Synthèse des Fichiers

Laëtitia Monnier

Sommaire

Définition

Un fichier est un ensemble d'informations homogènes regroupées sous un même nom.

En algorithmique, un fichier est aussi un ensemble de données organisées en unités d'accès appelées « enregistrements » ou « articles » (tous de même type). Celui-ci est toujours enregistré sur un support externe à la mémoire centrale (disque dur, clé USB, disquette…). Ainsi, les informations sont dites « *non-volatiles* ».

Il existe 3 actions courantes afin de manipuler les fichiers :

- Création → définir le type du fichier ainsi que son emplacement physique.
- Consultation → exploiter les articles du fichier sans en modifier un seul.
- Mise à jour → ajouter, modifier ou supprimer des enregistrements ou des champs.

Le type Fichier

Pascal admet 3 classes → les fichiers typés, les fichiers non typés, les fichiers texte.

Remarque :

- Les fichiers typés et non typés sont des *fichiers binaires* → ils ne contiennent pas des informations directement lisibles par l'être humain.
- La taille maximale d'un fichier est *limitée* par l'espace disponible sur son support, et sa manipulation dans un programme se fait à l'aide d'un pointeur.

Déclaration des fichiers

Pour utiliser un ou des fichiers tout au long d'un programme ou dans une procédure, il faudra *l'identifier par une variable* dont le type est fonction de l'utilisation que l'on veut faire du fichier.

- ***Les fichiers typés (File of …) :***

  Ce sont des fichiers dans lesquels les données sont écrites telles qu'elles se présentent en mémoire (en binaire). Le plus grand avantage → obtient des *fichiers parfaitement formatés*, c'est-à-dire qu'on peut y lire et écrire directement des variables de type structuré qui contiennent plusieurs champs de données sans avoir à se soucier des divers champs qu'elles contiennent.

  Syntaxe n°1 :

  Type <identificateur_fichier> = fichier de <type_des_articles>

  Var f : <identificateur_fichier>

  Syntaxe n°2 :

  Var f : fichier de <type_des_articles>

  Exemple : Si on veut définir un fichier d'entier, on l'écrira comme cela → Var f : fichier de entier.

- ***Les fichiers textes (Text) :***

  Les fichiers textes ne peuvent seulement contenir que des informations de *type texte* dans lesquels on peut écrire et lire ligne par ligne ou à la file avec les procédures Write(Ln) et Read(Ln).

  Syntaxe n°1 :

  Type <identificateur_fichier> = texte

  Var f : <identificateur_fichier>

  Syntaxe n°2 : Var f : texte

- *Les fichiers non-typés (File) :*

  Les fichiers non-typés sont très peu utilisé. Ils permettent d'introduire dans un même fichier des articles appartenant à des types différents.

Syntaxe n°1 :

Type <identificateur_fichier> = fichier

    Var f : <identificateur_fichier>

Syntaxe n°2 : Var f : fichier

Utilisation des fichiers

L'assignation :

Avant de travailler sur un fichier, il faut le déclarer en lui affectant une variable qui servira à désigner le fichier tout au long du programme ou de la procédure dans laquelle il est utilisé.

Syntaxe : ASSIGNER (f, 'chemin_dacces_F')

Remarque :

- Valable pour tous les types de fichiers (binaires ou texte).
- On peut utiliser une même variable f pour effectuer des modifications sur 2 fichiers physiques F et G de même type.

- L'ouverture :
  - OUVRIR(f) → si F est un *fichier existant* déjà sur disque (ouverture en lecture).
  - REECRIRE(f) → si on veut *créer* un fichier F qui *n'existe pas* encore sur disque, ou bien qui existe déjà mais dont on veut *écraser* tout le contenu (ouverture en écriture).
  - AJOUTER(f) → si on veut ouvrir un *fichier texte F* en pouvant y ajouter des données (*uniquement en fin de fichier*).

Remarques :

- OUVRIR() rend F consultable et positionne le pointeur en *début de fichier*.
- Sur un fichier texte, la primitive OUVRIR() *n'autorise que la lecture*.
- Dans le cas où F n'existe pas sur le disque, REECRIRE() → crée un fichier dont le nom et le chemin d'accès sont ceux précisés pendant l'assignation.
- Possibilité d'ouvre un fichier qui ne contient aucune donnée. Le début de fichier sera alors la fin de fichier. La variable booléenne Eof(f) permet à tout moment de vérifier si le pointeur s'y trouve déjà.

- La lecture et la mise à jour :

Après l'ouverture du fichier, plusieurs possibilités sont offertes.

- Les fichiers typés :

La lecture et la mise à jour de F dans l'algorithme nécessitent la présence d'une variable p du type de ces articles.

Syntaxes :

Lire(f, p) → lit un article de F et insère les données qu'il contient dans la variable p.

Ecrire(f, p) → insère dans F un article ayant les données contenues dans la variable p. Il y a écrasement si un autre article occupait déjà le même emplacement dans le fichier.

Note : après la lecture/écriture d'un article, le pointeur se positionne immédiatement sur l'article suivant. S'il n'y en a plus, la position du pointeur devient alors la fin de fichier.

- Les fichiers textes :

Ici, notre variable p sera obligatoirement une chaîne de caractères.

Syntaxes :

Lire(f, p) → lit une ligne de F et l'insère dans la variable p.

Ecrire(f, p) → insère **à la fin** de F une ligne de texte dont la valeur est contenue dans p.

Remarque :

- Lire() n'est possible que si on a *ouvert* le fichier texte en lecture avec *OUVRIR()*. Après la lecture, le pointeur se positionne immédiatement sur la ligne suivante. S'il n'y en a plus, la position du pointeur devient alors la fin de fichier.
- Ecrire() n'est possible que si on a *ouvert* le fichier texte en écriture avec *AJOUTER()*.

La fermeture :

Il est impératif de *fermer* son fichier afin d'éviter les quelconques erreurs d'entrées/sorties !

Syntaxe : FERMER(f)

Exemples d'utilisation

Les fichiers textes :

Voici un algorithme qui crée un fichier texte lors de la première utilisation du programme par un utilisateur. Une fois le fichier crée, il pourra entrer les informations demandées et en ajoutés.

```
ALGO : FichierTexte
BUT : Crée un fichier si celui est inexistant, et rajoute des informations par la suite.
ENTREE : Les informations necessaires.
SORTIE : Affichage du fichier.

TYPE
    client = ENREGISTREMENT
        nom : CHAINE
        prenom : CHAINE
        adresse : CHAINE
        cp : CHAINE
        ville : CHAINE
FINENREGISTREMENT

PROCEDURE renseignement(var XClient : client; var f : Texte)
debut
    ECRIRE('Entrez votre nom.')
    LIRE(XClient.nom)
    XClient.nom <- upCase(XClient.nom)
    ECRIRE(f,XClient.nom)
    ECRIRE('Entrez votre prenom.')
    LIRE(XClient.prenom)
    XClient.prenom <- upCase(XClient.prenom)
    ECRIRE(f,XClient.prenom)
    ECRIRE('Entrez votre adresse.')
    LIRE(XClient.adresse)
    ECRIRE(f,XClient.adresse)
    ECRIRE('Entrez votre code postal.')
    LIRE(XClient.cp)
    XClient.cp <- copy(XClient.cp,1,5)
    ECRIRE(f,XClient.cp)
    ECRIRE('Entrez le nom de votre ville.')
    LIRE(XClient.ville)
    XClient.ville <- upCase(XClient.ville)
    ECRIRE(f,XClient.ville)
    ECRIRE(f,' ')
FINPROCEDURE
```

```
38
39  PROCEDURE modifier(var XClient : client; var f : Texte)
40  debut
41      AJOUTER(f)
42      SI IOResult <> 0 ALORS
43          REECRIRE(f)
44      FINSI
45      renseignement(XClient, f)
46      FERMER(f)
47  FINPROCEDURE
48
49  PROCEDURE menu()
50  debut
51      ECRIRE('-- Fiche de Renseignement Client --')
52      ECRIRE('1 : Inscription.')
53      ECRIRE('2 : Afficher la liste renseignement client.')
54      ECRIRE('0 : Sortir !')
55  FINPROCEDURE
56
57  PROCEDURE affichage(var f : Texte)
58  VAR
59      s : CHAINE
60  debut
61      Reset(f)
62      SI PAS OUVRIR(f) ALORS
63          ECRIRE('Une erreur s''est produite avec le fichier.')
64      SINON
65          TANTQUE NON Eof(f) FAIRE
66              LIRE(f,s)
67              ECRIRE(s)
68          FINTANTQUE
69          FERMER(f)
70      FINSI
71  FINPROCEDURE
72
73  PROCEDURE effaceEcran()
74  VAR
75      compteur : ENTIER
76  debut
77      GOTOXY(1, 5)
78      POUR compteur DE 1 A 12 FAIRE
79          ECRIRE('                              ')
80      GOTOXY(1, 5)
81  FINPROCEDURE
```

```
83  Programme Principal
84
85  VAR
86      XClient : client
87      f : Texte
88      choix : ENTIER
89
90  DEBUT
91      Ajoutez un clrscr ici.
92      ASSIGNER(f, 'FicheClient.txt')
93      menu()
94      REPETER
95          LIRE(choix)
96          SI choix = 1 ALORS
97              modifier(XClient, f)
98          FINSI
99          effaceEcran()
100     JUSQUA (choix = 2) OR (choix = 0)
101     SI choix = 2 ALORS
102         affichage(f)
103     SINON SI choix = 0 ALORS
104         ECRIRE('Exit')
105     FINSI
106 FIN
```

- Les fichiers typés :

Voici un algorithme qui demande à l'utilisateur d'entrer des scores qui se stockeront dans un fichier. Les 3 premiers scores du fichier s'affichent ensuite à l'écran.

```
CONST
    MAX = 3

VAR
    f : file of integer
    score, compteur : ENTIER
    resultat : tableau [1..MAX] de ENTIER

DEBUT
    ASSIGNER(f, 'score.txt')
    OUVRIR(f)
    ECRIRE('Entrez votre score / 000 pour sortir')
    REPETER
        LIRE(score)
        ECRIRE(f, score)
        GOTOXY(1,2)
        ECRIRE('               ')
        GOTOXY(1,2)
    JUSQU'A score = 000
    PositionCourante(f,0)
    LectureBloc(f, resultat, MAX)
    POUR compteur DE 1 A MAX FAIRE
        ECRIRE(resultat[compteur])
    FERMER(f)
FIN
```