# LES STRUCTURES

## 1. Structures

Une structure est le regroupement de plusieurs variables pouvant être de types différents sous un même nom, ces variables sont appelées champs.

Pour utiliser une variable structurée, il est préférable de créer un type. La syntaxe est :

```
typedef struct { type champ_1 ; type champ_2; … ; type champ_n ; } nom_du_type ;
```

Pour utiliser le type défini on crée des variables ou des pointeurs.

```
nom_du_type var1, var2, *var3;
```

L'accès aux différents champs des variables structurées se fait en indiquant précisant le champ à l'aide d'un point.

```
var1.champ_1 ou var1.champ_2 etc.
```

S'il s'agit d'un pointeur vers une structure, on accède aux champs à l'aide de l'opérateur -> ou en utilisant des parenthèses :

```
var3->champ_1 ou (*var3).champ_2
```

Il est possible d'affecter des structures entre elles, pour une fonction de retourner une structure.

```
var1 = var2; var2 = *var3; // affectation de structures
var1.champ_1=var2.champ_1; var2.champ_2 = var3->champ_2; // affectation de champs
nom_du_type MaFonction( ... ) ; // prototype d'une fonction retournant une structure
```

## 2. Exercice

L'objet est de définir les fonctions permettant de réaliser des calculs en complexe : addition, soustraction, multiplication, conjugué, module, division, angle, logarithme.

★ Saisie / affichage de nombres complexes. Dans chaque cas vous déterminerez le type des fonctions écrites ainsi que leurs paramètres d'entrée.

- ✦ Créez un type de variable structurée comportant deux champs re pour la partie réelle, im pour la partie imaginaire.
- ✦ Ecrivez une fonction qui affiche un nombre complexe transmis en paramètre à l'écran sous la forme a + ib. Les nombres seront affichés avec 4 décimales au minimum.
- ✦ Ecrivez une fonction qui réalise la saisie d'un nombre complexe et le retourne. La fonction devra demander la partie réelle, puis la partie imaginaire successivement. Cette fonction a-t-elle besoin d'un paramètre en entrée ?
- ✦ Testez votre programme en saisissant puis affichant un nombre complexe.

★ Ecrivez dans cet ordre les fonctions réalisant les opérations suivantes. Vous pouvez utiliser les fonctions précédemment écrites. Respectez le nom de chacune des fonctions qui vous est proposé. Déterminez pour chaque fonction le nombre de paramètres et le type de valeur retournée.

- ✦ Addition : C_add
- ✦ Soustraction : C_Sous
- ✦ Multiplication : C_mul
- ✦ Conjugué : C_conj
- ✦ Module : C_mod
- ✦ Division : C_div
- ✦ Angle : C_angle (regardez l'aide en ligne -fonction man- sur la fonction atan2)
- ✦ Logarithme Népérien : C_log

Testez vos fonctions avec des exemple connus (cf ci-après)

★ Modifiez la fonction saisie pour qu'elle soit de type void. (**A TRAITER IMPERATIVEMENT**)

## 3. Exemples de résultats

```
Entrez la partie reelle : 2
Entrez la partie imaginaire : 3
Entrez la partie reelle : -1
Entrez la partie imaginaire : 1
Addition :
   1.0000 + i   4.0000
Soustraction :
   3.0000 + i   2.0000
Multiplication :
  -5.0000 + i  -1.0000
Conjugue :
   2.0000 + i  -3.0000
Module :
  3.6056
Division :
   0.5000 + i  -2.5000
Angle :
  0.9828
Logarithme :
   1.2825 + i   0.9828
```