Université A.Mira – Bejaia
Faculté de Technologie
Département de Technologie
1ère Année ST – Cycle Licence

Année Universitaire : 2022-2023

Semestre 2. Module : Informatique II

Chapitre III

Enregistrements & Fichiers

Sommaire

Chapitre III : Enregistrement et Fichiers

III.1. Notion d'Enregistrement

Un enregistrement est une structure (complexe) de données permettant de représenter un objet, réel ou abstrait, sous forme d'un ensemble de données (dites champs) hétérogènes, c'est-à-dire de types différents.

On peut considérer un enregistrement comme étant un tableau à une dimension, où chaque case (composante) du tableau représente un champs dans l'enregistrement. La différence réside dans :

– Une case de tableau est accessible à travers un indice entier, par contre un champs est repéré par un nom (identificateur) ;

– Les cases du tableau sont toutes du même type (structure homogène), par contre, les champs de l'enregistrement ne sont pas obligatoirement du même type (structure hétérogène).

Exemple

Pour représenter les informations liées à un produit quelconque dans une même structure, on peut utiliser un enregistrement constitué de champs suivants :

- La désignation du produit
- La référence du produit
- Sa quantité en stock
- Son prix unitaire
- Etc…

Les informations précédentes (les champs de l'enregistrement) sont de types différents, elles ne peuvent être décrites par un tableau (dont les cases sont du même type). Les types de ces informations sont respectivement :

- Chaîne de caractères
- Chaîne de caractères
- Entier
- Réel

III.2. Déclaration d'un Enregistrement

La syntaxe générale pour déclarer un enregistrement est comme suit :

En Algorithme

```
Type <id_Enreg> = Enregistrement
    <id_ch1> : <type1>
    <id_ch1> : <type2>
    ...
    <id_chN> : <typeN>
Fin
```

En Pascal

```
Type <id_Enreg> = RECORD
    <id_ch1> : <type1>
    <id_ch1> : <type2>
    ...
    <id_chN> : <typeN>
End;
```

Où <id_ch1>, <id_ch2>, …, <id_chN> sont les identificateurs des champs et <type1>, <type2>, …, <typeN> leurs types respectifs.

Pour l'exemple du Produit précédent, la déclaration sera comme suit :

En Algorithme

```
Type Produit = Enregistrement
    Designation : Chaîne
    Reference : Chaîne
    Quantite : Entier
    Prix_U : Réel
Fin
```

En Pascal

```
Type Produit = RECORD
    Designation : String[30];
    Reference : String[13];
    Quantite : Integer;
    Prix_U : Real;
End;
```

III.3. Accès aux champs d'un enregistrement

Pour référencer ou accéder à un champ quelconque d'un enregistrement, on utilise la notation pointée. La notation pointée utilise la variable enregistrement (identificateur) suivie d'un point puis du nom du champ auquel on veuille accéder. Soit :

```
<Variable_Enregistrement>.<Nom du champ>
```

Exemple :

Un nombre complexe peut être décrit par un enregistrement . Soit :

En Algorithme

```
Type Complexe = Enregistrement
   X, Y : réel
Fin;
```

En Pascal

```
Type Complexe = RECORD
   X, Y : real;
End;
```

Un nombre complexe peut alors être décrit (représenté) par un enregistrement de deux champs : X est la partie réelle et Y est la partie imaginaire. Par exemple :

Algorithme	#	Programme PASCAL

Algorithme Exemple01;
  Type
  Complexe = Enregistrement
  x, y : réel;
  Fin;
  Variables
  nbr : Complexe;
Début
  nbr.x ← 10;
  nbr.y ← -2;
 	
  écrire ('Nbr = ', nbr.x, ' + ', nbr.y, ' * i ') ;
Fin.
	01
	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
		Program Exemple01;
		  Type
		  Complexe = Record
		  x , y : real;
		  End;
		  Var
		  nbr : Complexe;
		Begin
		  nbr.x := 10 ;
		  nbr.y := -2;
		
		  write ('Nbr = ', nbr.x:0:2, ' + ', nbr.y:0:2, ' * i ') ;
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/8Ti1NOvF7 Lien de l'explication :		

Ce qui correspond au nombre complexe *Nbr = 10 - 2i*

III.4. L'instruction WITH (Avec)

L'instruction PASCAL **WITH**, permet d'éviter les répétitions de la variable enregistrement lors de l'accès à ses différents champs. Sa syntaxe générale est :

En Algorithme

```
Avec <var_enreg> Faire
   <instructions_champs_enreg>;
Fin-Avec;
```

En Pascal

```
With Complexe Do
Begin
  <instructions_champs_enreg>;
End;
```

Dans le cas des deux affectations précédentes, on peut écrire :

Algorithme	#	Programme PASCAL

Algorithme Exemple02;
  Type
  Complexe = Enregistrement
  x, y : réel;
	01
	02
	03
	04
		Program Exemple02;
		  Type
		  Complexe = Record
		  x , y : real;

Fin;
Variables
nbr : Complexe;
Début
Avec nbr faire
x ← 10;
y ← -2;
 	
écrire ('Nbr = ', nbr.x, ' + ', nbr.y, ' * i ') ;
Fin-Avec;
Fin.
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
		End;
		Var
		nbr : Complexe;
		Begin
		With nbr do
		begin
		nbr.x := 10 ;
		nbr.y := -2;
		write ('Nbr = ', nbr.x:0:2, ' + ', nbr.y:0:2, ' * i ') ;
		end;
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/8xXTPuWhU
Lien de l'explication :

Avec l'instruction **WITH**, on évite de répéter la variable d'enregistrement pour chaque champs. Un deuxième exemple, avec le type enregistrement Produit :

Algorithme	#	Programme PASCAL

Algorithme Exemple03;
Type
Produit = Enregistrement
Designation : Chaîne[40];
Reference : Chaîne[20];
Quantite : entier;
Prix_U : réel;
Fin;
Variables
p : Produit;
Début
Avec p faire
Designation ← 'Ordinateur';
Reference ← 'HP-Z00365';
Quantite ← 6;
Prix_U ← 57000.00;
 	
écrire ('Produit : ') ;
écrire(' Désignation : ', Designation);
écrire(' Référence : ', Reference);
écrire(' Quantité : ', Quantite);
écrire(' Prix Unitaire : ', Prix_U);
Fin-Avec;
Fin.
	01
	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
	16
	17
	18
	19
	20
	21
	22
	23
	24
		Program Exemple03;
		Type
		Produit = Record
		Designation : String[40];
		Reference : String[20];
		Quantite : integer;
		Prix_U : real;
		End;
		Var
		p: Produit ;
		Begin
		With p do begin
		p.Designation := 'Ordinateur';
		Reference := 'HP-Z00365';
		Quantite := 6;
		Prix_U := 57000.00;
		
		writeln ('Produit : ') ;
		writeln(' Désignation : ', Designation);
		writeln(' Référence : ', Reference);
		writeln(' Quantité : ', Quantite);
		writeln(' Prix Unitaire : ', Prix_U:0:2);
		end;
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/rFBTt7UoX
Lien de l'explication :

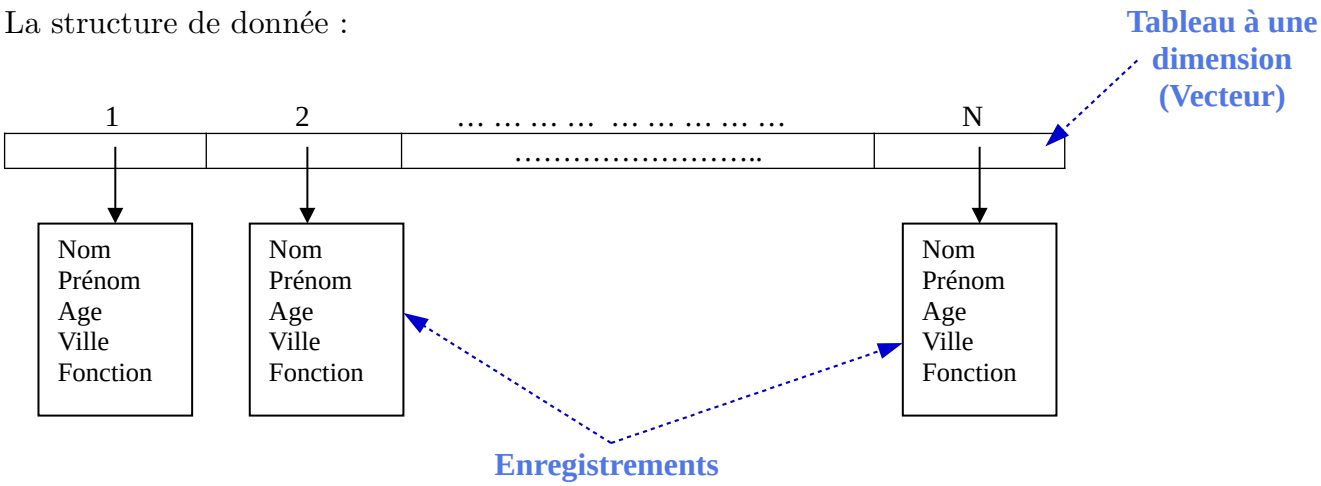
III.5- Exemple : Tableau d'enregistrement

Considérons des données comportant des individus décrits par les informations suivantes :

- Nom
- Prénom
- Age
- Ville
- Fonction

On veut organiser (Structurer) ces informations de telle sorte qu'on puisse sélectionner les individus ayant un certain âge, ou habitant une certaine ville ou ayant un fonction donnée etc…(Gérer et accéder à ces données). Proposer une structure de donnée.

La structure de donnée :



a) Déclaration et initialisation de la structure

On utilise un tableau d'enregistrements pour représenter cette liste d'individus :

Algorithme	#	Programme PASCAL
Algorithme Exemple04_Donnes_Init;	01	Program Exemple04_Donnes_Init;
Type	02	Type
Inidividu = Enregistrement	03	Individu = Record
Nom, Prenom : Chaîne[30];	04	Nom, Prenom : String[40];
Age : entier;	05	Age : integer;
Ville : Chaîne[40] ;	06	Ville : String[40];
Fonction : Chaîne[30] ;	07	Fonction : String[30];
Fin;	08	End;
Variables	09	Var
T : Tableau [1..10000] de Individu;	10	T : Array [1..10000] of Individu;
i, n : entier ;	11	i, n : integer ;

```
Début                                                   |  Begin
    Écrire('Donner le nombre d''individus : ');         |      write('Donner le nombre d''individus : ');
    Lire(n) ;                                           |      readln(n) ;
    Écrire('Donner les informations des individus : '); |      writeln('Donner les informations des individus : ');
    Pour i ← 1 à n faire                                |      for i := 1 to n do
      Avec T[i] faire                                   |        with T[i] do begin
        Lire(nom) ;                                     |          readln(nom) ;
        Lire(prenom) ;                                  |          readln(prenom) ;
        Lire(age) ;                                     |          readln(age) ;
        Lire(ville) ;                                   |          readln(ville) ;
        Lire(fonction) ;                                |          readln(fonction) ;
      Fin-Avec;                                         |        End;
    Fin-Pour;                                           |
                                                        |
    Écrire('Affichage des données : ');                 |      Writeln('Affichage des données : ');
    Pour i ← 1 à n faire                                |      for i := 1 to n do begin
      écrire('Individu N° ', i, ' :  ');                |        writeln('Individu N° ', i, ' :  ');
      Avec T[i] faire                                   |        with T[i] do begin
        écrire(nom) ;                                   |          write(nom) ;
        écrire(prenom) ;                                |          write(prenom) ;
        écrire(age) ;                                   |          write(age) ;
        écrire(ville) ;                                 |          write(ville) ;
        écrire(fonction) ;                              |          write(fonction) ;
      Fin-Avec;                                         |        end;
    Fin-Pour;                                           |      end ;
Fin.                                                    |  End.
```

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/G2COKNGSK
Lien de l'explication :

b) Afficher les éléments vérifiant des critères

Dans cet exemple, nous voulons modifier le programme précédent pour afficher uniquement les individus qui ont au moins 20 ans et au plus 35 ans, et qui habitent la ville de « El Kseur ».

Algorithme	#	Programme PASCAL
Algorithme Exemple05_Donnes_Filtre;	01	Program Exemple05_Donnes_Filtre;
Type	02	Type
Inidividu = Enregistrement	03	Individu = Record
Nom, Prenom : Chaîne[30];	04	Nom, Prenom : String[40];
Age : entier;	05	Age : integer;
Ville : Chaîne[40] ;	06	Ville : String[40];
Fonction : Chaîne[30] ;	07	Fonction : String[30];
Fin;	08	End;

```
Variables
    T : Tableau [1..10000] de Individu;
    i, n : entier ;
Début
  Écrire('Donner le nombre d''individus : ');
  Lire(n) ;
  Écrire('Donner les informations des individus : ');
  Pour i ← 1 à n faire
    Avec T[i] faire
      Lire(nom) ;
      Lire(prenom) ;
      Lire(age) ;
      Lire(ville) ;
      Lire(fonction) ;
    Fin-Avec;
  Fin-Pour;

  Écrire('Affichage des données : ');
  Pour i ← 1 à n faire
    Avec T[i] faire
    Si (age>=20)ET(age<=35)ET(ville='El Kseur')
  alors
      écrire('Individu N° ', i, ' :  ');
      écrire(nom) ;
      écrire(prenom) ;
      écrire(age) ;
      écrire(ville) ;
      écrire(fonction) ;
    Fin-Si;
    Fin-Avec;
  Fin-Pour;
Fin.
```

```
Var
    T : Array [1..10000] of Individu;
    i, n : integer ;
Begin
  write('Donner le nombre d''individus : ');
  readln(n) ;
  writeln('Donner les informations des individus : ');
  for i := 1 to n do
    with T[i] do begin
      readln(nom) ;
      readln(prenom) ;
      readln(age) ;
      readln(ville) ;
      readln(fonction) ;
    End;


  Writeln('Affichage des données : ');
  for i := 1 to n do begin
    with T[i] do
    if  (age>=20)AND(age<=35)AND(ville='El Kseur') then
     begin
      writeln('Individu N° ', i, ' :  ');
      write(nom) ;
      write(prenom) ;
      write(age) ;
      write(ville) ;
      write(fonction) ;
    end;


End.
```

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/TIn5akEwZ
Lien de l'explication :

Remarque

Dans les deux exemple précédents, les données devront être introduites à chaque exécution du programme, puis qu'il sont sauvegarder uniquement dans la RAM. Ce qui n'est pas normal surtout avec de grandes quantités de données (par exemple, nous avons 50 individus). La question qui se pose ici, est la suivante : comment saisir les données une seule fois et les sauvegarder d'une façons permanente. La réponse à cette question est d'utiliser les ***FICHIERS*** *qui sera étudié dans la section suivant.*

III.6. Les Fichiers

Un fichier est une collection de données sauvegardée dans un support de stockage secondaire : Disque dur, flash-disk, CD, … . En langage de programmation, un fichier est une structure de données permettant de manipuler les données sauvegardées en dehors de la mémoire centrale. En langage PASCAL, nous pouvons manipuler 3 types de fichiers.

- Les Fichiers à 'accès direct' (*structurés*) de type **FILE OF**.
- Les Fichiers à accès séquentiel (*Textuel*) de type **TEXT**
- Les Fichiers 'sans type' (*Binaire*) de type **FILE**.

Dans ce qui suit, nous allons voir uniquement les fichiers à accès direct : FILE OF … Ce type de fichier est constitué de blocs homogènes (de même type) dits : cellules, pouvant être de type simple : entier, réel, caractères, … ou de type enregistrements.

En langage PASCAL, on peut déclarer un fichier, selon son type, par :

```
<Nom Fichier> : FILE OF  <TYPE> ;
<Nom Fichier> : TEXT ;
<Nom Fichier> : FILE ;
```

Comme nous l'avons mentionné, dans ce qui suit, nous nous intéressons uniquement aux fichiers de type **FILE OF**.

III.6.1. Éléments d'un fichier

Parmi les éléments qui sont liés au fichier, nous allons voir :

- *Nom externe (nom physique)*
- *Nom interne (nom logique)*
- *Tampon (Buffer)*
- *FDF : Fin De Fichier (EOF : End Of File)*
- *Indice (pointeur interne)*

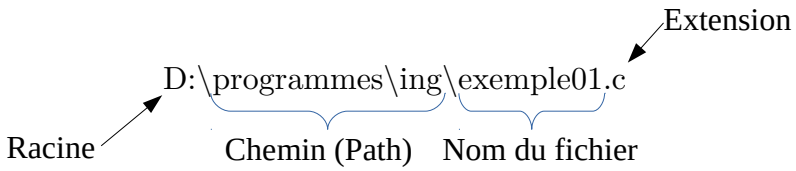
a) Nom externe (nom physique)

Nom externe est le nom avec lequel le fichier est identifié sur la mémoire secondaire, ce nom est composé de :

- identifiant du support (Racine)

- le chemin vers le fichier (path) : il y a le chemin absolue et le chemin relatif.
- nom du fichier proprement dit
- extension du fichier

Exemples :

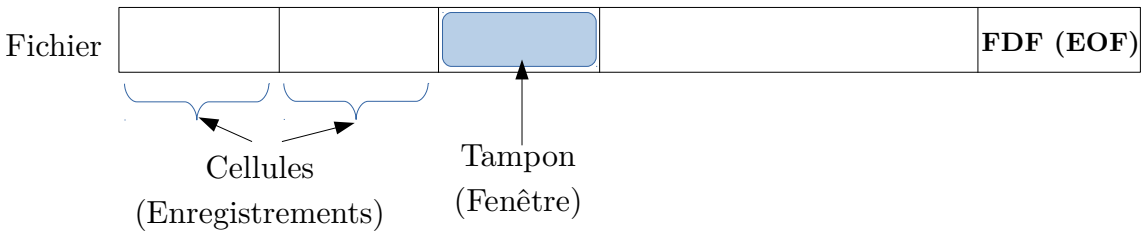


b) Nom interne (nom logique)

Nom interne est le nom avec lequel le fichier est identifié dans un programme (algorithme). C'est le nom logique qui doit être associé avec le nom externe (le nom physique). Plus précisément, un nom interne (logique) est un identificateur d'une variable pour accéder au données physiques du fichier.

c) Tampon (Buffer)

On appel tampon (buffer) d'un fichier, une zone de la mémoire central (RAM) pouvant contenir une cellule (éventuellement un enregistrement) de ce fichier. C'est à travers ce tampon qu'on rends les cases du fichier visible : pour cela, on appelle le tampon ***fenêtre*** à travers laquelle on *voit* le fichier.



d) FDF (EOF)

Chaque fichier se termine par une cellule spéciale dite FDF (pour Fin de Fichier), on anglais c'est : EOF (**End Of File**). Ça permet de savoir la fin de fichier lors du parcours des données du fichiers (voir les algorithmes et programmes) dans la suite du cours).

e) Indice (pointeur)

Pour chaque fichier, un indice (pointeur interne) caché, qui est géré automatiquement par le type fichier, est utilisé pour lire ou écrire la cellule pointée par cet indice.

III.6.2. Manipulation des fichiers de type FILE OF

Dans ce qui suit, nous allons voir comment :

- *Déclarer un fichier de type* ***File Of***
- *Créer un nouveau fichier*
- *Lire et Afficher le contenu d'un fichier*
- *Modifier le contenu d'un fichier*
- *Supprimer une cellule (Enregistrement) d'un fichier*

Pour illustrer les opération ci-dessus, nous prenons un exemple de fichier d'étudiants, où chaque étudiant est caractérisé par : *Matricule*, *Nom, Prénom, date de naissance, filière, niveau et sa moyenne.*

a) Déclaration de fichier

L'exemple ci-dessous, explique comment déclarer un fichier de type **File Of** contenant des enregistrements d'étudiant :

Algorithme	#	Programme PASCAL

Algorithme Exemple06_Fichier;
  Type
  Etudiant = Enregistrement
  Matricule : Chaîne[15] ;
  Nom, Prenom : Chaîne[30];
  Date_N : Chaîne[10];
  Filiere : Chaîne[50];
  Niveau : entier ;
  Moyenne : réel;
  Fin;
  Variables
  F : Fichier de Etudiant ;
Début
 	
Fin.
	01
	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
		Program Exemple06_Fichier;
		  Type
		  Etudiant = Record
		  Matricule : String[15] ;
		  Nom, Prenom : String[30];
		  Date_N : String[10];
		  Filiere : String[50];
		  Niveau : integer ;
		  Moyenne : real;
		  End;
		  Var
		  F : File of Etudiant;
		Begin
		
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/Kgdczxk8A Lien de l'explication :		

Remarque :

- Un fichier peut être vu comme un tableau à une dimension (vecteur) illimité (pratiquement illimité) sauvegardé sur une mémoire secondaire

- Les cases (cellules) du fichier sont accessible par un pointeur invisible.

- Avec les fichiers structurés (**FILE OF**), on peut déclarer un fichier d'entier, fichier de réel, ..., comme suit :

Algorithme	#	Programme PASCAL

Algorithme Exemple07_Fichier;
  Variables
  F1 : Fichier de Réel ;
  F2 : Fichier d'entier ;
Début
 	
Fin.
	01
	02
	03
	04
	05
	06
	07
		Program Exemple07_Fichier;
		  Var
		  F1 : File of Real;
		  F2 : File of Integer;
		Begin
		
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/Rdra-9EXt Lien de l'explication :		

b) Création d'un nouveau fichier

Pour créer un nouveau fichier, on doit :

1- Déclarer la structure de fichier : on continue sur l'exemple 6 (fichier d'étudiants)

2- Assigner le nom logique du fichier avec le nom physique (chemin et nom complets du fichier)

3- Ouvrir le fichier en écriture : Réécrire / Rewrite

4- écrire les données dans un fichier

5- fermer le fichier

L'exemple illustre les étapes ci-dessus pour créer un nouveau fichier :

Algorithme	#	Programme PASCAL

Algorithme Exemple08_Fichier_Creer;
  Type
  Etudiant = Enregistrement
  Matricule : Chaîne[15] ;
  Nom, Prenom : Chaîne[30];
  Date_N : Chaîne[10];
	01
	02
	03
	04
	05
	06
		Program Exemple08_Fichier_Creer;
		  Type
		  Etudiant = Record
		  Matricule : String[15];
		  Nom, Prenom : String[30];
		  Date_N : String[10];

```
                Filiere : Chaîne[50];
                Niveau : entier ;
                Moyenne : réel;
            Fin;
  Variables
      F : Fichier de Etudiant ;
      N, i : entier ;
      E : Etudiant ;
Début
    Assigner (F, 'etudiants.dat ') ;
     Réécrire(F) ;//Ouvrir le fichier en écriture
    écrire ('Données le nombre d'étudiants : ') ;
    Lire(N);
    écrire ('Données les informations : ') ;
    Pour i←1 à N faire
        Avec e faire
           Lire(Matricule, Nom, Prenom) ;
           Lire(Date_N);
           Lire(Filiere) ;
           Lire(Niveau) ;
           Lire(Moyenne);
        Fin-Avec;
        écrire(F, e);//Ajouter e au fichier F
    Fin-Pour;

    Fermer(F) ;
Fin.
```

```
                Filiere : String[50];
                Niveau : integer ;
                Moyenne : real;
            End;
  Var
      F : File of Etudiant;
      N, i : integer ;
      E : Etudiant ;
Begin
    Assign(F, 'etudiants.dat ') ;
    Rewrite(F);//Ouvrir le fichier en écriture
    écrire ('Données le nombre d'étudiants : ') ;
    Lire(N);
    écrire ('Données les informations : ') ;
    Pour i←1 à N do begin
        with e do begin
           Readln(Matricule, Nom, Prenom) ;
           Readln(Date_N);
           Readln(Filiere) ;
           Readln(Niveau) ;
           Readln(Moyenne);
        end;
        write(F, e);//Ajouter e au fichier F
    end;

    Close(F) ;
End.
```

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/Cegk4OtIM
Lien de l'explication :

Remarques

- L'instruction Réécrire(F); / Rewrite(F); permet de créer un nouveau fichier si le fichier n'existe pas, sinon (le fichier existe), cet instruction écrasera le fichier existant : ***IL FAUT FAIRE ATTENTION POUR NE PAR PERDRE DES DONNÉES.***

- L'instruction Écrire(F, e), en mode écriture, permet d'ajouter l'enregistrement e à la fin du fichier F.

- À la fin du programme, il ne faut pas oublier de fermer le fichier F : Fermer(F) ;/ Close(F);

c) Lire le contenu d'un fichier

Une fois un fichier est crée, on peut lire son contenu en l'ouvrant en mode lecture. L'exemple ci-dessous, montre comment lire le contenu du fichier ***etudiants.dat*** affiche son contenu sur l'écran :

Algorithme	#	Programme PASCAL

Algorithme Exemple09_Fichier_Afficher;
Type
Etudiant = Enregistrement
Matricule : Chaîne[15] ;
Nom, Prenom : Chaîne[30];
Date_N : Chaîne[10];
Filiere : Chaîne[50];
Niveau : entier ;
Moyenne : réel;
Fin;
Variables
F : Fichier de Etudiant ;
N, i : entier ;
E : Etudiant ;
Début
Assigner (F, 'etudiants.dat ') ;
Reouvrir(F) ;//Ouvrir le fichier en Lecture/éc.
 	
écrire ('Le contenu du fichier etudiants : ') ;
Tantque non(Eof(F)) faire
Lire(F, e); //Lire e à partir du fichier F
Avec e faire
Écrire(Matricule, Nom, Prenom, Date_N);
Écrire(Filiere, Niveau, Moyenne);
Fin-Avec;
Fin-Tantque;
 	
Fermer(F) ;
Fin.
	01
	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
		Program Exemple09_Fichier_Afficher;
		Type
		Etudiant = Record
		Matricule : String[15] ;
		Nom, Prenom : String[30];
		Date_N : String[10];
		Filiere : String[50];
		Niveau : integer ;
		Moyenne : real;
		End;
		Var
		F : File of Etudiant;
		i : integer ;
		E : Etudiant ;
		Begin
		Assign(F, 'etudiants.dat ') ;
		Reset(F);//Ouvrir le fichier en lecture/écriture
		
		écrire ('Le contenu du fichier etudiants : ') ;
		While not(Eof(f)) do begin
		Read(F, e); //Lire e à partir du fichier F
		With e do begin
		Writeln(Matricule, Nom, Prenom, Date_N);
		Writeln(Filiere, Niveau, Moyenne);
		end ;
		end ;
		
		Close(F) ;
		End.

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/7cLdsW5pC Lien du fichier *etudiants.dat* : https://www.dropbox.com/s/89nhm06qcdzdxg3/etudiants.dat?dl=0 Lien de l'explication :		

Remarques

- L'instruction Reourvir(F); / Reset(F) permet d'ouvrir fichier existant pour lire son

contenu ou le modifier. Si le fichier n'existe pas, il y aura une erreur pendant l'exécution : Runtime Error (Exception). Le pointeur de fichier est automatiquement au début du fichier (sur le premier enregistrement).

- L'instruction Lire(F, e) / Read(F, e) ;, en mode lecture, permet lire l'enregistrement en cours et passe le pointeur à l'enregistrement suivant. À la fin du fichier, en aura le résultat de : *eof(F)* est *True*.

d) Modifier le contenu d'un fichier

Modifier un fichier signifie modifier un ou plusieurs enregistrements, c'est-à-dire modifier quelques champs (ou tous les champs) de ces enregistrements. Dans la modification, on ne change pas le nombre d'enregistrements de fichiers, donc, il n y aura pas d'ajout ou de suppression dans la modification. Pour ces deux opérations, il seront faites dans les sections suivantes.

Pour modifier un ou plusieurs enregistrements, on doit colonnaire le filtre (conditions) pour sélectionner les enregistrements à modifier. Pour notre cas, on utilisera uniquement l'indice (le rang : numéro d'ordre) de l'enregistrement comme critère.

Algorithme	#	Programme PASCAL
Algorithme Exemple10_Fichier_Modifier;	01	Program Exemple10_Fichier_Modifier;
Type	02	Type
Etudiant = Enregistrement	03	Etudiant = Record
Matricule : Chaîne[15] ;	04	Matricule : String[15];
Nom, Prenom : Chaîne[30];	05	Nom, Prenom : String[30];
Date_N : Chaîne[10];	06	Date_N : String[10];
Filiere : Chaîne[50];	07	Filiere : String[50];
Niveau : entier ;	08	Niveau : integer ;
Moyenne : réel;	09	Moyenne : real;
Fin;	10	End;
Variables	11	Var
F : Fichier de Etudiant ;	12	F : File of Etudiant;
i, indicie : entier ;	13	i, indice : integer ;
E : Etudiant ;	14	E : Etudiant ;
Début	15	Begin
Assigner (F, 'etudiants.dat ') ;	16	Assign(F, 'etudiants.dat ') ;

Ré-ouvrir(F) ;//Ouvrir le fichier en lecture/écriture	17	Reset(F);//Ouvrir le fichier en lecture/écriture
écrire ('Données l''indice à modifier : ') ;	18	écrire ('Données l''indice de l''étudiants : ') ;
Lire(indicie);	19	Read(indicie);
écrire ('Données les informations : ') ;	20	écrire ('Données les informations : ') ;
Avec e faire	21	with e do begin
Lire(Matricule, Nom, Prenom) ;	22	Readln(Matricule, Nom, Prenom) ;
Lire(Date_N);	23	Readln(Date_N);
Lire(Filiere) ;	24	Readln(Filiere) ;
Lire(Niveau) ;	25	Readln(Niveau) ;
Lire(Moyenne);	26	Readln(Moyenne);
Fin-Avec;	27	end;
	28	
positionner(f, indice);//Positionner le pointeur	29	seek(f, indice); //Positionner le pointeur du fichier f
écrire(F, e);//Modifier l'enregistrement N° indice	30	write(F, e);//Modifier l'enregistrement N° indice
	31	
Fermer(F) ;	32	Close(F) ;
Fin.	33	End.
Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/bet1AL1PD Lien de l'explication :		

e) Insérer / ajouter un enregistrement

Dans l'exemple ci-dessous, nous allons voir comment réaliser l'insertion d'un nouveau enregistrement dans un fichier existant. En pascal, il n y a pas de procédure pour insérer directement dans le fichier initiale, dons ce cas nous allons suivre les étapes suivante :

1- Ouvrir le fichier *f1* concerné en mode lecture/écriture (modification)

2- Créer un nouveau fichier *f2* temporaire (ouvrir en mode écriture)

3- Introduire les informations du nouveau enregistrement *e* à insérer

4- Introduire l'*indice* (la position) de l'insertion de l'enregistrement (0 : au début, 1 en deuxième position, …, filesize(*f1*) en diernier)

5- Recopier les enregistrements de 0 jusqu'à (indice-1) de *f1* vers *f2*

6- ajouter enregistrement *e* au fichier *f2*

7- Recopier les enregistrements de *indice* jusqu'à filesize(f1)-1 de *f1* vers *f2*.

8- Supprimer le fichier *f1*

9- Renommer le fichier *f2* par le même nom physique de *f1*.

10- Fermer le fichier *f2*.

L'algorithme suivant, et le programme Pascal correspondant, montre comment insérer un nouveau enregistrement dans un fichier :

Algorithme	#	Programme PASCAL
Algorithme Exemple11_Fichier_Inserer;	01	Program Exemple11_Fichier_Inserer;
Type	02	Type
Etudiant = Enregistrement	03	Etudiant = Record
Matricule : Chaîne[15] ;	04	Matricule : String[15];
Nom, Prenom : Chaîne[30];	05	Nom, Prenom : String[30];
Date_N : Chaîne[10];	06	Date_N : String[10];
Filiere : Chaîne[50];	07	Filiere : String[50];
Niveau : entier ;	08	Niveau : integer ;
Moyenne : réel;	09	Moyenne : real;
Fin;	10	End;
Variables	11	Var
f1, f2 : Fichier de Etudiant ;	12	f1, f2 : File of Etudiant;
i, indicie : entier ;	13	i, indice : integer ;
e, e2 : Etudiant ;	14	e, e2: Etudiant ;
Début	15	Begin
Assigner (f1, 'etudiants.dat ') ;	16	Assign(f1, 'etudiants.dat ') ;
Ré-ouvrir(f1) ;//Ouvrir le fichier en lecture/écriture	17	Reset(f1);//Ouvrir le fichier en lecture/écriture
Assigner (f2, 'temp.dat ') ;	18	Assign(f2, 'temp.dat ') ;
Réécrire(f2) ;//Ouvrir le fichier en écriture	19	Rwrite(f2);//Ouvrir le fichier en écriture
écrire ('Données l''indice d''insertion : ') ;	20	écrire ('Données l''indice d''insertion : ') ;
Lire(indicie);	21	Read(indicie);
écrire ('Données les informations à insérer : ') ;	22	écrire ('Données les informations à insérer : ') ;
Avec e faire	23	with e do begin
Lire(Matricule, Nom, Prenom) ;	24	Readln(Matricule, Nom, Prenom) ;
Lire(Date_N);	25	Readln(Date_N);
Lire(Filiere) ;	26	Readln(Filiere) ;
Lire(Niveau) ;	27	Readln(Niveau) ;
Lire(Moyenne);	28	Readln(Moyenne);
Fin-Avec;	29	end;
	30	
//Copier les enregistrement de f1 à f2	31	//Copier les enregistrement de f1 à f2
Pour i←0 à (indice -1) faire	32	for i:=0 to (inice-1) do begin
Lire(F1, e2) ;	33	Read(F1, e2);
Écrire(F2, e2) ;	34	Write(F2, e2);
Fin-Pour;	35	end ;
	36	
écrire(F2, e); //Ajouter e au fichier f2	37	write(F2, e) ; //Ajouter e au fichier f2
	38	
	39	

```
//Copier les enregistrement de f1 à f2
Pour i←indice à (taille(f1)-1) faire
    Lire(F1, e2) ;
    Écrire(F2, e2) ;
Fin-Pour;

Fermer(f1) ;
Supprimer(f1) ;

Fermer(f2) ;
Renommer(f2, 'etudiants.dat ') ;
Fin.
```

```
40  //Copier les enregistrement de f1 à f2
41  for i:=indice to (filesize(f1)-1) do begin
42      Read(F1, e2);
43      Write(F2, e2);
44  end ;
45
46  Close(F1) ;
47  Erase(F1) ;
48
49  Close(F2) ;
50  Rename(F2, 'etudiants.dat ') ;
51  End.
```

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/IP4tC9isLb
Lien de l'explication :

Remarques

- Les procédures Supprimer(Fichier) et Renommer(Fichier, 'nom_fichier') doivent être appelés uniquement après la fermeture des fichiers.

f) Supprimer un enregistrement d'un fichier

La suppression d'un enregistrement à partir d'un fichier suit la même logique que l'insertion : utiliser un deuxième fichier temporaire. Voir l'exemple ci-dessous :

Algorithme	#	Programme PASCAL
Algorithme Exemple12_Fichier_Supprimer;	01	Program Exemple12_Fichier_Supprimer;
Type	02	Type
Etudiant = Enregistrement	03	Etudiant = Record
Matricule : Chaîne[15] ;	04	Matricule : String[15];
Nom, Prenom : Chaîne[30];	05	Nom, Prenom : String[30];
Date_N : Chaîne[10];	06	Date_N : String[10];
Filiere : Chaîne[50];	07	Filiere : String[50];
Niveau : entier ;	08	Niveau : integer ;
Moyenne : réel;	09	Moyenne : real;
Fin;	10	End;
Variables	11	Var
f1, f2 : Fichier de Etudiant ;	12	f1, f2 : File of Etudiant;
i, indicie : entier ;	13	i, indice : integer ;

```
        e : Etudiant ;
Début
    Assigner (f1, 'etudiants.dat ') ;
    Ré-ouvrir(f1) ;//Ouvrir le fichier en lecture/écriture
    Assigner (f2, 'temp.dat ') ;
    Réécrire(f2) ;//Ouvrir le fichier en écriture
    écrire ('Données l''indice d''insertion : ') ;
    Lire(indicie);

    //Copier les enregistrement de f1 à f2
    Pour i←0 à (indice -1) faire
        Lire(F1, e) ;
        Écrire(F2, e) ;
    Fin-Pour;

   //Copier les enregistrement de f1 à f2
    Pour i←(indice+1) à (taille(f1)-1) faire
        Lire(F1, e) ;
        Écrire(F2, e) ;
    Fin-Pour;

    Fermer(f1) ;
    Supprimer(f1) ;

    Fermer(f2) ;
    Renommer(f2, 'etudiants.dat ') ;
Fin.
```

```
14      e : Etudiant ;
15  Begin
16      Assign(f1, 'etudiants.dat ') ;
17      Reset(f1);//Ouvrir le fichier en lecture/écriture
18      Assign(f2, 'temp.dat ') ;
19      Rwrite(f2);//Ouvrir le fichier en écriture
20      écrire ('Données l''indice d''insertion : ') ;
21      Read(indicie);
22
23      //Copier les enregistrement de f1 à f2
24      for i:=0 to (inice-1) do begin
25          Read(F1, e);
26          Write(F2, e);
27      end ;
28
29      //Copier les enregistrement de f1 à f2
30      for i:=(indice+1) to (filesize(f1)-1) do begin
31          Read(F1, e);
32          Write(F2, e);
33      end ;
34
35      Close(F1) ;
36      Erase(F1) ;
37
38      Close(F2) ;
39      Rename(F2, 'etudiants.dat ') ;
40  End.
```

Le lien du programme PASCAL ci-dessus : https://onlinegdb.com/KyAVENW1l
Lien de l'explication :

III.6.3. Quelques fonctions/procédures sur les fichiers

Dans cette section nous énumérons quelques fonctions et procédures prédéfinies pour qui sont utilisé pour manipuler les fichiers structurés (fichiers FILE OF). Quelques fonctions / procédures sont utilisé dans les exemples précédent.

Le tableau suivant présente les principales fonctions / procédures sur les fichiers :

Fonction / Procédure	Signification
a) ASSIGN(Fichier, 'Nom_Fichier') ;	Procédure permettant d'associer le nom logique du fichier 'Fichier' au nom physique du fichier.
b) REWRITE (Fichier);	Procédure qui ouvre un fichier en mode écriture. Si le

	fichier n'existe pas, il sera créé, sinon, il sera écrasé (vider le fichier).
c) RESET (Fichier);	Procédure qui ouvre un fichier déjà existant et le prépare pour une lecture ou une écriture. Le pointeur de fichier est positionné sur le premier enregistrement du fichier. (ouverture en lecture/écriture).
d) READ(Fichier, v1, v2, …, vn) ;	Procédure de *lecture* des variables (tampons) v1, v2, …, vn à partir de fichier (lecture par fichier et non par clavier).
e) WRITE(Fichier, v1, v2, …, vn) ;	Procédure *d'écriture* des variables (tampons) v1, v2, …, vn dans le fichier (écriture sur le fichier et non sur l'écran).
f) FILESIZE(Fichier) ;	Fonction qui retourne le nombre d'enregistrements dans le fichier.
g) FILEPOS(Fichier) :	Fonction qui retourne la position actuelle du pointeur de fichier (entre 0 et Filesize(File)).
h) SEEK(Fichier, indice) ;	Procédure qui permet de positionner le pointeur de fichier sur l'indice (0 ≤ indice ≤ FileSize(File)-1).
i) EOF(Fichier) ;	Fonction booléenne qui retourne TRUE si on est à la fin du fichier, ou bien FALSE dans le cas contraire.
j) CLOSE(Fichier) ;	Fermer le fichier après son utilisation.
k) RENAME(Fichier, 'nom_fichier2');	Procédure pour renommer le fichier logique *Fichier* au nom : '*nom_fichier2*' . Le fichier doit être fermé avant RENAME.
l) ERASE(Fichier);	Procédure pour supprimer le fichier logique *Fichier*. Le fichier doit être fermé avant ERASE.

Bon Courage
&
Travaillez bien.