

COURS DE C++

4 - Fonctions

APPELS DE SOUS-PROGRAMMES

- Jusqu'à présent, même si nous avons structuré nos programmes, ceux-ci sont monolithiques.
- Pour pouvoir réutiliser un même calcul autrement qu'en effectuant une boucle, il est nécessaire de le copier.
- Les fonctions vont permettre de définir des sous-programmes que l'on pourra appeler.

DÉFINITION D'UNE FONCTION

- Une fonction c'est un bloc d'instructions dans lequel est prédéfini un certains nombres de variables, appelées arguments, et renvoyant une valeur appelée valeur de retour.

```
type_retour nom_fonction(type1 arg1, ...,  
{  
    instructions...  
    return valeur_de_retour;  
}
```

Pour l'appeler il suffit d'utiliser `nom_fonction(v1, ..., vn)`

DÉROULEMENT D'UN APPEL

```
int x = somme(2,3);
```

↓ Copie des arguments dans un nouvel environnement

```
n ← 2  
m ← 3
```

```
int somme(n,m) {  
    return n+m;  
}
```

On dit que C++ effectue de l'appel par valeurs.
On ne peut pas modifier les arguments directement.

ARGUMENT VOID

Si votre fonction ne renvoie rien, son type de retour doit être `void`

Pour ne rien renvoyer vous pouvez utiliser `return` ;

- cela sert pour interrompre le cours d'une fonction

Si vous n'avez pas d'arguments vous pouvez déclarer `f(void)` ou plus simplement `f()`

RÉCURRENCE

- Une fonction peut s'appeler elle-même : fonction récurrente.
- A réserver à des cas très particuliers.

VARIABLES STATIQUES

- Il peut être intéressant de garder un état courant dans la fonction qui va évoluer au cours des appels
- Pour cela on peut déclarer une variable comme étant statique.

- Exemple :

```
int compteur() {  
    static int c = 0;  
    return c++;  
}
```

- A noter que la l'initialisation ne s'effectue que lors du premier passage.