Série d'exercices N°2
Les fichiers de données

EXERCICE 1

Soient le tableau de déclaration des nouveaux types et le tableau de déclaration des objets suivants :

TDNT

Type
Renseignement = Enregistrement Nom : Chaîne Num : Entier Etat_civil : caractère Fin Eleve = Fichier de Renseignement Fent = Fichier d'entier Tsec = Tableau de 50 Renseignement

TDO

Objet	Type/Nature
E	Renseignement
P	Booléen
B	Entier
M	Réel
Felev	Eleve
Fe	Fent
T	Tsec

Dans le tableau ci-dessous, valider chacune des instructions en mettant dans la case correspondante de la 2ème colonne la lettre **V** si l'instruction est valide ou la lettre **F** dans le cas contraire.

Justifier la réponse si l'instruction est invalide.

Instruction	Valide/Invalide	Justification
B ← Fin_Fichier (Fe)	F	B est un entier et fin fichier (Fe) retourner booléen
T[2] ← E.Num	F	T[2] est enregistrement et E.Num entier
P ← E.Num>10	V	~~Felev fichier d'enregistrement et E~~
Ecrire (Felev, E.Nom)	F	Felev fichier d'enregistrement et E.Nom chaine
Ecrire (Fe, M)	~~V~~ F	
P ← E.Num > B	V	
Lire (Fe, T[1].Num)	~~X~~ F	

EXERCICE 2

Un entier est dit cubique ou d'Armstrong s'il est égal à la somme des cubes de ses chiffres.

Exemple : 153 est cubique car $153 = 1^3 + 5^3 + 3^3$.

On se propose d'écrire un programme qui permet de :

- Remplir un fichier nommé « **Nombres.dat** » par n entiers positifs de 3 chiffres, avec ($4 \leq n \leq 20$).
- Afficher tous les entiers d'Armstrong du fichier « **Nombres.dat** » suivis de leurs positions.

Exemple :

Pour **n=5** et le fichier **Nombres.dat** ci-après : ⇨ Le programme affiche :

100
407
153
921
371

Les entiers cubiques sont :
407 de position 2
153 de position 3
371 de position 5

EXERCICE 3

Ecrire un programme permettant de :

- Remplir un fichier nommé « **Alphabet.dat** » par des lettres alphabétiques. La saisie s'arrête une fois que la réponse à la question « **Saisir une autre Lettre O/N ?** » soit « **N** » ou « **n** »
- Afficher le nombre des voyelles et celui de consonnes.

EXERCICE 4

Ecrire un programme permettant de :

- Remplir un fichier nommé « **NbresAlea.dat** » par n entiers aléatoires compris entre 1 et 1000 ($5<n<100$)
- Trier le contenu du fichier par ordre croissant.
- Afficher le fichier trié.

EXERCICE 5

On se propose de remplir un fichier **F** nommé physiquement « **C:\Entiers.dat** » par les informations relatives à n entiers positifs. ($3 \leq n \leq 100$).

Chaque entier est caractérisé par :

- **val :** sa valeur.
- **som :** somme de ses chiffres.
- **Test :** prend la valeur Vrai si les chiffres de val forment une suite croissante, sinon prend la valeur Faux.

Travail demandé :

1) Donner le tableau de déclaration des nouveaux types (TDNT) correspondant à ce programme.
2) Ecrire l'instruction d'ouverture du fichier **F**.
3) Donner un algorithme d'un module permettant de remplir le fichier « **C:\Entiers.dat** »

EXERCICE 6

Soit un fichier **F** nommé physiquement « **Produit.dat** » rempli par des produits.

Chaque produit est caractérisé par **un code** (chaîne), **un prix unitaire** (réel positif) et **une quantité** (entier positif).

Travail demandé :

1) Donner le tableau de déclaration des nouveaux types correspondant.
2) Ecrire un algorithme d'un module permettant d'ajouter **5%** aux prix des produits dont la quantité est inférieure ou égale à 5.

EXERCICE 7

On se propose d'écrire un programme permettant de :

- Remplir un fichier « **Eleve.dat** » par des élèves, la saisie s'arrête lorsque l'utilisateur répond à la question « **Voulez-vous continuer : O/N ?** » par « **N** » ou « **n** »

  Chaque élève est identifié par :
  - **Un nom et prénom :** chaîne non vide.
  - **Sa classe :** chaîne non vide.
  - **Le nombre d'Absences :** entier supérieur ou égal à zéro.
- Supprimer les élèves éliminés du fichier. Un élève est éliminé si le nombre d'absences dépasse 10.
- Trier les élèves (non éliminés) par ordre croissant des classes, et les afficher.
- Afficher le nombre d'élèves ayant un nombre d'absence nul.

EXERCICE 8

Soit **M** une matrice carrée de **n*n** entiers supposés distincts avec $3 \leq n \leq 20$. Chaque ligne contient un seul maximum.

On désigne par colonne dominante d'une matrice, la colonne qui contient le plus de maximums des lignes de cette matrice.

Exemple :

Pour n=5 et la matrice M ci-contre, la 4ème colonne est la colonne dominante puisque parmi les 5 maximums des 5 lignes il y a 3 maximums qui se trouvent dans la 4ème colonne.

M

32	12	10	**89**	15
3	**33**	14	1	18
54	5	22	**76**	50
34	21	6	29	17
19	9	11	**84**	25

On se propose de générer un fichier d'enregistrements « **F_Max.dat** » à partir de **M**. chaque enregistrement contient la valeur **Vmax** du maximum d'une ligne de la matrice **M**, le numéro **NL** de la ligne et le numéro **NC** de la colonne de **Vmax**

Exemple :
Pour la matrice **M** présentée ci-dessus, le premier enregistrement du fichier **F_Max.dat** contiendra les valeurs suivantes : **Vmax = 89, NL=0, NC=3**

Travail demandé :
Ecrire un programme qui permet de :

- Remplir une matrice **M** par **n** entiers ($3 \leq n \leq 20$)
- Remplir le fichier **F_Max.dat** nommé logiquement **F.**
- Chercher et afficher le numéro de la colonne dominante à partir du fichier **F_Max.dat**, en utilisant la fonction **Frequence (NC, F)** qui retourne le nombre d'occurrences d'un numéro de colonne **NC** dans le fichier d'enregistrements **F**.

NB :

- Lorsque plus qu'une colonne est dominante, on affichera la dernière rencontrée dans le fichier.
- Le candidat n'est pas appelé à la saisie de **n** et au remplissage de la matrice **M**.