ALGORITHMIQUE / Pascal - SERIE N° 4

OBJECTIFS PEDAGOGIQUES : Manipulation des objets structurés de type tableau à une et à deux dimensions, chaînes de caractères et enregistrements. Construction détaillée des algorithmes de tri dont les principes généraux ont été donnés en cours.

L'approche modulaire doit être appliquée SYSTEMATIQUEMENT.

I. À traiter en TD

Exercice 1 : Rechercher le nombre de suites d'ordre croissant contenues dans un tableau à une dimension.

Nota : une suite est composée d'au moins 2 éléments)

Exercice 2 : Comment obtenir un tableau à une dimension (RES) trié (sans utilisation des tris). A partir de n tableaux à une dimension triés.

Exercice 3 : Tri « bulles ». Les éléments mal classés remontent dans le tableau comme des bulles à la surface d'un liquide. La « remontée » des éléments s'effectue avec des permutations si élém1>élém2. Il est évident que plusieurs passages sur l'ensemble des éléments sont nécessaires.

Exercice 4 : Tri par comptage avec 3 tables. . Il se fait en 2 étapes. Dans la première étape, pour chaque élément on compte le nombre d'éléments qui lui sont inférieurs et on range le résultat trouvé dans une table de comptage (dans des cases de même indice bien entendu) de taille identique à celle du tableau à trier.

Dans la deuxième, chaque élément de la table initiale est mis dans son emplacement (puisque l'on a le nombre d'éléments qui lui sont inférieurs) dans la troisième table.

Exercice 5 : Construire la matrice transposée (A') d'une matrice. (Rappel : si A(m,n) est la matrice de départ, sa transposée est A' (n,m)).

Exercice 6 : Sachant qu'un « point de selle » est, en même temps, l'élément le plus grand de la ligne et le plus petit de la colonne ou il se trouve. Rechercher le point de selle d'une matrice

	1	2	3	4	5
1	30	1	38	24	5
2	1	5	36	2	3
3	15	18	**32**	28	9
4	38	4	37	3	16
5	4	7	51	8	12

exemple

Exercice 7 : Des températures sont relevés chaque jour de la semaine et toutes les heures. Les données de la semaine se trouveront dans un tableau de sept éléments, dont chaque élément représente un jour. De plus, chaque jour est représenté par un enregistrement qui contiendra : la date sous la forme JJMMAAAA et un tableau de 24 éléments, dont chaque case contient la température d'une heure de la journée.

On voudrait avoir la température moyenne d'un jour donné et celle d'une heure donnée de la semaine.

Nota : afin de faciliter le remplissage des tableaux contenant les températures, nous vous suggérons de les remplir avec des nombres aléatoires compris entre -10 et +50.

Exercice 8 : Lors d'un concours de Sudoku on souhaite garder les noms des candidats, leurs âges, la grille initiale, le niveau de la grille, la grille remplie, et le temps mis (en secondes) pour la remplir.

1. Proposer une structure de données sachant que l'on a au maximum 100 candidats.
2. Comment remplir les données de x candidats (x<=100) ?
3. Comment afficher l'enregistrement d'un candidat donné ?

Nota : pour notre jeu d'essai et afin de gagner du temps, les grilles de Sudoku seront remplies avec des nombres aléatoires

Pour Info :

La grille de jeu sudoku est une matrice de 9 cases de côté, subdivisé en 9 carrés de 3 par 3 appelés "régions". Le jeu consiste à remplir les cases de la grille afin que tous les chiffres de 1 à 9 figurent une seule fois dans chaque colonne, dans chaque ligne et dans chaque région.

En d'autres termes, chaque colonne, chaque ligne et chaque région doit contenir tous les chiffres de 1 à 9.

Exemple :

Voici un exemple d'une grille sudoku valide (Pour la valeur 5, on peut bien vérifier qu'il ne figure qu'une seule fois dans sa région, sa ligne et sa colonne)

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

Exercice 9 : Calcul de la fréquence d'apparition d'une sous-chaîne dans une chaîne de caractères.

Exercice 10 : Recherchez les mots d'une chaîne comprenant n fois un caractère donné. (Les mots sont séparés par un ou plusieurs blancs)

Exercice 11 :

La méthode RLE (En anglais Run Length Encoding), ou encore codage de la longueur des séquences, permet de réduire la taille des chaînes de caractères en remplaçant les séquences de caractères identiques par une indication du caractère à répéter ainsi que du nombre de répétitions. Quand un caractère apparaît n fois de suite en entrée, il n'apparaît en sortie qu'une fois, précédé de [n].

Par exemple : la chaîne « axxxxxxb » sera compressée en « a[6]xb ».

Ceendand, quand un caractère est répété seulement deux ou trois fois, la chaîne compressée devient en fait plus longue que l'originale. A cet effet, seul les chaînes de caractère supérieur à un « seuil » de compression donnée le seront. Ainsi, si un caractère est répété un nombre de fois inférieur au seuil, la compression n'est pas effectuée. Par ailleurs, la compression, telle qu'elle est réalisée, peut introduire des ambiguïtés. En effet, si l'on rencontre en entrée les chaînes « abbbc » et « a[3]bc », les deux seront compressées de la même façon. Pour éviter ceci, si la chaîne d'entrée contient un crochet ouvrant « [», en sortie il sera doublé « [[». Ainsi, les deux chaînes précédentes deviendront respectivement après compression « a[3]bc » et « a[[3]bc ».

Travail demandé :

1) Proposer une solution modulaire (avec justification) comportant tous les modules jugés nécessaires de compression avec un seuil « d » donné (compresseRLE) et de décompression (décompresseRLE).

2) On souhaite garder le message initial, sa longueur, le type de traitement effectué (compression ou décompression), le seuil de compression, le message résultant. Proposez une structure de données (2 pts).

3) Ecrire l'algorithme du module qui permet de rechercher le message compressé, s'il existe, d'un message donné (2 pts).

À traiter en Ateliers

Exercice 1 :

Nous disposons d'un tableau X à une dimension et contenant des nombres entiers dont certains appartiennent à la suite de Fibonacci (1, 1,2, 3, 5,…) , d'autres ont une partie centrale composée des mêmes chiffres (exples : 1722259, 56777734) et des intrus, qui n'appartiennent à aucune des deux catégories précédemment citées.

Travail à Faire :

Partie A (à traiter complètement) :

Nous souhaitons construire le tableau (R) à deux dimensions de la façon suivante.

La 1ère ligne contiendra tous les éléments de X appartenant à la suite de fibonnacci

La 2ième ligne contiendra tous les éléments dont la partie centrale est composée des mêmes chiffres

La 3ième ligne contiendra tous les intrus !

Partie B : Si nous disposons de plusieurs tableaux de la nature de X et on voudrait associer chacun d'eux à son tableau résultat (R) et les garder dans un enregistrement, Donnez la description de cet enregistrement

Exercice 2 :

Nous disposons d'un objet qui contient un caractère, un entier, un booléen et un tableau de 3 lignes et 3 colonnes dont chaque élément contient un mot. On commence par remplir les mots du tableau et ensuite nous souhaitons savoir la fréquence d'apparitions d'un caractère donné et s'il existe dans plus de un mot. Alors on met à jour notre enregistrement en mettant dans la variable de type caractère : le caractère donné, dans celle de type entier : la fréquence et dans celle de type booléen : vrai si le caractère existe dans plus de 1 mot. Et on imprime cet enregistrement.

Exercice 3 : Problématique : Compression et Décompression d'un caractère.

Lors de la conception d'un logiciel de traitement d'images pour les caractères, chaque caractère est représenté par un tableau (matrice) M [7,5] à valeurs binaires (Gris = 1 et blanc = 0). Par exemple :

1. →	0	0	1	0	0
2. →	0	1	0	1	0
3. →	1	0	0	0	1
4. →	1	1	1	1	1
5. →	1	0	0	0	1
6. →	1	0	0	0	1
7. →	1	0	0	0	1

1. →	1	1	1	1	1
2. →	1	0	0	0	1
3. →	1	0	0	0	1
4. →	1	1	1	1	1
5. →	0	0	0	0	1
6. →	0	0	0	0	1
7. →	1	1	1	1	1

1. Le processus de Compression se présente comme suit :

A. Lecture des valeurs binaires de la Matrice M [7,5].

B. La matrice M [7, 5] du caractère est convertie en un vecteur V à valeurs binaires. Par exemple pour les deux caractères précédents on aura :

– Pour le A : V = 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1

– Pour le 9 : V = 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1

C. Le vecteur V est compressé en un vecteur C de manière suivante :

– C [1] = V [1] ;

Pour les autres éléments, ils présentent le nombre de répétition d'une cellule binaire. Par exemple pour les caractères précédents :

– Pour le A : C = 0 2 1 3 1 1 1 1 1 3 7 3 2 3 2 3 1

– Pour le 9 : C = 1 6 3 2 3 6 4 1 4 6

2. Le processus de décompression se fait selon l'ordre inverse.

Travail demandé :

Proposer une solution modulaire (avec justification) comportant tous les modules jugés nécessaires de lecture, de conversion, de compression, de décompression et d'affichage. Prévoir un menu.

NB :

– Afin de ne pas avoir à saisir à chaque fois le caractère de votre jeu d'essai une solution est envisageable, c'est celle de mettre dans un premier temps les valeurs du caractère au début de votre programme.

Par exemple (pour le A) : M [1, 1] := 0 ; M [1, 2] := 0 ; M [1, 3] := 1 ; M [1, 4] := 0 ; M [1, 5] := 0 ; …… ; M [7,5] := 1 ;

– Si vous ne terminez pas la réalisation de TOUS les modules, mais seulement une partie : N'oubliez pas de mettre les petits programmes qui servent UNIQUEMENT à tester vos modules et faciliter la correction.

– Si vous utilisez des modules déjà faits : notez simplement leurs en-têtes et leurs rôles.

I.Exercices supplémentaires

Exercice 1 : Convertir un nombre entier dans une base quelconque comprise entre 2 et 16.

Exercice 2 : On voudrait remplir automatiquement un tableau (R) de 10 lignes et 20 colonnes avec des nombres aléatoires compris entre 0 et 5000.

Puis les éléments de R seront mis dans un tableau (X) à une dimension que l'on triera.

Une fois que X sera trié on affichera, tous les éléments qui sont premiers et ceux qui sont composés de 2 parties identiques (exemples : 77, 4949), de la manière suivante :

23 Premier

1414 2 parties distinctes

Exercice 3 :

Dans un tableau à 2 dimensions, il y a des éléments dont le contenu est égal à la somme des 4 cases qui ont un coté adjacent (un coté commun) avec chacun d'eux,

Exemple :

2	36	1
6	72	20
7	10	12

On constate, sur l'exemple, que 72 = 36 + 6 + 20 + 10 .

Il vous est demandé de trouver tous ces éléments, s'il y en a, et de les mettre dans un tableau à 2 dimensions aussi, dont chaque ligne contient la valeur de l'élément, son indice de ligne et son indice de colonne.

Puis une fois que ce tableau est rempli, il faut composer une chaîne de caractères à partir des valeurs trouvées dans l'étape précédente. (Chaque valeur représentant le rang d'un caractère dans la liste des caractères ASCII.)

Prenez votre temps et observez attentivement l'exemple complet ci-dessous :

```
Matrice de depart :

      1       3      36       8      15       9       1
      0       6      72      20      69      25       2
     35      76      10      76       9       0      27
      4      25      79      37      11      10       7
      8       1       7       1       0      14      17
```

```
les elements recherches sont :

     Valeur   ligne   colonne
     72       2       3
     69       2       5
     76       3       2
     76       3       4
     79       4       3
```

```
la chaine est :HELLO
```

Jeu d'essai final :

```
Matrice de depart :

      2       3       4      33       1      14       7
      2      69       1      85      46      82       2
      1      63      69       5      75      20       8
      1      65       0      32       4      33       6
      3       1       4      23       2       3       1
```

Exercice 4 : Supprimer un mot d'une chaîne avec décalage vers la gauche des mots qui le suivent.

Exercice 5 : Compter les occurrences (fréquence d'apparitions) du 1er mot d'une chaîne de caractères.

Exercice 6 : Le propriétaire d'un coffre-fort a oublié le code pour l'ouvrir. Il se rappelle quand même que :

- lorsqu'il prend le nombre constitué par le chiffre des milliers suivi de celui des dizaines, puis de celui des unités et enfin celui des centaines du code, les chiffres de ce nombre sont en ordre croissant et de plus ce nombre est premier,
- la somme de tous les chiffres du code est égale à 27
- Le code est composé de 5 chiffres et de plus c'est un carré parfait
  Pouvez-vous, dans un premier temps, l'aider à retrouver ce code ?

Nota : un carré parfait est tout simplement le carré d'un entier (exples 25 (= 5^2), 784 (= 28^2) sont des carrés parfaits

Ensuite on souhaite construire une matrice carrée T (10,10) , que l'on remplira ligne par ligne de la manière suivante : chaque case T[i,j] , doit avoir une structure d'enregistrement qui contiendra un nombre et son carré parfait. On commencera à remplir cette matrice par le nombre 200 (donc T(1,1] contiendra 200 et 40000, T[1,2] contiendra 201 et 40401,…).

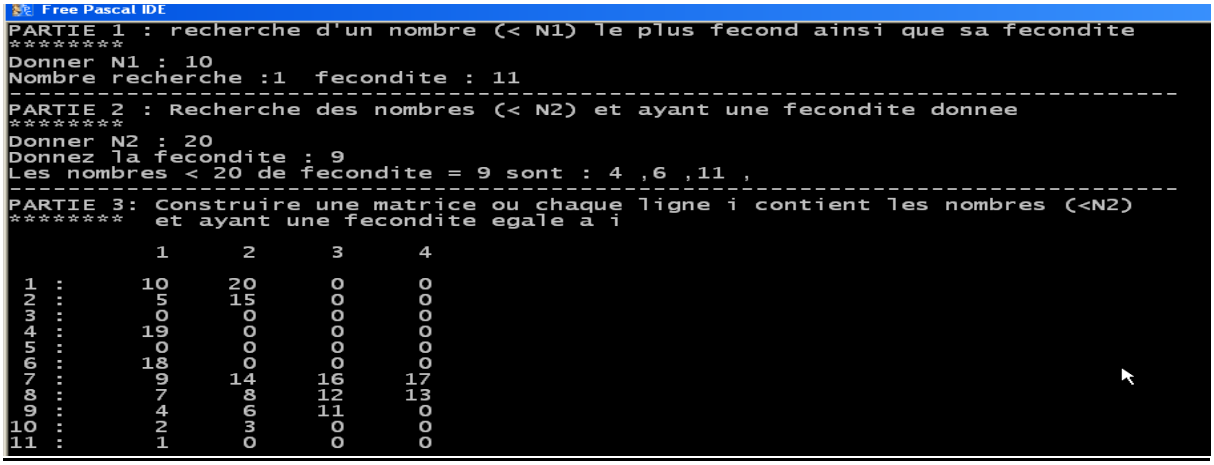
On vous demande de retrouver la ligne et la colonne qui contient le code secret trouvé, puis de les afficher ainsi que le nombre dont il est le carré parfait.

Exercice 6 : Lorsque vous prenez un nombre N, que vous lui rajoutez le produit de ses chiffres, et que vous refaites la même chose jusqu'à ce que le chiffre zéro apparaisse. Vous construisez une suite dont le nombre d'éléments s'appelle la fécondité de N. Exemple : pour N = 36 la suite est 36, 54, 74, 102. La fécondité de 36 est 4

Nota : par convention si un nombre a un seul chiffre, on rajoute ce nombre. Pour N = 7 la suite est : 7, 14, 18, 26, 38, 62, 74, 102.

La fécondité de 7 est 8.

Le travail à faire est décrit dans l'écran de sortie suivant :



Exercice 6 : Vous connaissez surement les carrés magiques. Non ! eh bien il s'agit de ces tableaux d'ordre (n) impair dans lesquels la somme des éléments de chaque ligne, celle des éléments de chaque colonne et celles des éléments des deux diagonales sont égales, de plus les cases sont remplies par les nombres allant de 1 à n^2.

Par exemple dans le carré magique d'ordre 3, les sommes des lignes, des colonnes et des diagonales sont égales à 15, et les cases sont remplies par les nombres allant de 1 à 3^2 c'est-à-dire 1,2,3,4,5,6,7,8 et 9.

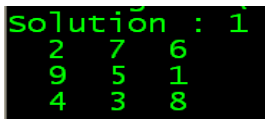
ordre 3

4	9	2
3	5	7
8	1	6

= 15

Nous allons nous limiter à l'ordre 3, car au-delà le problème est un plus complexe, et vous demander :

1. de chercher automatiquement s'il y a plusieurs solutions, leur nombre éventuellement et de les afficher une à une selon le format suivant :



2. ensuite, nous souhaitons stocker pour chaque solution : son N° et la solution elle-même (c'est-à- dire un carré magique d'ordre3). Proposez une structure de données.
3. comment faire pour afficher une solution à partir de son numéro ?

Exercice 7 : Tri par transposition. Deux éléments qui se suivent sont comparés puis permutés si le $2^{ème}$ élément est plus petit que le premier, puis un retour en arrière est effectué afin de vérifier si l'ordre n'a pas été modifié, auquel cas on le rétablit.

Exercice 8 : Tri par comptage avec 2 tables.

Il se fait aussi en 2 étapes.

1. Dans la première étape, on construit le tableau de comptage ;
2. Dans la deuxième, on parcourt le tableau de comptage et l'élément pointé, dans le tableau à trier, est permuté avec l'élément qui se trouve à son emplacement (puisque l'on connaît le nombre d'éléments qui lui sont inférieurs). Et, il ne faudra pas oublier de permuter aussi les éléments de la zone de comptage en même temps que ceux du tableau à trier).

Attention aux cas ou des éléments du tableau à trier sont identiques

Exercice 9 : Tri par comptage avec 1 table.

Il n'y a ni table de comptage, ni table résultat. Autrement dit dès que l'on trouve le nombre d'éléments inférieurs à un élément, ce dernier sera mis à son emplacement. Mais attention aussi aux éléments de même valeur.