

### 5.4. Manipulation des variables en C

#### 5.4.1 Structure d'un programme en C :

Un programme en C se présente de la façon suivante :

[Directives au préprocesseur]	}	<b>Partie 1 : déclaration</b>
[Déclaration des variables externes]		
[Fonctions secondaires]		
Main ()	}	<b>Partie 2 : Corps du programme</b>
{		
Déclaration des variables internes		
Instructions		
}		

La partie **des déclarations** comporte la déclaration des fonctions des bibliothèques (bibliothèque standard ou autre) par inclusion de fichiers fournis avec le langage et peut comprendre des déclarations des variables « globales ». Elle peut contenir aussi des définitions de fonctions qui seront utilisées par l'intermédiaire de la fonction principale **main**.

Les variables internes et les instructions constituent le **corps du programme**. Elles sont plus ou moins complexes et nombreuses selon les programmes.

**Remarque :** parmi les bibliothèques standards fréquemment utilisées en C nous pouvons citer :

#### **Stdio.h**

Il s'agit des fonctions d'une librairie standard utilisées avec les unités classiques d'entrées-sorties, qui sont respectivement le clavier et l'écran. L'appel à la librairie *stdio.h* se fait par la directive au préprocesseur

```
#include <stdio.h>
```

#### **math.h**

Il s'agit un groupe de fonctions de la *bibliothèque* standard du C qui permet d'utiliser un ensemble de fonctions mathématiques de base. L'appel à la librairie *math.h* se fait par la directive au préprocesseur

```
#include <math.h>
```

#### 5.4.2. La fonction de saisie scanf (Anne CANTEAUT)

La fonction *scanf* permet de saisir des données au clavier et de les stocker aux adresses spécifiées par les arguments de la fonction.

Scanf ("chaîne de contrôle", argument-1,..., argument-n)

La chaîne de contrôle indique le format dans lequel les données lues sont converties. Elle ne contient pas d'autres caractères (notamment pas de \n). Comme pour *printf*, les conversions

de format sont spécifiées par un caractère précédé du signe %. Les formats valides pour la fonction scanf diffèrent légèrement de ceux de la fonction printf.

Les données à entrer au clavier doivent être séparées par des blancs ou des <RETURN> sauf s'il s'agit de caractères. On peut toutefois fixer le nombre de caractères de la donnée à lire.

Par exemple %3s pour une chaîne de 3 caractères, %10d pour un entier qui s'étend sur 10 chiffres,

signe inclus.

Exemple :

```
#include <stdio.h>
main()
{
    int i;
    printf("entrez un entier sous forme hexadecimale i = ");
    scanf("%x", &i);
    printf("i = %d\n", i);
}
```

Si on entre au clavier la valeur 1a, le programme affiche i = 26.

format	type d'objet pointé	représentation de la donnée saisie
%d	int	décimale signée
%hd	short int	décimale signée
%ld	long int	décimale signée
%u	unsigned int	décimale non signée
%hu	unsigned short int	décimale non signée
%lu	unsigned long int	décimale non signée
%o	int	octale
%ho	short int	octale
%lo	long int	octale
%x	int	hexadécimale
%hx	short int	hexadécimale
%lx	long int	hexadécimale
%f	float	flottante virgule fixe
%lf	double	flottante virgule fixe
%Lf	long double	flottante virgule fixe
%e	float	flottante notation exponentielle
%le	double	flottante notation exponentielle
%Le	long double	flottante notation exponentielle
%g	float	flottante virgule fixe ou notation exponentielle
%lg	double	flottante virgule fixe ou notation exponentielle
%Lg	long double	flottante virgule fixe ou notation exponentielle
%c	char	caractère
%s	char*	chaîne de caractères

### 5.4.3. La fonction d'écriture `printf` (Anne CANTEAUT)

La fonction `printf` est une fonction d'impression formatée, ce qui signifie que les données sont converties selon le format particulier choisi. Sa syntaxe est `printf ("chaîne de contrôle", expression-1, ..., expression-n);`

La *chaîne de contrôle* contient le texte à afficher et les spécifications de format correspondant à chaque expression de la liste. Les spécifications de format ont pour but d'annoncer le format des données à visualiser. Elles sont introduites par le caractère %, suivi d'un caractère désignant le format d'impression. On peut éventuellement préciser certains paramètres du format d'impression, qui sont spécifiés entre le % et le caractère de conversion dans l'ordre suivant :

- largeur minimale du champ d'impression : `%10d` spécifie qu'au moins 10 caractères seront réservés pour imprimer l'entier. Par défaut, la donnée sera cadrée à droite du champ. Le signe - avant le format signifie que la donnée sera cadrée à gauche du champ (`%-10d`).
- précision : `%.12f` signifie qu'un flottant sera imprimé avec 12 chiffres après la virgule. De même `%10.2f` signifie que l'on réserve 12 caractères (incluant le caractère.) pour imprimer le flottant et que 2 d'entre eux sont destinés aux chiffres après la virgule. Lorsque la précision n'est pas spécifiée, elle correspond par défaut à 6 chiffres après la virgule. Pour une chaîne de caractères, la précision correspond au nombre de caractères imprimés : `%30.4s` signifie que l'on réserve un champ de 30 caractères pour imprimer la chaîne mais que seulement les 4 premiers caractères seront imprimés (suivis de 26 blancs).

format	conversion en	écriture
<code>%d</code>	<code>int</code>	décimale signée
<code>%ld</code>	<code>long int</code>	décimale signée
<code>%u</code>	<code>unsigned int</code>	décimale non signée
<code>%lu</code>	<code>unsigned long int</code>	décimale non signée
<code>%o</code>	<code>unsigned int</code>	octale non signée
<code>%lo</code>	<code>unsigned long int</code>	octale non signée
<code>%x</code>	<code>unsigned int</code>	hexadécimale non signée
<code>%lx</code>	<code>unsigned long int</code>	hexadécimale non signée
<code>%f</code>	<code>double</code>	décimale virgule fixe
<code>%lf</code>	<code>long double</code>	décimale virgule fixe
<code>%e</code>	<code>double</code>	décimale notation exponentielle
<code>%le</code>	<code>long double</code>	décimale notation exponentielle
<code>%g</code>	<code>double</code>	décimale, représentation la plus courte parmi <code>%f</code> et <code>%e</code>
<code>%lg</code>	<code>long double</code>	décimale, représentation la plus courte parmi <code>%lf</code> et <code>%le</code>
<code>%c</code>	<code>unsigned char</code>	caractère
<code>%s</code>	<code>char*</code>	chaîne de caractères