TP 8 - Récursivité

Exercice 1 : Affichage ordonné

Q1.1 Que fait la procédure `affichage` ? Vérifier en écrivant une macro VBA qui l'appelle.

```
Sub affichage(n As Integer)
    If n >= 0 Then
        MsgBox n
        affichage n - 1
    End If
End Sub
```

Q1.2 Modifier la procédure `affichage` pour qu'elle effectue les mêmes affichages successifs mais dans l'ordre inverse.

Exercice 2 : Conjecture de Syracuse

Soit la fonction s suivante :

$$s(n) = \begin{cases} 1 & \text{si } n = 0 \text{ ou } 1 \\ s(\frac{n}{2}) & \text{si } n \text{ est pair} \\ s(3n+1) & \text{sinon} \end{cases}$$

Si la conjecture de Syracuse est vérifiée, alors $s(n) = 1$ quel que soit n positif.

Q2.1 Écrire en VBA une fonction récursive `syra` qui prend un entier n en paramètre et qui retourne $s(n)$. On suppose que n est positif ou nul. Vérifier la conjecture de Syracuse.

Q2.2 Écrire une version itérative du calcul de $s(n)$ (cf exercice 3 du TP4). Vérifier que les résultats obtenus sont bien les mêmes pour la version itérative et pour la version récursive en affichant les valeurs intermédiaires.

Exercice 3 : Puissance

Q3.1 Écrire en VBA une fonction récursive `puiss` qui prend deux entiers x et n en paramètre et qui retourne x^n, sans utiliser l'opérateur $\wedge$. On suppose que n est positif ou nul.

Q3.2 Tester la fonction VBA `puiss` sous Excel, et comparer les résultats obtenus à ceux donnés par la fonction `PUISSANCE` d'Excel.

Q3.3 Écrire une macro VBA qui demande à l'utilisateur de saisir deux entiers (types numériques) x et n, et qui affiche la valeur x^n et le nombre d'appels récursifs effectué par `puiss`. Pour compter le nombre d'appels récursifs, on peut utiliser une variable globale, incrémentée à chaque appel, ou stocker un compteur dans une cellule de la feuille Excel, lui aussi incrémenté à chaque appel.

Q3.4 Sachant que $x^0 = 1, x^n = (x^2)^{n/2}$ si n est pair et $x^n = x \cdot (x^2)^{(n-1)/2}$ si n est impair, écrire une autre fonction récursive `puissMaligne` calculant x^n. Tester et comparer le nombre d'appels des deux fonctions

récursives.

Exercice 4 : Tri fusion

On reprend la comparaison des algorithmes de tri du TP précédent. Les questions de cet exercice sont donc à faire dans le fichier du TP10 sur le tri, à la suite de ce qui a été fait (écriture en VBA de la macro génération, des algorithmes de tri par sélection et de tri à bulles). On rappelle que l'on génère aléatoirement des entiers en colonne 1 de la feuille Excel, on recopie ces entiers en colonnes 2 et 3, puis on les trie par un tri par sélection en colonne 2, et par un tri à bulles en colonne 3. La colonne 4 va être utilisée pour y effectuer un tri fusion, après recopie des entiers non triés de la colonne 1.
Pour cela, on va utiliser la procédure récursive `triFusion` suivante :

```
Sub triFusion(col As Long, deb As Long, fin As Long)
        Dim milieu As Long
        If deb >= fin Then
                Exit sub
        End if
        'Copie des éléments de col à col+1 pour les trier
        Range(Cells(deb, col+1), Cells(fin, col + 1)).Value = _
              Range(Cells(deb,col),Cells(fin, col)).Value
        milieu = (deb + fin) \ 2
        'tri récursif de la première moitié dans col + 1
        ............
        'tri récursif de la seconde moitié dans col + 1
        ............
        'interclasser les deux moitiés précédentes dans col
        ............
        'effacer les éléments dans col + 1
        Range(Cells(deb, col + 1), Cells(fin, col + 1)).clearcontents
End Sub
```

`triFusion` trie les éléments de la colonne `col` de la ligne `deb` à la ligne `fin` en effectuant les opérations suivantes :

- copie des éléments dans la colonne suivante (`col`+1) ;
- tri par un algorithme de tri fusion de la première moitié des éléments de la colonne `col`+1 ;
- tri par un algorithme de tri fusion de la seconde moitié des éléments de la colonne `col`+1 ;
- fusion (interclassement) dans la colonne `col` des deux moitiés triées de la colonne `col`+1 en une collection triée ;
- suppression des éléments de la colonne `col`+1.

Q4.1 Écrire la procédure d'interclassement qui prend les numéros de colonne `c` et de ligne `d1`, `f1`, `d2` et `f2`, et qui place en colonne `c-1` à partir de la ligne `d1` le résultat de la fusion triée (interclassement) de deux collections triées d'éléments situées en colonne `c`, l'une de la ligne `d1` à la ligne `f1` et l'autre de la ligne `d2` à la ligne `f2`.
Pour interclasser, on utilise deux variables `pos1` et `pos2` indiquant la ligne du plus petit élément courant dans chacune des deux collections. On prend le plus petit de ces deux premiers éléments courants et on le place à la position courante dans la colonne `c`-1 (i-ème position pour la i-ème selection d'un élément). On met à jour la variable de la position du plus petit élément courant correspondante (si le plus petit élément

était dans la première collection par exemple, alors la variable `pos1` est incrémentée de 1). On réapplique le même procédé jusqu'à ce que les deux collections à interclasser ait été parcourues intégralement.

Q4.2 Recopier et compléter maintenant la procédure `triFusion` donnée ci-dessus.

Q4.3 Écrire la macro `triFus` qui place les éléments de la première colonne de la feuille de calcul dans la quatrième colonne dans l'ordre croissant. Pour cela, `TriFus` copie les éléments de L1C1 :L300C1 dans L1C4 :L300C4, puis applique l'algorithme de tri fusion pour trier les éléments de la quatrième colonne en faisant appel à la procédure récursive `triFusion`.

Q4.4 Modifier la macro `verifieTris` du TP précédent pour qu'elle teste aussi le résultat de l'exécution de `TriFusion`.

Q4.5 Comparer les temps d'exécution des trois algorithmes de tri. Quelle est la méthode de tri la plus efficace ? Pourriez-vous expliquer pourquoi ?