Les *tableaux* permettent de désigner sous un seul nom un ensemble de valeurs de même type. En C et C++, il existe un autre type structuré définissable par le programmeur, qui cette fois peut désigner sous un seul nom un ensemble de valeurs avec des types différents. L'accès à chaque élément de la structure ne se fera pas par sa position, mais par son nom au sein de la structure. Un tel nom est communément appelé un *champ* de sa *structure*.

Voici un exemple de déclaration :

```
struct enreg
{  int numero ;
   int qte ;
   float prix ;
} ; // Attention au point-virgule qui est obligatoire
```

Cette déclaration définit un type (i.e. un modèle) de structure mais ne déclare pas de variable correspondant à cette structure. Ce type s'appelle *enreg* et il est constitué de 3 champs, numero, qte et prix.
Une fois défini un tel type, il est possible de déclarer une variable de type *enreg* :

```
enreg article ;
```

Pour manipuler les champs de la variable *article*, on utilisera l'opérateur '.' :

```
article.numero = 57 ;
```

Pour manipuler les champs d'une variable pointeur *p_article* (de type *enreg**), on utilisera l'opérateur '->' :

```
p_article->numero = 47 ;
```

Il est possible de faire des affectations globales entre 2 variables de type *enreg* (article_bis=article). Cela revient à effectuer une affectation des champs 1 à 1.

Remarque : une structure peut être vue comme une classe où tout est publiques (champs et méthodes de la structure).

Ce TD se réalise en binôme.

Exercice 1.
Définir une structure *pizza* qui contient le nom (une chaîne de caractères), la description des ingrédients (une chaîne de caractères), et le prix.

Exercice 2.
Ecrire les définitions des fonctions associées aux déclarations suivantes (séparer les déclarations des définitions des fonctions) :

```
void saisir( pizza& ) ;         // pour demander à l'utilisateur de saisir
                                // les champs (via std::cin)
void saisir( pizza* ) ;
void affiche( const pizza& ) ; // pour afficher les champs (via std::cout)
void affiche( const pizza* ) ;
```

Exercice 3.

- Déclarer 1 pizza (classe d'allocation automatique). L'initialiser (avec la fonction saisir) et l'afficher.
- Déclarer 1 pizza (classe d'allocation dynamique, donc allocation mémoire via l'opérateur `new`). L'initialiser et l'afficher. Libérer la mémoire allouée (opérateur `delete`).

- Déclarer un tableau de 5 pizzas (classe d'allocation automatique). Initialiser chaque pizza, et afficher le prix total des 5 pizzas.
- Déclarer un tableau dynamique de 4 pizzas (classe d'allocation dynamique, donc allocation mémoire via l'opérateur `new`). Initialiser les, et afficher le nombre de pizza au fromage. Libérer la mémoire allouée (opérateur `delete`).