LES ENREGISTREMENTS ET LES FICHIERS

A – Les enregistrements

I- Introduction

1. Activité

On veut écrire un programme permettant de :

- Saisir les informations de vente de deux produits
- Déterminer le produit le plus bénéfique

Référence	Désignation	Quantité vendue	Prix de vente unitaire	Prix de vente total
Cf145				

2. Questions:

1. Quelle est la structure de donnée nécessaire ?

- Plusieurs tableaux

Référence				
Désignation				

.
.
.

2. Est-il possible d'utiliser une même structure pour toutes les informations d'un produit ?

- Si on veut établir une seule structure de données qui comporte à la fois les données numériques (quantité, prix unitaire, prix total) et les données alphanumériques (Référence, désignation) nous devons créer un nouveau type qui permet de les regrouper

⇨ **Les Enregistrements** ou **articles (Record** en Pascal**)**

II- Définition et déclaration

1. Définition :

Un enregistrement est un type de données défini par l'utilisateur et qui permet de rassembler un ensemble des éléments (ou champs) de type différents

Champ1 *Type1*	**Champ2** *Type2*	**Champ3** *Type3*	**Champ4** *Type4*	**Champ5** *Type5*

Un enregistrement

Référence *Chaîne*	**Désignation** *Chaîne*	**Quantité** *Entier*	**Prix de vente unitaire** *réel*	**Prix de vente total** *réel*

Un enregistrement produit

2 . Déclaration :

Méthode 1 :

En algorithmique

T. D. N.T

Type
Nom_type = **Enregistrement** Champ1 : type 1 … Champ n : type n **Fin** nom_type

T. D. O

Objet	Type/ Nature	Rôle
Nom_objet	Nom_type	Un enregistrement pour ...

En Pascal

```
TYPE
    Nom_type = Record
                 Champ1 : type 1 ;
                 …
                 Champ n : type n ;
               End ;
VAR
    Nom_objet : nom_type ;
```

Méthode 2 :

En algorithmique

T. D. N.T

Type
Reference = **Enregistrement** Part1 : "A".."Z" Part2 : entier Fin Reference Informations = **Enregistrement** Ref : reference Design : chaîne Qte : Entier PVU, PVT: Réel Fin Informations

T. D. O

Objet	Type/ Nature	Rôle
Produit	Informations	Enregistrement pour une fiche produit

En Pascal

```
TYPE
    Reference = Record
                  Part1: 'A'..'Z' ;
                  Part2: integer ;
                End ;
    Informations = Record
                     Ref : reference ;
                     Design : string ;
                     Qte : integer ;
                     PVU, PVT: Real ;
                   End ;
VAR
    Produit : Informations ;
```

❋ **Déclaration de type informations et de l'objet Produit**

Méthode 1 :

En algorithmique

T. D. N.T

Type
Informations = **Enregistrement** Ref, Design : chaîne Qte : Entier PVU, PVT: Réel Fin Informations

T. D. O

Objet	Type/ Nature	Rôle
Produit	Informations	Enregistrement pour une fiche produit

En Pascal

```
TYPE
    Informations = Record
                     Ref, Design : string ;
                     Qte : integer ;
                     PVU, PVT: Real ;
                   End;
VAR
    Produit : Informations ;
```

Méthode 2 :

En algorithmique

T. D. N.T

Type
Informations = **Enregistrement** Ref : **Enregistrement** Part1 : "A".."Z" Part2 : entier **Fin** ref Design : chaîne Qte : Entier PVU, PVT: Réel Fin Informations

T. D. O

Objet	Type/ Nature	Rôle
Produit	Informations	Enregistrement pour une fiche produit

En Pascal

```
TYPE
    Informations = Record
                     Ref : Record
                             Part1: 'A'..'Z' ;
                             Part2: integer ;
                           End ;
                     Design : string ;
                     Qte : integer ;
                     PVU, PVT: Real ;
                   End ;

VAR
    Produit : Informations ;
```

Remarques

- Les types des champs peuvent être prédéfinis ou définis par l'utilisateur
- Un champ a exactement les mêmes propriétés qu'une variable de même type
- Le champ d'une variable enregistrement peut être lui même un enregistrement

Exemple : déclarer le champ ref comme un enregistrement formé de 2 champs (part1 : Lettre majuscule, part2 : entier)

III- Utilisation des enregistrements

1. Affectation :

En algorithmique	*En Pascal*
Objet.champ ← valeur	Objet.champ := valeur ;

Remarque : Il est possible d'affecter une variable enregistrement dans une autre à condition qu'ils soient de même structure :

(e1 ← e2) ou (e2 ← e1)

Tous les champs de la variable enregistrement à affecter seront recopies dans les champs de l'autre.

2. Lecture :

En analyse	En algorithme	En Pascal
Objet.champ = donnée	Lire (objet.champ)	Readln (Objet.champ) ;

3. Ecriture :

En analyse & algorithme	*En Pascal*
Ecrire (objet.champ)	Writeln (Objet.champ) ;

Solution algorithmique de l'activite1:

Pascal

```
program produits;
uses wincrt;
type   Informations = record
                        ref, design : string;
                       qte : integer;
                       pvu, pvt : real;
   end;
var   prod1 , prod2 : informations;

procedure saisie ( var prod : informations);

begin
write ('reference =  ');   readln (prod.ref);
write ('designation =  ');   realdln(prod.design);
write ('Qte vendue =  ');  readln (prod.qte);
write ('prix de vente unitaire  =');
readln (prod.pvu);
end;
```

```
function test (prod1, prod2 : informations) : string;
begin
prod1.pvt := prod1.qte* prod1.pvu;
prod2.pvt := prod2.qte*prod2.pvu;
if prod1.pvt > prod2.pvt then test := 'produit 1 '
                          else
if prod1.pvt < prod2.pvt then test := 'produit 2 '
                          else  test := 'egalité';
end;

begin {PP}
writeln ('introduire les informations du produit 1');
saisie (prod1);
writeln ('introduire les informations du produit 2');
saisie (prod2);
writeln (test(prod1, prod2));
end.
```

4 . La structure avec…faire :

Pour simplifier l'écriture et éviter l'utilisation répétée de la notion **objet.champ**, nous pouvons utiliser l'instruction **Avec.. Faire**

En algorithmique

```
Avec  variable  Faire
      Action champ 1
      Action champ 2
      …
Fin Avec
```

En Pascal

```
With Variable  Do
  Begin
     Action champ1 ;
     Action champ2;
     …
  end;
```

Solution Avec.. Faire

En algorithmique

```
0.  Début  procedure saisie
                (var prod : informations)

1.  Avec  Prod  Faire
      Ecrire ("référence = "), Lire (ref)
      Ecrire ("désignation = "), Lire (design)
      Ecrire ("quantité vendue = "), Lire (Qte)
      Ecrire ("prix de vente unitaire= ")
      Lire (pvu)
    Fin Avec
2.  Fin saisie
```

En Pascal

```
procedure saisie(var prod : informations);

begin
     with prod do
       begin
       write('reference = '); readln(ref);
       write('designation = '); readln(design);
       write ('Qte vendue = '); readln(Qte);
       write ('prix de vente unitaire = ');
       readln (pvu);
       end;
end;
```

5. Vecteur d'enregistrements :

Activité : Reprenons l'activité1 pour n produits (avec 10 < n < 50)

Questions : Que peut être la structure de donnée nécessaire → tableau d'enregistrements

Déclaration

En algorithmique

T. D. N.T

Type
Informations = **Enregistrement** Ref, Design : chaîne Qte : Entier PVU, PVT: Réel Fin Informations Vect_info = tableau [1..50] d'informations

T. D. O

Objet	Type/ Nature	Rôle
Tab	Vect_info	Tableau de 50 enregistrements pour des fiches produits

En Pascal

```
TYPE
    Informations = Record
                     Ref, Design : string ;
                     Qte : integer ;
                     PVU, PVT: Real ;
                   End ;

    Vect_info =  array[1..50] of informations;

VAR
    Tab : Vect_info ;
```

B- Les fichiers :

I- Introduction :

Avec les structures précédemment utilisées, les données d'un programme seront perdues dès l'arrêt de l'exécution de ce programme. Dans certains cas la sauvegarde des données est nécessaire d'où on fait recours à une nouvelle structure ☞ **les fichiers**

Un fichier est un ensemble structuré des données de même type (réel, entier, caractère, chaîne, enregistrement …) enregistrées sur une mémoire auxiliaire.

II- Organisation des fichiers

L'organisation d'un fichier désigne le mode d'implémentation des informations et des enregistrements dans ce fichier et fournit les propriétés d'accès.

1. ***Organisation séquentielle*** : l'accès aux informations se fait en parcourant les enregistrements les uns après les autres
2. ***Organisation relative*** (dite aussi directe) : les enregistrements sont identifiés par un numéro d'ordre

III- Types d'accès

En informatique, nous distinguons deux types d'accès aux données d'un fichier :

- ***Accès séquentiel :*** pour accéder à l'information d'ordre n, on doit passer par les (n-1) informations précédentes
- ***Accès direct :*** On accède directement à l'information désirée, en précisant le numéro d'emplacement (le numéro d'ordre) de cette information.

Remarque : Tout fichier peut être utilisé avec l'un des deux types d'accès. Donc le choix de type d'accès dans un fichier ne concerne pas le fichier lui-même mais concerne la manière dont il va être traité par la machine (le choix de type d'accès se fait seulement dans le programme).

IV- Les Fichiers à accès séquentiel

1. Présentation

Un fichier est dit à accès séquentiel (ou fichier séquentiel) si l'accès à son n^{ième} information nécessite le passage par les (n-1) informations précédentes.

2. Déclaration

En algorithmique

T. D. N.T

Type
Nom_fichier = **Fichier** de type_composants

T. D. O

Objet	Type/ Nature	Rôle
Nom_logique	Nom_fichier	Fichier pour…

En Pascal

```
TYPE
        Nom_fichier = File of type_composants ;
VAR
        Nom_logique : nom_fichier ;
```

Remarques

1. Comme on a déjà dit un fichier doit être enregistré sur un support externe, donc ce fichier doit avoir un nom et de préférence une extension. Ce nom est appelé le **nom externe** (ou le **nom physique**)

2. Le nom de l'objet déclaré dans le tableau des objets comme nom de fichier est le **nom interne** du fichier (ou aussi le **nom logique**). C'est le nom utilisé dans les instructions du programme.

3. Les fichiers de données (data) ont une extension **.dat**, **.fch**

Activité :

On veut écrire un programme permettant de saisir et enregistrer les informations de plusieurs produits présentés dans l'activité 1 des enregistrements.
Présenter la déclaration en algorithme et en pascal de la structure de données nécessaire.

En algorithmique

T. D. N.T

Type
Informations = **Enregistrement** Ref, Design : chaîne Qte : Entier PVU, PVT: Réel Fin Informations Liste_prod = **Fichier** de informations

T. D. O

Objet	Type	Rôle
Fiches_prod	Liste_prod	fichier pour des fiches produits

En Pascal

```
TYPE
        Informations = Record
                        Ref, Design : string ;
                        Qte : integer ;
                        PVU, PVT: Real ;
                       End ;
        Liste_prod = File of informations;
VAR
        Fiches_prod : Liste_prod ;
```

3. Traitement sur les fichiers

Les fonctions et les procédures sur les fichiers :

Description	Syntaxe algorithmique	Syntaxe en Pascal
Avant d'utiliser un fichier il faut **associer** (Relier) son nom logique à son nom physique	**Associer (nom logique, nom physique)** *Ou bien* **Assigner (nom logique, nom physique)**	**Assign (nom logique, nom physique)**
Si le fichier existe : **Ouvrir** le fichier et effacer son contenu. Si le fichier n'existe pas : **Créer** le fichier.	**Recréer (nom_logique)**	**ReWrite (nom_logique) ;**
Ouvrir un fichier existant et repositionner ou remettre son pointeur à 0.	**Ouvrir (nom_logique)**	**ReSet (nom_logique) ;**
Ecrire ou **modifier** une valeur ou un enregistrement dans un fichier.	**Ecrire (nom_logique, variable)**	**Write (nom_logique, variable);**
Lire une valeur ou un enregistrement á partir d'un fichier.	**Lire (nom_logique, variable)**	**Read (nom_logique, variable)** **Remarque** *: read <> readln*
Fermer un fichier.	**Fermer (nom_logique)**	**Close (nom_logique) ;**
Fonction booléenne vérifiant si la **fin** du fichier a été atteinte.	**Fin_Fichier (nom logique)**	**EOF (nom logique)**

Remarque :

Les opérations de manipulation des fichiers sont sujettes à des nombreuses erreurs indépendantes du programme.

Tentative d'ouverture d'un fichier qui n'existe pas avec une des commandes suivantes :

- **Reset** (ouvrir un fichier)
- **Append** (ouvrir un fichier texte en mode ajout)
- **Rename** (renommer un fichier)
- **Erase** (supprimer un fichier)

Pour éviter cette erreur d'exécution :

- commencer par désactiver la détection automatique des erreurs par **{$I-}**
- réactiver la détection automatique des erreurs par **{$I+}** après ***la commande utilisée***
- faire recours à la fonction **IORESULT** qui retourne une valeur **<> 0** si le fichier n'existe pas

Exemple :

En Algorithme	En Pascal

Ecrire ("Entrer le nom du fichier : "),
lire (nom)
Associer fich_caractere,"c:\"+nom+".dat")
{$I-} {On désactive les erreurs d'E/S}
Ouvrir (fich_caractere)
{$I+} {On réactive les erreurs}
Si *ioresult* <> 0 **alors**
Recréer (Fich_caractere)
Saisie(Fich_caractere)
Fin si
	Write ('Entrer le nom du fichier : ');
	Readln (nom) ;
	Assign (fich_caractere,'c:\'+nom+'.dat');
	{$I-}
	ReSet (fich_caractere) ;
	{$I+}
	If *ioresult* <> 0 **Then**
	Begin
	ReWrite (Fich_caractere) ;
	Saisie(Fich_caractere) ;
	End ;

*On ouvre le fichier fich_caractere avec la commande **Ouvrir**, si le fichier n'existe pas l'ordinateur va détecter l'erreur sans se bloquer. La commande **ioresult** va contenir une valeur<>0, donc il faut dans ce cas créer le fichier avec la commande **Recréer**.*

Exercice : Ecrire un programme qui permet de remplir un fichier avec n caractères (n>0) et de déterminer la deuxième voyelle si elle existe.

En Algorithme	En Pascal

0. **Début** voyelle
1. Ecrire **("Entrer un nom de fichier :")**
lire **(nom)**
2. **Associer** (caractere, nom)
3. {$I-}
Ouvrir (caractere)
{$I+}
Si ioresult <>0 **alors**
Début
Recréer (caractere)
Proc saisie (caractere)
Fin
4. Proc recherche (caractere)
5. **Fermer** (caractere)
6. **Fin** voyelle
	BEGIN *{PP}*
	writeln **('Entrer un nom de fichier :') ;**
	readln **(nom) ;**
	Assign (caractere, nom) ;
	{$I-}
	ReSet (caractere) ;
	{$I+}
	If *ioresult* <>0 **Then**
	Begin
	Rewrite (caractere) ;
	Saisie (caractere) ;
	End ;
	Recherche (caractere) ;
	Close (caractere) ;
	End.

V- Les Fichiers à accès direct :

1. Présentation

Un fichier est dit à accès direct si on peut accéder directement à chacun de ses éléments.

2. Les fonctions et les procédures prédéfinies sur les fichiers

Les Fonctions et procédures d'accès direct sur les fichiers :

Rôle	Syntaxe en Algorithme	Syntaxe en Pascal
Accéder à un élément d'un fichier á accès direct.	**Pointer** (nom logique, numéro)	**Seek** (nom logique, numéro)
Retourner la taille d'un fichier à accès direct.	**Taille_fichier** (nom logique)	**Filesize** (nom logique)
Supprimer un fichier.	**Effacer** (nom logique)	**Erase** (nom logique)
Changer le nom d'un fichier.	**Renommer (**ancien nom logique, nouveau nom)	**Rename (**ancien nom logique, nouveau nom)
Tronquer le fichier, à la position courante du pointeur de fichier.	**Tronquer** (nom logique)	**Truncate (**nom logique)

Exercice :

Ecrire un programme en pascal permettant de :

- créer un fichier des élèves. Chaque élève est caractérise par :
  - nom
  - note de contrôle 1
  - note de contrôle 2
  - note de synthèse
  - moyenne

- ouvrir le fichier élève et constituer deux autres fichiers : le premier contiendra la liste des données correspondante aux élèves ayant une moyenne supérieure ou égale a 10 et le deuxième contiendra les données correspondante aux autres élèves.
- Déterminer et afficher le nombre des élèves qui ont une moyenne >= 10, leurs noms et leurs moyennes
- Déterminer et afficher le nombre des élèves qui ont une moyenne < 10, leurs noms et leurs moyennes
- Déterminer et afficher le numéro, le nom et la moyenne des élèves qui ont la moyenne la plus élevées.

```
program enlacement;
uses wincrt;
type
  eleve = record
          nom : string;
          n1,n2,ns,moy : real;
  end;
  feleve =file of eleve;
var
    f,f1,f2: feleve;     x : eleve;   nom1,nom2: string;

procedure saisie (var f : feleve ) ;
var i, n : integer ; rep: char;
begin
repeat
 with x do
  begin
  write (' Entrer le nom de l''eleve :'); readln (nom);
  write (' Entrer la note de devoir de controle n1 :');
  readln (n1);
  write (' Entrer la note de devoir de controle n2:');
  readln (n2);
  write (' Entrer la note de devoir de synthèse :');
  readln (ns);
  writeln;
  moy:= (n1+n2+ns*2)/4;
  write(f, x);
  end;
 repeat
   write (' voulez vous ajouter un autre élève : ');
   readln (rep);
 until (upcase (rep) in ['O','N']);
until upcase (rep)='N';
close (f);
end;

procedure creation ( var f : feleve );
var         nom : string;
begin
 write('introduire le nom : '); readln (nom);
assign (f, 'c:\'+nom+'.dat');
{$I-}
      reset(f);
{$I+}
if ioresult <> 0 then
                    begin
                       rewrite(f);
                       saisie(f);
                    end;
end;

procedure affiche (var ft : feleve ; v : string);
begin
reset (ft);
writeln ('le nombre des élèves qui ont une moyenne ', v, ' 10
est ', filesize (ft));

while not (eof (ft)) do
 begin
  read (ft, x);
  writeln (' l''eleve ', x.nom, ' a une moyenne = ', x.moy:5:2);
end;
end;

procedure eclat (var f, f1, f2: feleve);
var   p1, p2: integer;
begin
  reset (f); rewrite (f1); rewrite (f2);
  p1:=0 ; p2:=0;
  while not(eof(f)) do
     begin
       read (f, x);
       if x.moy>=10 then
                        begin
                             seek(f1,p1);
                             write(f1,x);
                              p1:=p1+1;
                         end
                         else
                         begin
                               seek(f2,p2);
                               write(f2,x);
                                p2:=p2+1;
                         end;
      end;
end;

procedure maximum(var f:feleve);
var   p : integer;   max : real;
begin
reset(f); read (f, x); max := x.moy;
while not(eof(f)) do
     begin
        read (f, x);
        if max < x.moy then
                              max := x.moy;
     end;
     reset (f);
 while not(eof(f)) do
 begin
   read (f, x);
   if max = x.moy then
    begin
    writeln;
    writeln (' l''eleve numéro : ', filepos (f),' de nom : ', x.nom,
                          ' a  une moyenne = ', x.moy:5:2);
     end;
     end;
end;

begin
   creation(f);
   write ('introduire le nom du premier fichier : ');
   readln (nom1);
   assign (f1,'c:\'+nom1+'.dat');
   write ('introduire le nom du deuxième fichier : ');
   readln (nom2);
   assign (f2,'c:\'+nom2+'.dat');
   eclat (f,f1,f2);  writeln;
   affiche (f1,' >= '); writeln;
   affiche (f2,' < ');
   maximum(f);
   close(f);
   close(f1);
   close(f2);
end.
```

VII- Les fichiers texte :

1. Présentation

Un fichier texte(ou ASCII) est un fichier contenant des caractères.

2. Déclaration

En algorithmique

T. D. O

Objet	Type/ Nature	Rôle
Nom_logique	Texte	Fichier texte pour…

En Pascal

```
VAR
    Nom_logique : text ;
```

Activité1 :

On veut écrire un programme permettant de saisir et enregistrer la liste des enseignants du lycée TAHAR SFAR Mahdia

En algorithmique

T. D. O

Objet	Type/ Nature	Rôle
enseignants	Texte	Fichier texte pour la saisie de la liste des enseignants

En Pascal

```
VAR
    Enseignants : text ;
```

Remarque :

Les fichiers texte contiennent des caractères de type "Retour chariot" (CR) ou "Fin de ligne" (Eoln) et "Fin de texte" (code CTRL-Z)

3. Procédures et fonctions prédéfinies

Rôle	Syntaxe en Algorithme	Syntaxe en Pascal
Fonction booléenne testant la fin d'une ligne dans un fichier texte.	**Fin_ligne** (nom logique)	**Eoln** (nom logique)
Fonction identique à Fin_ligne (Eoln) mais supprimant les espaces et les caractères de tabulation avant d'effectuer le test.	**Chercher_Fin_ligne** (nom logique)	**SeekEoln** (nom logique)
Fonction identique à Fin_Fichier (Eof) mais supprimant les espaces et les caractères de tabulation avant d'effectuer le test.	**Chercher_Fin_fichier** (nom logique)	**SeekEof** (nom logique)
Ouvrir un fichier et positionner son pointeur à la fin pour ajouter des enregistrements.	**Ajouter (**nom logique)	**Append (**nom logique)
Placer le pointeur du fichier texte au début de la ligne suivante	**Lire_nl** (nom logique)	**Readln (**nom logique)
Introduire dans le fichier texte une séquence CR_LF pour marquer la fin de ligne	**Ecrire_nl** (nom logique)	**Writeln (**nom logique)

Remarques:

- Pour écrire une ligne dans un fichier texte on peut utiliser l'une des deux méthodes suivantes :

▪ **Writeln** (F, ligne) ;	▪ **Write** (F, ligne) ; **Writeln** (F);

- Pour lire une ligne à partir d'un fichier texte on peut utiliser l'une des deux méthodes suivantes :
  - Repeat
    Readln (F, ligne) ;
    Until …….

    Si l'objet ligne est de type :
    Char : cette commande permet de lire un caractère par ligne
    Integer : cette commande permet de lire un entier par ligne
    String : cette commande permet de lire une ligne complète

  - Repeat
    Read (F, ligne) ;
    Until …….
    Readln(F);

    Si l'objet ligne est de type :
    Char, **integer** ou **string** : cette commande permet de lire une ligne complète

- Pour supprimer les espaces au début d'un fichier Texte :
  - **SeekEof** (F) ;

- Pour supprimer les espaces au début de chaque ligne d'un fichier Texte :
  - **SeekEoln** (F) ;

Exercice :

Ecrire un programme permettant de :

- saisir et enregistrer une liste des noms des enseignants. La liste se termine par un point (le point n'appartient pas au fichier)
- déterminer et afficher le nombre de lettres par nom d'enseignant.

```
Program Exercice;
Uses wincrt;
Var     F: text;   nom: string;

Procedure compter (var f: text);
Var   car: char; nbl: integer;
Begin
Reset (f);
While not (eof (f)) do
  Begin
     nbl: = 0;
     While not (eoln (f)) do
     Begin
       Read (f, car);
       If upcase (car) in ['A'..'Z'] then nbl: = nbl +1;
     End;
     Writeln ('le nombre de lettres = ', nbl);   Readln (f);
  End;
End;

Procedure saisie (var f: text);
Var   ph: string [10]; rep: char;
Begin
     Write ('donner un nom d''enseignant :');  readln (ph);
     While ph [length (ph)] <>'.' do
     Begin
          Writeln (f, ph);
          Write ('donner un nom d''enseignant :');
          Readln (ph) ;
     End;
End;

BEGIN {PP}
  Write ('entrer le nom du fichier : ');
  Readln (nom);
  Assign (F, 'c: \'+nom+'.fch');
 {$I-}
     Reset (F);
 {$I+}

If ioresult<>0 then
                    Begin
                       Rewrite (F);
                      Saisie (f);
                    End;
  Compter (f);
  Close (f);

END.
```