Examen d'informatique

Jeudi 30 janvier 2014

Durée : 1h30

Documents, calculettes et téléphones portables interdits
Le barème est donné à titre indicatif, il est susceptible d'être modifié

Remarques :
- la lisibilité et l'efficacité (optimisation du temps d'exécution par exemple) de vos programmes seront pris en compte pour l'évaluation de vos réponses ;
- les deux dernières pages de ce sujet sont une annexe dans laquelle figure des extraits de votre polycopié de cours qui peuvent vous être utiles pour répondre à certaines questions ;
- à chaque question, vous pouvez supposer que les fonctions et procédures des questions précédentes sont définies correctement, vous pouvez donc les utiliser.

Exercice 1 : Projets d'investissement (14 pts)

Dans cet exercice, on considère le problème de la sélection de projets dans lesquels une société voudrait investir. Pour cela, 199 projets ont été minutieusement étudiés par des experts. Afin de pouvoir les comparer, les experts ont synthétisé leurs évaluations en notant les projets selon trois critères : la *pertinence* des objectifs du projet, la *qualité* du projet et sa *durabilité*. Chaque projet est évalué sur chaque critère par une note allant de 0 (plus mauvaise note possible) à 50 (meilleure note posssible). Les experts ont reporté les évaluations des 199 projets dans une feuille Excel selon le format suivant : la première colonne contient le nom des projets, les deuxième, troisième et quatrième colonnes contiennent les notes obtenues par les projets selon le critère pertinence (colonne 2), qualité (colonne 3) et durabilité (colonne 4). Enfin, à la colonne 5 figure le coût d'investissement du projet. La figure 1 est un exemple du fichier Excel dans lequel sont regroupées les évaluations des projets.

	1	2	3	4	5	6
1	**Nom Projet**	**Pertinence**	**Qualité**	**Durabilité**	**Coût**	
2	AAA	35	42	18	25 000 €	
3	ABC	25	33	45	17 000 €	
4	DVD	32	40	15	30 000 €	
5	FAQ	46	19	31	8 000 €	

FIGURE 1 – Evaluation des projets effectuée par les experts

On remarque que la première ligne du fichier contient les en-têtes des colonnes, la liste des projets ne commence donc qu'à partir de la deuxième ligne. **Dans cet exercice, on travaille donc sur une feuille Excel dans laquelle 199 projets sont listés de la ligne 2 à la ligne 200 selon le format expliqué ci-dessus.**

Partie 1. Analyse des données (5.5 pts)

Dans un premier temps, la société souhaite analyser les données fournies par les experts.

Q 1.1 (1 pt) Ecrivez en VBA la fonction `controleProjet` qui prend en paramètre un numéro de ligne `li`, et retourne vrai si les notes du projet de la ligne `li` sont bien comprises entre 0 et 50 (inclus) sur chaque critère, et faux sinon. On suppose ici que le numéro de ligne est compris entre 2 et 200.

Q 1.2 (1.5 pt) Ecrivez en VBA la macro `controle` qui vérifie que **tous** les projets sont bien notés entre 0 et 50 sur chaque critère. Pour cela la macro `controle` met en gras les noms des projets ayant au moins une note non comprise entre 0 et 50, et affiche à l'aide d'une boîte de dialogue le message « Anomalie ! » s'il y a au moins un projet mis en gras. La macro n'affiche rien si toutes les notes de tous les projets sont bien comprises entre 0 et 50.

Q 1.3 (1 pt) Ecrivez en VBA la macro `moyenne` qui remplit la colonne 6 en calculant la moyenne des trois notes obtenues (une note par critère) pour chaque projet. Par exemple, l'exécution de la macro `moyenne` sur l'exemple de la figure 1 donnerait la feuille Excel suivante :

	1	2	3	4	5	6
1	**Nom Projet**	**Pertinence**	**Qualité**	**Durabilité**	**Coût**	
2	AAA	35	42	18	25 000 €	31,6666667
3	ABC	25	33	45	17 000 €	34,3333333
4	DVD	32	40	15	30 000 €	29
5	FAQ	46	19	31	8 000 €	32

FIGURE 2 – Exécution de `moyenne` sur l'exemple de la figure 1

On considère dorénavant que la macro `moyenne` a été exécutée. La colonne 6 contient donc la moyenne des notes des projets.

Q 1.4 (2 pts) Ecrivez en VBA la macro `recherche` qui demande à un utilisateur le nom d'un projet, puis affiche la note moyenne du projet en question s'il est dans la liste des projets de la feuille Excel ou le message « Projet X absent » sinon (où X est le nom du projet saisi par l'utilisateur).

Partie 2. Sélection des projets (8.5 pts)

On souhaite maintenant filtrer la liste des projets en supprimant ceux qui sont *Pareto-dominés*. Un projet X est considéré comme Pareto-dominé s'il existe un autre projet Y tel que la note de Y est strictement meilleure que celle de X sur tous les critères. Dans l'exemple de la figure 1, le projet DVD est Pareto-dominé par le projet AAA car il a obtenu une plus mauvaise note que AAA sur les trois critères (pertinence, qualité et durabilité). En revanche, le projet DVD n'est pas Pareto-dominé par le projet FAQ par exemple car DVD a obtenu une meilleure note que FAQ sur le critère qualité.

Q 1.5 (2.5 pts) Ecrivez en VBA la macro `pareto` qui indique les projets Pareto-dominés en colorant en rouge (`RGB (230,0,0)`) l'intérieur des cellules des lignes des projets Pareto-dominés, de la colonne 1 à la colonne 6. Par exemple, l'appel de la macro `pareto` sur l'exemple de la figure 1 colorerait en rouge l'intérieur des cellules de la ligne 4 (de la colonne 1 à la colonne 6) puisque le projet DVD est Pareto-dominé par le projet AAA. L'intérieur des autres cellules ne serait pas modifié car les projets AAA, ABC et FAQ ne sont Pareto-dominés par aucun autre projet.

On considère maintenant que les projets non Pareto-dominés ont été listés dans la Feuille 2 du fichier Excel en reportant leur nom en colonne 1, leur coût en colonne 2 et leur moyenne en colonne 3. Pour l'exemple de la figure 1, la liste des projets non Pareto-dominés de la Feuille 2 d'Excel serait la suivante :

	1	2	3
1	**Nom Projet**	**Coût**	
2	AAA	25 000 €	31,6666667
3	ABC	17 000 €	34,3333333
4	FAQ	8 000 €	32

FIGURE 3 – Liste des projets non Pareto-dominés (Feuille 2 d'Excel)

Dorénavant, on ne sait donc pas combien de projets sont listés dans la Feuille 2 d'Excel. Si 3 projets non Pareto-dominés sont listés dans la Feuille 2 d'Excel par exemple, alors la ligne 5 et les suivantes sont vides. On peut remarquer qu'une cellule vide a la valeur `""`.

Q 1.6 (1.5 pt) Ecrivez en VBA la fonction `meilleur` qui prend en paramètre un numéro de ligne `li` et retourne la ligne du projet qui a obtenu la meilleure moyenne (on suppose qu'il n'y en a qu'un) parmi les projets listés à partir de la ligne `li`. Sur l'exemple de la figure 3, si l'on suppose qu'il n'y a que 3 projets non Pareto-dominés, `meilleur(2)` retournerait 3 et `meilleur(4)` retournerait 4.

Q 1.7 (3 pts) Ecrivez en VBA la macro `tri` qui trie les projets dans l'ordre décroissant de leur moyenne. Dans cette question, on ne cherche pas à optimiser le nombre d'opérations effectuées par votre procédure, vous pouvez donc utiliser l'algorithme de tri que vous voulez. L'exécution de la macro `tri` sur l'exemple de la figure 3 donnerait le résultat suivant (si l'on suppose qu'il n'y a que 3 projets non Pareto-dominés) :

	1	2	3
1	**Nom Projet**	**Coût**	
2	ABC	17 000 €	34,3333333
3	FAQ	8 000 €	32
4	AAA	25 000 €	31,6666667

FIGURE 4 – Liste des projets non Pareto-dominés triés

Q 1.8 (1.5 pt) La société dispose d'un budget b. Elle décide de financer tous les meilleurs projets (selon leur moyenne) dans la limite de son budget b. Ainsi si la société a un budget de 11 k€ et que les trois meilleurs projets ont des coûts de 2k€ pour le premier, 7k€ pour le deuxième et 4 k€ pour le troisième, alors la société financera les deux meilleurs projets. Si la société disposait d'un budget de 8 k€ par exemple, elle ne financerait que le meilleur projet. Ecrivez la procédure `ProjetAcceptes` qui prend en paramètre un budget `b` et affiche successivement le nom de tous les projets qui seront financés par la société. Pour cela, la procédure `ProjetAcceptes` trie les projets dans l'ordre décroissant de leur moyenne puis affiche successivement les noms des projets acceptés (en fonction du budget).

Exercice 2 : Calcul sur des entiers (6 pts)

On considère dans cet exercice le problème du calcul sur des grands nombres entiers. On appelle grand nombre entier un entier pouvant contenir jusqu'à 40 chiffres. Des calculs avec de tels grands nombres ne peuvent pas être effectués avec des calculettes simples car on dépasse les capacités de représentation des calculettes. On cherche donc ici à effectuer les calculs avec de grands nombres entiers à l'aide de VBA. On considère qu'un grand nombre entier est représenté dans la feuille Excel sur une ligne à raison d'un chiffre par colonne. On considère que les grands nombres entiers n'ont pas plus de 40 chiffres et on utilise donc les 40 premières colonnes de la feuille Excel pour représenter les nombres. Ainsi le nombre 45 123 sera réprésenté dans la feuille Excel en mettant le chiffre 3 dans la colonne 40, le chiffre 2 dans la colonne 39, le chiffre 1 dans la colonne 38, le chiffre 5 dans la colonne 37, et le chiffre 4 dans la colonne 36. Les colonnes 1 à 35 seront vides ou remplies de 0.

Q 2.1 (2 pts) Ecrivez en VBA la procédure `additionne` qui prend en paramètre trois numéros de lignes `l1`, `l2` et `l3` et remplit la ligne `l3` par la somme des deux nombres représentés aux lignes `l1` et `l2`. Ainsi si la ligne 1 contient le nombre 234 et la ligne 2 contient le nombre 862, alors l'exécution de :
`additionne 1, 2, 3`
place à la ligne 3 le nombre 1096. Si la somme des deux grands nombres est un grand nombre de plus de 40 chiffres, alors la procédure `additionne` effectue la somme sur les 40 chiffres de droite et affiche le message « Dépassement de capacité ».

Q 2.2 (2 pts) Ecrivez en VBA la fonction `plusGrand` qui prend en paramètre deux numéros de ligne `l1` et `l2` et retourne le numéro de ligne qui contient le plus grand des deux nombres entre celui de la ligne `l1` et celui de la ligne `l2`. On suppose que les deux nombres sont différents.

Q 2.3 (2 pts) On suppose dans cette question (uniquement) que les grands nombres entiers sont binaires. Les chiffres de la feuille Excel ne sont donc que des 0 ou des 1. Les nombres représentés dans la feuille sont donc des nombres binaires représentés sur 40 bits. Ecrivez en VBA la fonction `binToDec` qui prend un numéro de ligne `li` en paramètre et retourne la valeur décimale (i.e. en base 10) du nombre binaire représenté à la ligne `li`. Par exemple, si la ligne 1 contient le nombre 100110, alors `binToDec(1)` retourne 38.
NB : on peut remarquer que $2^{40} = 1099511627776$.

Annexe

Classe `Range`

La classe `Range` est la classe des plages de cellules. Entre autres, elle possède les propriétés suivantes :

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

Property Range(Cell1 As Range, Cell2 As Range) As Range
' Lecture
' Plage dont les coins (supérieur gauche et inférieur droit) sont Cell1
' et Cell2 relativement à l'objet Range

Property Value As Variant
' Lecture-écriture
' Valeur de la cellule

Property FormulaR1C1Local As String
' Lecture-écriture
' Formule de la cellule (affichée dans la barre de formule)

Property Interior As Interior
' Lecture-écriture
' Objet qui représente le fond des cellules de la plage

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

Classe `Font`

La classe `Font` est la classe des styles de caractères. Entre autres, elle possède les propriétés suivantes :

```
Property Bold As Boolean
' Lecture-écriture
' Graisse des caractères.
```

Classe `Interior`

La classe `Interior` est la classe des fonds de cellules. Entre autres, elle possède les propriétés suivantes :

```
Property Color As Long
' Lecture-écriture
' Couleur (codée en RGB) du fond.
```

Types de données

Les différents types de données en VBA sont les suivants :

- types numériques :
  - type **Byte** : entiers compris entre 0 et 255 (8 bits)
  - type **Integer** : entiers compris entre -32768 et 32 767 (16 bits)
  - type **Long** : entiers compris entre -2 147 483 648 et 2 147 483 647 (32 bits)
  - type **Single** : réels compris entre $-3,402823 \times 10^{38}$ et $-1,401298 \times 10^{-45}$ pour les nombres négatifs et entre $1,401298 \times 10^{-45}$ et $3,402823 \times 10^{38}$ pour les nombres positifs
  - type **Double** : réels compris entre $-1,79769313486231 \times 10^{308}$ et $-4,94065645841247 \times 10^{-324}$ pour les nombres négatifs et entre $4,94065645841247 \times 10^{-324}$ et $1,79769313486231 \times 10^{308}$ pour les nombres positifs
- type **Boolean** : qui vaut **False** ou **True**
- type **String** : chaîne de caractères (texte mis entre guillemets)
- type **Date** : permet de mémoriser dates et heures ainsi que des durées
- type **Variant** : union de tous les types différents

Fonction `Application.InputBox`

```
Function Application.InputBox (Prompt As String, _
                               Optional Title As String, _
                               Optional Default As String, _
                               Optional Left As Integer, _
                               Optional Top As Integer, _
                               Optional HelpFile As String, _
                               Optional HelpContextId As Variant, _
                               Optional Type As Integer) As Variant
' Affiche une boîte de dialogue qui comporte un cadre de saisie et des
' boutons OK et Annuler et renvoie les informations saisies.
```

Valeur	Type de données renvoyé
0	Formule
1	Valeur numérique
2	Chaîne de caractères
4	Valeur booléenne (**True** ou **False**)
8	Référence de cellule (objet **range**)
16	Valeur d'erreur

TABLE 1 – Valeurs du paramètre `Type` de `Application.InputBox`

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.
```