LEÇON 2 : LES FICHIERS

UVCI

Table des matières

1- Généralité sur les fichiers

I

Définition : 1.1- Définition

Un *fichier* est une séquence de données du même types enregistrés sur un support informatique. Il sert à stocker des informations de manière permanente, entre deux exécutions d'un programme. Un fichier est toujours conservé sur un support externe à la mémoire centrale.

1.2- Les Types de fichiers

Nous distinguerons les fichiers textes et les fichiers binaires.

- *Les fichiers textes :* Un fichier texte est formé de caractères ASCII, organisé en ligne, chacune se termine par un caractère de contrôle de fin de ligne. Si chaque ligne contient le même genre d'informations, les lignes sont appelées des enregistrements.
- *Les fichiers binaires :* Un fichier binaire contient des données non textuelles. Il n'est pas organisé sous forme d'enregistrement. Les fichiers binaires ne prennent sens que s'ils sont traités par un programme adapté. Par exemple un fichier son, une vidéo, une image, un programme exécutable, etc.

1.3- Types d'accès aux fichiers

Le type d'accès est la technique que la machine doit suivre pour aller chercher les informations contenues dans un fichier. On distingue trois types d'accès aux fichiers :

L'accès séquentiel : Cet accès consiste à traiter les informations séquentiellement, c'est à dire dans l'ordre où elles apparaissent dans le fichier. On ne peut donc accéder à une information qu'en ayant au préalable examiné celle qui la précède.

L'accès direct : Ce type d'accès consiste à se placer directement sur l'information souhaitée sans parcourir celles qui la précèdent, en précisant la position de l'élément recherché.

L'accès indexé : Ce type d'accès combine la rapidité de l'accès direct et la simplicité de l'accès séquentiel. Il est particulièrement adapté au traitement des gros fichiers, comme les bases de données. On parcourt un index pour rechercher une clef. On obtient ainsi l'adresse exacte de l'information recherchée. On peut utiliser des opérateurs de comparaisons sur les index (=, <>, <, <=, >, >=).

Application 1 :

II

Exercice

Les types de fichiers existants sont :

- ☐ Les fichiers textes
- ☐ Les fichiers séquentiels
- ☐ Les fichiers directs
- ☐ Les fichiers binaires
- ☐ Les fichiers indexés

Exercice

Lorsqu'on veut accéder à une information, il faut au préalable examiné celle qui la précède. De quel type d'accès s'agit-il ?

- ○ Accès direct
- ○ Accès séquentiel
- ○ Accès indexé

2- Traitement sur les fichiers

III

2.1- Ouvrir un fichier

Pour effectuer une opération sur un fichier, la première chose à faire est de l'ouvrir.

Syntaxe :

Ouvrir "Nom_fichier" *sur* Num_canal *en* Mode

Nom_fichier : il représente le nom du fichier physique.

Num_canal : il représente le nom logique du fichier. Pour ouvrir un fichier il faut lui attribuer un numéro du canal disponible et valide.

Mode : Il s'agit de l'opération que l'on veut réaliser sur le fichier. Il existe 3 modes d'ouverture du fichier en l'occurrence :

- *En lecture,* on pourra uniquement récupérer les informations du fichier, sans les modifier en aucune manière.
- *En écriture,* on pourra mettre toutes les informations que l'on veut dans le fichier. Mais les informations précédentes, si elles existent, seront intégralement écrasées et on ne pourra pas accéder aux informations qui existaient précédemment.
- *En ajout,* on ne peut ni lire, ni modifier les informations existantes du fichier. Mais on pourra, ajouter de nouvelles lignes.

Exemple :

1. Ouvrir "fichier1.txt" sur 1 en Lecture // ouvre le fichier fichier1.txt en lecture à partir du canal 1
2. Ouvrir "fichier2.txt" sur 2 en Lecture // ouvre le fichier fichier2.txt en lecture à partir du canal 2

2.2- Fermer un fichier

Après avoir utiliser un fichier , il est nécessaire de le fermer pour libérer le canal qu'il occupait.

Syntaxe :

Fermer(Nom_fichier) ou *Fermer*(Num_canal)

Exemple :

fermer("fichier1.txt") ou fermer(1)

2.3- Écrire dans un fichier

Syntaxe :

EcrireFichier Num_canal,variable

Num_canal : il représente le nom logique du fichier.

variable : Il s'agit de la variable qui va contenir les innées à mettre dans le fichier, il doit être de type enregistrement.

Exemple :

Écrire un algorithme qui permet d'ajouter un enregistrement de personne dans un fichier F_Person.txt

Résolution :

```
Algorithme EcritureFichier
Type
Personne = enregistrement
nom : chaîne
prenom : chaine
cel : chaîne
ville : chaîne
finenreg
Var
Pers : Personne
i : entier
Début
Ouvrir "F_Person.txt" sur 1 en écriture // ouverture du fichier F_Person.txt en mode écriture
//********** Lecture de l'enregistrement dans le fichier via le nom logique qui est le numéro de canal **********
EcrireFichier 1,Pers
//********** Fermeture du fichier **********
Fermer(1)
Fin
```

2.3- Lire dans un fichier

Syntaxe :

LireFichier Num_canal,variable

Num_canal : il représente le nom logique du fichier.

variable : Il s'agit de la variable qui va contenir les innées à mettre dans le fichier, il doit être de type enregistrement.

Remarque :

Lorsque nous avons plusieurs données à lire dans un fichier il convient d'utiliser la fonction *EOF* (acronyme pour End Of File). Cette fonction renvoie la valeur Vrai si on a atteint la fin du fichier.

Exemple :

Écrire un algorithme qui permet lire le fichier F_Person.txt contenant les enregistrements de personne.

Résolution :

```
Algorithme LectureFichier
Type
Personne = enregistrement
nom : chaîne
prenom : chaine
cel : chaîne
ville : chaîne
finenreg
Var
Person : Personne
Début
Ouvrir "F_Person.txt" sur 1 en lecture // ouverture du fichier F_Person.txt en mode lecture
//********** Récupère les enregistrement au fur à mesure dans la variable Person jusqu'à la dernière ligne du fichier**********
Tantque Non EOF(1) faire
LireFichier 1, Person
Afficher Person
FinTq
//********** Fermeture du fichier **********
Fermer(1)
Fin
```

Application 2 :

IV

Exercice

Sujet :

Soit un fichier client.txt, écrire un algorithme qui permet de saisir un client dans le fichier client.txt à l'aide d'un enregistrement client ayant pour champs nom, prenoms, age et montantportefeuille. L'algorithme devra lire le fichier et afficher le client. Enfin on doit pouvoir ajouter trois autres clients dans le fichier client.txt.

NB :

1. La création des champs doit respecter l'ordre de leurs énumérations ci-dessus.
2. Toutes les réponses doivent être en minuscule

Algorithme TrtClient

Type

=

:

:

:

:

finenreg

Var

clt : client

i : entier

Début

sur 2 en

Afficher "Entrer le nom du client "

Saisir

Afficher "Entrer le prénom du client "

Saisir

Afficher "Entrer l'âge du client"

Saisir

Afficher "Entrer le montant du portefeuille client"

Saisir

ouvrir "client.txt" sur 1 en

Tantque Non faire

Afficher clt

Application 2 :

FinTq

ouvrir sur 1 en

Pour i ← 1 à 3 faire

Afficher "Entrer le nom du client ",i

Saisir

Afficher "Entrer le prénom du client ",i

Saisir

Afficher "Entrer l'âge du client ",i

Saisir

Afficher "Entrer le montant du portefeuille client ",i

Saisir

ecrirefichier

finpour

fin

3 - Travaux dirigés

V

Énoncé :

Soit un fichier listeetudiant.txt, écrire un algorithme qui permet de saisir 10 étudiants dans le fichier listeetudiant.txt à l'aide d'un enregistrement étudiant ayant pour champs nom, prénoms, matricule et spécialité. L'algorithme devra permettre de recherche un étudiant à partir de son nom Si le nom recherché n'existe pas, le programme devra le signaler.

Solution :

Algorithme TrtEtudiant

Type

Etudiant= Enregistrement

matricule : chaîne

nom : chaîne

prenoms : chaîne

specialite : chaîne

finenreg

Var

//********** *Permet de créer un tableau de type étudiant*********

etud : tableau[1..10] d'Etudiant

etudR : Etudiant *//permet de créer une variable étudiant pour la lecture de chaque enregistrement du fichier*

i : entier

trouver : booléen

nomR : chaîne

Début

//********** *Permet ouvrir le fichier listeetudiant.txt en mode écriture*********

Ouvrir "listeetudiant.txt" sur 2 en Ecriture

//********** *Permet remplir un tableau de type étudiant*********

Pour i ← 1 à 10 faire

Afficher "Entrer le matricule de l'étudiant N° ",i

Saisir etud[i].matricule

Afficher "Entrer le nom de l'étudiant N° ",i

Saisir etud[i].nom

Afficher "Entrer le prénom de l'étudiant N° ",i

Saisir etud[i].prenoms

Afficher "Entrer la spécialité de l'étudiant N° ",i

Saisir etud[i].specialite

finpour

*//********** Permet d'écrire les informations saisies par l'utilisateur dans le fichier listeetudiant.txt par l' intermédiaire de son canal 2**********

EcrireFichier 2, etud

*//********** Âpres écriture, nous fermons le fichier listeetudiant.txt par l'intermédiaire de son canal 2**********

Fermer(2)

*//********** Permet de donner le nom à rechercher dans la variable nomR**********

Afficher " Entrer le nom de l'étudiant à rechercher "

Saisir nomR

*//**********Permet ouvrir le fichier listeetudiant.txt en mode lecture sur la canal 1**********

Ouvrir "listeetudiant.txt" sur 1 en Lecture

*//**********Permet d'initialiser la variable trouver à faux (on suppose que l a valeur recherchée n'est pas trouvée)**********

trouver ← faux

*//**********Permet ouvrir le fichier listeetudiant.txt en mode lecture sur la canal 1**********

Tantque Non EOF(1) faire *// Vérifie si la lecture n'est pas à la fin du fichier*

LireFichier 1, etudR *// lecture de chaque ligne un fichier pour l'ajouter dans la variable etudR*

Si nomR = etudR.nom alors *//vérifie si l'information ajouter dans la variable etudR est pareille à la valeur recherchée.*

trouver ← vrai *//si la vérification est vraie alors trouver qui était à faux passe à vrai*

finsi

FinTq

*//********** Âpres la lecture, nous fermons le fichier listeetudiant.txt par l'intermédiaire de son canal 1**********

Fermer(3)

*//********** Affiche le résultat final**********

Si trouver = vrai alors

Afficher nomR ," existe dans la liste des étudiants"

Sinon

Afficher nomR ," n'existe pas dans la liste des étudiants"

Finsi

Fin