentent les valeurs du paramètre **Buttons** et les valeurs de retour possibles :

Valeur	Description	Valeur	Description
0	OK	1	OK
1	OK et Annuler	2	Annuler
2	Abandonner, Recommencer et Ignorer	3	Abandonner
3	Oui, Non et Annuler	4	Recommencer
4	Oui et Non	5	Ignorer
5	Réessayer et Annuler	6	Oui
		7	Non

Valeurs du paramètre Buttons de MsgBox Valeurs renvoyées par MsgBox

Quelques propriétés de certaines classes VBA :

Classe Range :

Property Cells (i **As Long** , j **As Long**) **As Range**
 ' Cellule de la ligne i et de la colonne j relativement à l'objet Range

Property Value **As Variant**
 ' Valeur de la cellule

Property Interior **As Interior**
 ' Objet qui représente le fond des cellules de la plage

Property Font **As Font**
 ' Style des caractères affichés dans les cellules de la plage

Classe Interior :

Property Color **As Long**
 ' Couleur (codée en RGB) du fond.

Classe Font :

Property Bold **As Boolean**
 ' Gras des caractères.
