

Chapitre 5 : LES Fonctions et chaines de caractères

Définition d'une fonction

Pour construire une fonction, il faut coder les instructions qu'elle doit exécuter.

type nom_de_fonction (liste des paramètres)

```
{  
    corps de la fonction  
}
```

Remarque :

S'il n'y a pas de paramètres, les parenthèses sont quand même obligatoires

Exemple 1 :

#include

```
void message()  
  
{  
    printf("cours de langage C\n");  
}
```

//L'appel

```
void main()  
  
{  
    message();  
}
```

Si les fonctions sont définies après le main (comme conseillé), il est nécessaire de les déclarer avant toute utilisation.

Cette déclaration s'appelle aussi le **prototype** de la fonction.

Déclaration de fonction

type NomFonction (liste des types des paramètres);

Exemple 2 : calcul du cube d'un nombre.

```
#include

double cube(double);    /*déclaration ou prototype de la fonction*/

void main()
{
    double e;
    double volume;
    printf("longueur de l'arête : ");
    scanf("%lf",&e);
    volume=cube(e);      /*appel de la fonction cube et affectation de
                           la valeur retournée à la variable volume*/
    printf("\nle volume du cube est : %lf",volume);
}

double cube(double x)    /*définition de la fonction */
{
    double y;
    y=x*x*x;
    return y;
}
```

Nous voyons dans cet exemple que la fonction cube renvoie une valeur de type double. Cette valeur est ici renvoyée à la fonction main.

Appel d'une fonction

Pour appeler la fonction cube définie précédemment, il suffit d'écrire l'instruction :

y=cube(3); ou directement **printf("%f",cube(3));**

La fonction appelée reçoit le contrôle et commence l'exécution de ses instructions.

Tableaux comme paramètre d'une fonction

1. Tableau à une dimension

Soit T un tableau d'entiers défini dans main(). Si on veut transmettre l'adresse du tableau T à une fonction fonc, il suffira d'écrire : fonc(T)

Le paramètre formel pourra s'écrire :fonc(int T[]).

Exemple :

Affichage d'un tableau.

```
#include <stdio.h>

void affiche(int*,int);

void main()
{
    int T[10],i;
    for(i=0;i<10;i++)
        scanf("%d",&T[i]);
    affiche(T,10);
}

void affiche(int z[],int l)
{
    int i;
    for(i=0;i<l;i++)
        printf("\n%d",z[i]);
}
```

Les chaînes de caractères :

Initialisation

Comme tous les autres tableaux, les variables chaîne peuvent être initialisées **dès leur définition.**

```
char s[10]="bonjour";
```

Après la définition :

Après avoir défini la chaîne s:

```
char s[13];
```

on peut alors, pour affecter la chaîne "bonjour" au tableau s, écrire :

```
s[0]='b';
```

```
s[1]='o';
```

```
s[2]='n';
```

```
.  
. .  
. .
```

```
s[7]='\0';
```

La fonction strcpy

La fonction **strcpy** copie une chaîne dans une variable

Soit s défini par :

```
char s[8];
```

```
strcpy(s,"bonjour"); y recopie la chaîne "bonjour" dans s.
```

Remarque :

L'utilisation de strcpy et de quelques autres fonctions de chaînes de caractères exige l'inclusion dans le programme du fichier d'en-tête : **string.h** qui contient les déclarations des fonctions concernées.

strcat, strcmp, strlen

La fonction strcat

Concaténation de chaînes de caractères

On peut concaténer deux chaînes de caractères en accrochant l'une d'elles à la fin de l'autre. Cette opération s'effectue via la fonction strcat dont la syntaxe générale est :

strcat (ch1,ch2);

La fonction strcmp

Comparaison de chaînes de caractères

La fonction **strcmp** compare deux chaînes caractère par caractère. La comparaison s'effectue dans l'ordre alphanumérique.

Int n=Strcmp(ch1,ch2);

Cette fonction retourne une valeur entière avec les conventions suivantes :

- si la valeur est inférieure à 0, alors ch1 est inférieure à ch2
- si la valeur est égale à 0, alors les deux chaînes sont égales
- si la valeur est supérieure à 0, ch1 est supérieure à ch2.

La fonction strlen

Longueur d'une chaîne de caractères

La fonction **strlen** calcule la longueur d'une chaîne de caractères:

Int nb=strlen(ch);