Chapitre 3 : Les fichiers

Les données utilisées dans nos programmes proviennent de deux manières :

1. Elles sont incluses dans le code source du programme.
2. Elles sont saisies lors de l'exécution du programme.

Cela ne suffit pas aux besoins réels, surtout quand les données sont nécessaires pour la prochaine exécution du programme.

Exemple : Dans un programme gérant un carnet d'adresses, l'utilisateur doit, d'une exécution à l'autre, retrouver son carnet à jour, avec les modifications apportées lors de la dernière exécution de programme. Les données du carnet d'adresse ne peuvent donc être inclues dans le code source de l'algorithme, et encore moins être entrées au clavier à chaque nouvelle exécution.

Pour combler ce manque, utiliser les fichiers afin de stocker les données de manière permanente, entre les différentes exécutions du programme.

1. **Définition d'un fichier :** Un fichier est un regroupement logique de données mémorisées sur un support permanent afin de permettre une réutilisation ultérieure des informations qu'il contient.

2. Structuration des données dans un fichier

L'organisation des données dans un fichier correspond à la manière selon laquelle les données sont organisées à l'intérieur du fichier.

2.1. Fichiers NON structurés : Cette catégorie regroupe tous les fichiers constitués d'un texte dont la structure n'est pas déterminée (documents texte, codes sources, etc.). Donc, on ne trouve pas de notion de données élémentaires et le fichier doit être pris dans son ensemble.

2.2. Fichiers structurés : Les fichiers structurés sont composés d'un ensemble de données élémentaires identifiables de manière précise au sein du fichier.

a) **Structure déterminée par un balisage des données élémentaire :** Dans cette catégorie, on trouvera les fichiers de type XML, HTML, etc. Chaque donnée élémentaire est encadrée par un balisage.
b) **Structure organisée en enregistrements :** Un enregistrement est un bloc de données élémentaires qui décrit une entité. Par exemple, au sein d'un fichier « Clients », un enregistrement correspond aux données relatives à un client.

Un champ est l'une des données élémentaires d'un enregistrement : le numéro de client est l'un des champs d'un enregistrement du fichier client (les autres champs seront par exemple, le nom, le prénom, la date de naissance, l'adresse, la raison sociale, etc.).

Il existe deux manières d'isoler les champs dans un enregistrement :

1. A l'aide d'un caractère séparateur de champs : les données sont codées en ASCII (ou autre format de caractères) ; un enregistrement correspond à une ligne d'un fichier texte.
2. A l'aide d'une définition précise de l'enregistrement et de ses champs grâce à un type structuré (enregistrement).

Résumé sur les caractéristiques des fichiers

<table>
<tr><td>Organisation</td><td>Non structuré</td><td colspan="3">Structuré en regroupement de données élémentaire</td></tr>
<tr><td rowspan="2">Délimitation des champs</td><td></td><td>Balise des données élémentaires</td><td colspan="2">Enregistrements</td></tr>
<tr><td></td><td></td><td>Séparateurs de champs</td><td>Structure</td></tr>
<tr><td>Formats</td><td>.cpp, .docx</td><td>.xml, .html</td><td>Fichiers de données au format texte : .csv</td><td>Fichiers de données au format binaire : .bmp</td></tr>
</table>

4. Les fichiers de données structurés en enregistrements

A. Détermination des champs de données

	Nombre de champs	Longueur des champs	Délimitation des champs	commentaire
Enregistrements avec séparateur de champs	Fixe	Variable	Un caractère séparateur est défini pour l'ensemble du fichier	On parle de « fichier plat » ou fichier texte avec séparateur
Enregistrements avec type structuré	fixe	Fixe	Enregistrement sous forme de type structuré	Les données numériques sont codées en binaire et donc non lisibles

Exemple d'enregistrements d'un fichier texte avec caractère séparateur « ; » :

1 ;"Tim";"Burtley";"tb@free.fr"
2 ;"Max";"Ximum";"max.ximum@yahoo.fr"

Exemple d'enregistrements d'un fichier avec un type structuré :

000@Tim Burtley tb@free.fr 000çMax Ximum max.ximum@yahoo.fr

B. Organisation du placement des données

L'organisation des données dans un fichier détermine comment seront placés chacun des enregistrements.

1. Organisation séquentielle :

Dans des fichiers séquentiels, les enregistrements sont mémorisés consécutivement dans l'ordre de leur entrée et peuvent seulement être lus dans cet ordre. Si on a besoin d'un enregistrement précis dans un fichier séquentiel, il faut lire tous les enregistrements qui le précèdent, en commençant par le premier.

Il y a deux grandes variantes pour structurer les données au sein d'un fichier séquentiel : la délimitation et les champs de largeur fixe.

Reprenons le cas du carnet d'adresses, avec comme champs : le nom, le prénom, le téléphone et l'email. Les données, peuvent être organisées de deux manières :

Structure n°1

"Fonfec";"Sophie";0142156487;"fonfec@yahoo.fr"
"Zétofrais";"Mélanie";0456912347;"zétofrais@free.fr"
"Herbien";"Jean-Philippe";0289765194;"vantard@free.fr"

Structure n°2

Fonfec	Sophie	0142156487	fonfec@yahoo.fr
Zétofrais	Mélanie	0456912347	zétofrais@free.fr
Herbien	Jean-Philippe	0289765194	vantard@free.fr

- La structure n°1 est dite délimitée ; Elle utilise un caractère spécial, appelé caractère de délimitation, qui permet de repérer quand finit un champ et quand commence le suivant. Ce caractère de délimitation doit être strictement interdit à l'intérieur de chaque champ, sinon la structure devient proprement illisible.

- La structure n°2, elle, est dite à champs de largeur fixe. Il n'y a pas de caractère de délimitation, mais on sait que les x premiers caractères de chaque ligne stockent le nom, les y suivants le prénom, etc. Cela implique de ne pas saisir un renseignement plus long que le champ prévu pour l'accueillir.

2. Organisation calculée

Dans des fichiers à placement calculé, les enregistrements sont placés à une position précise du fichier. Cette position est :

- Soit donnée comme numéro d'ordre de l'enregistrement dans le fichier.
- Soit calculée selon un algorithme de placement appliqué à une clef (on parle de placement aléatoire).

3. Organisation indexée

Dans des fichiers à organisation indexée, au moins deux fichiers sont nécessaires pour chaque fichier géré :

- le premier fichier contient les données
- le second fichier contient une table d'index qui à chaque clef d'un enregistrement indique la position dans le fichier à laquelle se trouve l'enregistrement.

La clef correspond à l'un des champs de l'enregistrement permettant l'identification d'un enregistrement unique au sein du fichier.

C. Modes d'accès aux enregistrements d'un fichier

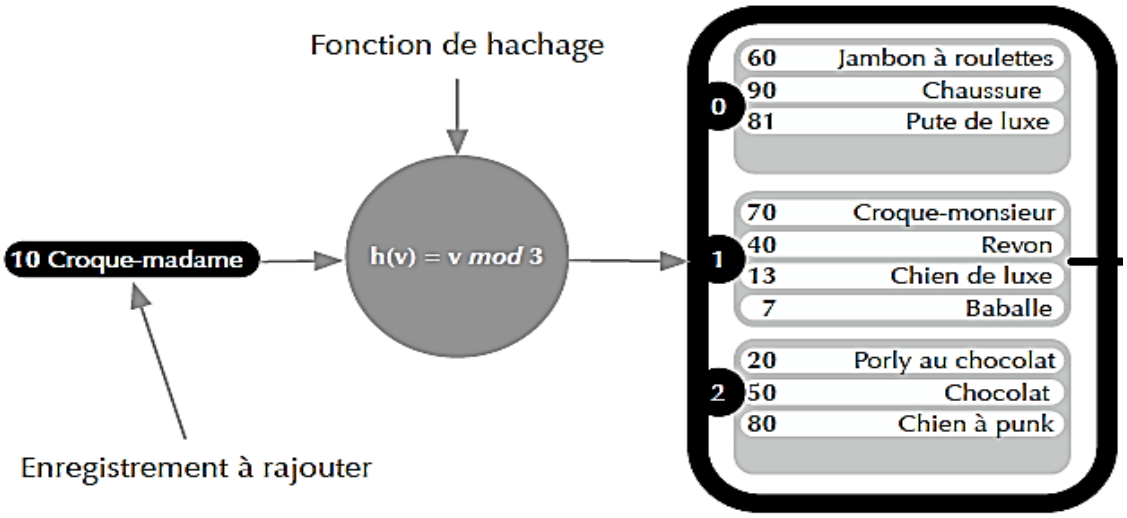
L'accès aux enregistrements d'un fichier est déterminé par l'organisation de ce fichier.

1. Accès séquentiel : L'accès séquentiel consiste à parcourir, dans l'ordre dans lequel ils sont stockés, les enregistrements d'un fichier. Pour accéder à un enregistrement, il faut avoir lu tous les enregistrements qui le précèdent. L'accès séquentiel est effectué dans un seul « sens » : on parcourt les enregistrements du début à la fin, sans retour en arrière.

2. Accès direct : L'accès direct permet de retrouver directement à un enregistrement ou à un bloc d'enregistrements dans le fichier, l'accès se fait :

- Soit grâce à un numéro d'ordre de placement ;
- Soit grâce à une clef.

Par exemple : avec une fonction hash-code du style h(v) = v mod B on peut reconnaître le bloc dans lequel on range ou on cherche un enregistrement. Si B = 3, alors on aura une fonction h(v) = v mod 3, et l'enregistrement de clef 10 sera mis dans le bloc 1 car : h(10) = 10 mod 3 = 1. Voir la figure suivante :



D. Modes d'ouverture d'un fichier

Le mode d'ouverture d'un fichier détermine les actions autorisées sur le fichier

Mode d'ouverture	**Actions autorisée**
Lecture	En mode lecture, seules seront autorisés l'accès aux données, sans modification possible du contenu
Ecriture	En mode écriture, le fichier est vidé de son contenu, et ne sera possible que l'écriture de nouveaux enregistrements
Ajout	En mode ajout, les données présentes dans le fichier seront préservées, et il sera possible d'ajouter de nouveaux enregistrements
Lecture/Ecriture	En mode lecture/écriture, les données présentes dans le fichier seront préservées, et il sera possible de modifier le contenu de certains enregistrements.

Résumé des modes d'organisation et d'accès et d'ouvertures

		Organisation - placement		
		Séquentielle	Calculée	Indexée
Accès	Séquentiel	Seul possible	Dans une boucle de calcul des clefs	Par parcours du fichier d'index
	Direct		Naturel	Naturel
Structure des enregistrements		Fichiers textes avec séparateurs ou structurés	Fichiers à type de donnée structuré	
Modes d'ouverture du fichier	Lecture	Oui	Oui	Oui
	Ecriture	Oui	Oui	Oui
	Ajout	Oui	Oui	Oui
	Modification	-	Oui	Oui

Algorithmique – fichiers séquentiels

Le langage algorithmique (et la plupart des langages de programmation) met à disposition du programmeur un ensemble d'instructions permettant la manipulation des fichiers.

A. Déclaration d'une variable de type fichier

var mon_fichier : fichier ;

B. Ouvrir un fichier séquentiel

Mon_fichier ← ouvrir(nom_du_fichier , mode_ouverture) ;

Avec mode_ouverture ϵ { lecture, écriture, ajout }

C. Lire un fichier

Lire_fichier(mon_fichier, liste_de_variables) ;

D. Tester la fin du fichier

fin_fichier(mon_ichier) ; retourne VRAI si la fin du fichier a été atteinte.

E. Ecrire dans un fichier

écrire_fichier(mon_fichier, liste de variables) ;

Les données sont ajoutées soit à partir du début du fichier soit à partir de la fin (selon le mode ouverture)

F. Fermer un fichier

fermer(mon_fichier) ;

G. Exemple de programme

```
Var f1, f2 : fichier ;
    num, nom, prenom, email : chaine ;
debut
        f1 ←ouvrir("fichier1.txt" ,lecture) ;
        f2 ← ouvrir("fichier2.txt" , écriture) ;
       lire_fichier(f1, num, nom, prenom, email) ;
       tant que (non fin_fichier(f1)) faire
       debut
              écrire_fichier(f2, num, nom, prenom, email) ;
              lire_fichier(f1, num, nom, prenom, email) ;
       fin tq
       fermer(f1) ;
```

```
        fermer(f2) ;
fin
```

Autre exemple

```
Enreg : enregistrement
        num : entier ;
        nom : chaine ;
        email : chaine ;
fin
var f3 : fichier ;
ligne : enreg ;
debut
        f3 ← ouvrir ("fichier3.dat" , écriture) ;
        écrire("entrez le numéro, le nom et l'adresse email")
        lire (ligne.num, ligne.nom, ligne.email) ;
        écrire_fichier (f3, ligne) ;
        fermer (f3) ;
fin
```