

Chapitre 6 LES STRUCTURES

Une structure permet de regrouper sous une même appellation des informations de type différent.

Déclaration

```
struct nom_structure
{
    type_champ1 nom_champ1;
    type_champ2 nom_champ2;
    .
    .
    .
    type_champN nom_champN;
};
```

Remarque :

Le mot clé **struct** peut être suivi d'un nom définissant le type de la structure

Affectations

soit la structure :

Typedef struct livre

```
{
    char auteur[12];
    char titre[20];
    int an;
} livres;
```

Les champs de la variable structurée l sont accessibles via les noms :

Livres lv ;

lv.auteur, lv.titre et lv.an

On pourrait avoir :

lv.an=1993;

strcpy(lv.auteur,"simenon");

Si on a des structures imbriquées :

Remarque :

On peut mettre :

l2=l1

Par contre, on ne peut pas comparer globalement des variables structurées. Il faut comparer chacun de leurs champs. **if(l1==l2)** n'est pas autorisé.

Tableaux de structures

Lorsque le nombre d'enregistrements à gérer est important, on peut alors créer un tableau dont les éléments sont des variables structurées.

struct livre2 lv[50];

Permet de déclarer un tableau de 50 éléments dont chacun est une variable structurée de type "struct livre2". Comme d'habitude, les éléments seront ici désignés par le nom du tableau avec un index, soit lv[0] à lv[49] dans notre exemple.

Pour accéder à un certain champ d'un quelconque élément du tableau, il faut le désigner selon la syntaxe suivante :

nom_tableau[index].nom_champ

On pourrait écrire :

lv[36].an=1997;

gets(lv[4].auteur);