CHAPITRE : LES ENREGISTREMENTS

1. STRUCTURE DE DONNEES HETEROGENES

- Si on souhaite stocker des données qui n'ont pas forcément le même type au sein d'une même structure, par exemple : les informations liées à un produit quelconque, La première question qui se pose est : quelle est la structure de donnée nécessaire ?
- Il est clair que dans un tableau, tous les éléments le constituant ont obligatoirement le même type, Les tableaux ne sont pas une solution adéquate.
- La solution est de créer une nouvelle structure de type ' *enregistrement*' qui *comporte* à la fois les données *numériques* (quantité, prix unitaire, prix total) et les données (*alphanumériques* (Référence, désignation),
- Ce nouveau type ou cette *une nouvelle structure* permet de représenter des données hétérogènes (c'est-à-dire *de différents types)* par le regroupage de plusieurs éléments qui peuvent être simples fondée sur d'autres types existants ou complexes.

2. STRUCTURE D'UN ENREGISTREMENT (NOTION DE CHAMPS) :

- Un enregistrement est une structure de données qui est un peu complexe par rapport aux structures vues précédemment qui sont : les types primitifs tels que : entier, réel, caractère), chaine de caractères, et les vecteurs a une et deux dimension (tableaux et matrices).
- Cette structure (enregistrement) nous permette de représenter un ensemble de données dites ***champs*** *hétérogènes (pas nécessairement le même type)* dans la même variable.
- En comparaison avec la structure de vecteur, on peut considérer un enregistrement comme étant un tableau *à une dimension*, *où chaque case de ce tableau représente une composante d'un type donné qui s'appelle* ***un champ****.*
- *La différence réside dans :*
  - Une case de tableau est accessible à travers un indice entier, par contre un champ 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).
  - La déclaration de tableau en algorithmique se faite dans la clause variable seulement, mais la déclaration de l'enregistrement se réalise par les deux clauses type et variables.

2.1 DEFINITION

Un enregistrement appelé aussi un article ou record en anglais est composé d'un nombre fixe de champs défini par l'utilisateur qui peuvent être de types différents. La liste des champs est composée de définitions identiques à celle de variables. Par exemple l'enregistrement « date » est composé de trois champs « jour », « mois » et année.

2.2 DECLARATION

La déclaration des types structures se fait dans une section spéciale des algorithmes appelée ***Type,*** qui précède la section des variables et succède à la section (clause) des constantes.

La syntaxe générale pour déclarer un enregistrement est comme suit :

En algorithmique	En langage C
Type *<id_Enreg> =* ***Enregistrement*** *<id_ch1> : <type1>* *<id_ch1> : <type2>* *...* *<id_chN> : <typeN>* ***Fin_ <id_Enreg>***	***Typedef struct {*** *<type1> <id_ch1> ;* *<type2> <id_ch1> ;* *...* *<typeN> <id_chN> ;* ***}id-structure ;*** *.....*

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 de l'enregistrement sera comme suit :

En algorithmique	***En langage C***
Type *Produit = Enregistrement* *Reference : Chaîne [5]* *Designation : Chaîne[25]* *Quantite : Entier* *Prix_U : Réel* *Fin_produit* ***Variables*** *Prod : produit*	***Typedef struct*** *{* *char ref[5] ;* *char Designation[25];* *int quantite* *float prix_u ;* *} produit ;*

2.2.1 TYPE ENUMERE

- les variables de ce type prennent une valeur parmi un ensemble. par exemple pour un feu de circulation, la couleur est : vert, orange ou rouge.
- le langage c propose deux syntaxes pour définir ce type.

A). première syntaxe :

enum nouveautype {liste de symboles} ;

- Cette syntaxe définit un nouveau type qui s'appelle ***enum nouveautype***
- le domaine des variables de ce type est la liste des symboles fournis.
- ces symboles sont équivalents à des constantes entières commençant par 0.

En langage C	En algorithmique
Enum jour{dimanche, lundi, mardi, jour=(dimanche,lundi mercredi,jeudi,vendredi **Enum** jour j ; Jour j ; Main(){ J= jeudi ; ……..	Déclaration **Type** jour=(dimanche,lundi,mardi,mercredi jeudi,vendredi, samedi} ; , **variables** J : jour ; **début** J ← jeudi ; **Fin**

B DEUXIEME SYNTAXE :

Cette syntaxe permet de définir un nouveau (sans évoquer le mot clé ***enum***).

Syntaxe : **Typedef enum {liste de symboles} nouveautype** ;

Exemple : **Typede**f enum{dimanche, lundi, mardi, mercredi, jeudi, vendredi, samedi} jour;
Jour j ;

REMARQUES

- Chaque champ a exactement les mêmes propriétés qu'une variable de même type et doit avoir un nom différent.
- Les éléments d'un enregistrement peuvent être à leur tour des structures (tableaux, enregistrements, ……)

3. MANIPULATION DES STRUCTURES D'ENREGISTREMENTS

Comme pour les tableaux, il n'est pas possible de manipuler un enregistrement globalement, La manipulation d'un enregistrement se fait au travers de « ses champs ».

3.1 ACCES AUX CHAMPS D'UN ENREGISTREMENT

- Alors que les éléments d'un tableau sont accessibles au travers de leur indice, les champs d'un enregistrement sont accessibles à travers leur nom, grâce à l'opérateur '.'.
- Pour référencer ou accéder à un champ quelconque d'un enregistrement, on utilise La variable enregistrement (identificateur) suivie d'un point puis du nom du champ auquel on veuille accéder. Syntaxe :

  <Variable_Enregistrement>.<Nom du champ>
- Par exemple, pour accéder au champ *Quantite* de la variable *prod,* de l'exemple précédent, on utilise l'expression : **prod. Quantite**.

3.2 AFFECTATION

L'affectation de valeurs aux différents champs d'une variable de type d'enregistrement se fait en respectant la syntaxe :

En algorithmique	En langage C
Nom_variable ← valeur **Exemple** **prod. Quantite ← 50**	**Nom_variable = valeur ;** **Exemple** **prod. Quantite = 50**

NB : il est possible d'affecter une variable enregistrement dans une autre de même structure, dans ce cas tous les champs seront recopies.

3.3 LECTURE

Cette opération sert à remplir les champs d'un enregistrement par les valeurs entrées par l'utilisateur à travers le clavier, Sa syntaxe est :

En algorithmique	En langage C
Lire(Nomvariable.champ) **Exemple** **Lire(prod. Designation)**	**Scanf(''code format'',& Nom_variable .champ)** **Exemple** **scanf(''%s'',&prod. Designation) ;**

3.4 ECRITURE

Cette opération sert à afficher sur écran, le contenu des champs d'un enregistrement composant l'enregistrement, Sa syntaxe est :

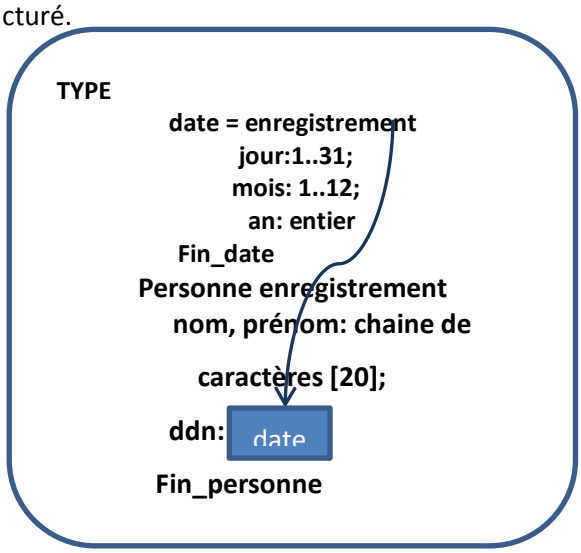
En algorithmique	En langage C
ecrire(Nomvariable.champ) Exemple **ecrire(prod.** Prix_U**)**	**printf(''code format'', Nom_variable .champ)** Exemple **printf (''%f'',** prod**.** Prix_U**) ;**

Remarque 1: la seule opération possible sur les enregistrements de même type est l'affectation

Remarque 2 : Nous ne pouvons pas lire ou écrire globalement un enregistrement, pour faire, il faudra lire ou écrire chaque champ qui le compose.

Remarque 3 : Un type structuré (enregistrement) peut être utilisé comme type pour des champs d'un autre type structuré.

Exemple :



- Pour accéder à l'année de naissance d'une personne, il faut utiliser deux fois l'opérateur '.'

Remarque 4: On peut définir et initialiser une structure enregistrement, en même temps et cela est possible grâce aux deux syntaxes suivantes :

- **Syntaxe 1**
  Struc nom-type nom-variable = {valeurs-champs} ;
- **Syntaxe 2**
  Nom-typedef nom-variable = {valeurs-champs} ;

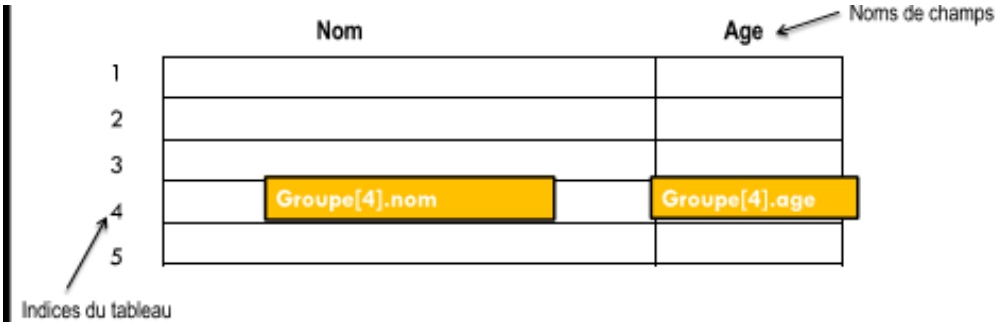
3.5 Tableau d'enregistrement

- Il arrive souvent que l'on veuille traiter non pas **un seul enregistrement** mais **plusieurs**.
- Par exemple, on veut pouvoir traiter un groupe de personne. On ne va donc pas créer autant de variables du type personne qu'il y a de personnes.
- On va créer un tableau regroupant toutes les personnes du groupe. « Il s'agit alors d'un ***tableau d'enregistrements*** »
  Exemple

En algorithmique	En langage C
Type Personne = enregistrement nom: chaine de caractère [15]; age: entier; Fin_personne **Variables** groupe: tableau [1..20] de personne;	***#define N 20*** ***Typedef struct*** *{* *Char nom[15] ;* *int age ; } personne ;* *main(){* typedef personne groupe[N];

Illustration :

- Chaque élément du tableau est un enregistrement, contenant plusieurs variables de type différent.
- On accède à un enregistrement par **son indice** dans le tableau.
- **groupe[4]** représente *la quatrième* personne du groupe
- **groupe[4].nom** représente **le nom** de la quatrième personne du groupe.
- **groupe[4].age** représente **l'âgé** de la quatrième personne du group



3.5 L'instruction avec faire

Pour ***simplifier*** l'écriture et éviter l'utilisation de la notion nom_variable.champ , nous pouvons utiliser l'accès **avec ….faire**

```
Type Tpersonne = record
        Age: integer;
        nom: String[20];
   end;

Var pers1: Tpersonne;
```

```
1- Accès direct
Pers1.age := 24;
Pers1.nom := 'Ali';

2- Accès avec l'instruction « With »
With pers1 do
   age := 24;
   nom := 'ALI';
End;
```