

BÀI 11: HÀM

1. Cấu trúc của hàm

Cú pháp

```
<kiểu DL> tên_hàm([<DS tham số>])  
{  
    <thân hàm>;  
}
```

Trong đó:

+ <kiểu DL> kiểu dữ liệu của kết quả trả về cho hàm sau khi thực hiện, có thể là DL cơ bản hoặc kiểu được định nghĩa. Nếu hàm không trả về giá trị nào thì kiểu DL là void;

+ <ds tham số> là các tham số hình thức được khai báo tương tự khai báo biến. Các tham số hình thức cách nhau bởi dấu phẩy;

+ <thân hàm> có thể gồm: khai báo biến cục bộ và các câu lệnh. Có thể sử dụng câu lệnh return ([biểu thức]) để chuyển giá trị trả về cho nơi gọi hàm.

Chú ý:

+ Hàm có thể khai báo trước hoặc sau hàm main(). Nếu hàm khai báo sau hàm main() thì cần khai báo nguyên mẫu hàm;

+ Nếu không chỉ định kiểu trả về cho hàm thì mặc định là kiểu int;

+ Không sử dụng câu lệnh return (<biểu thức>) để kết thúc hàm kiểu void bằng

+ Có thể dùng tham số void cho hàm không có tham số.

2. Khai báo nguyên mẫu hàm

Nếu định nghĩa hàm sau hàm main() thì cần phải khai báo nguyên mẫu hàm

Cú pháp

```
<kiểu dl> tên_hàm([<ds tham số>]) ;
```

Ví dụ

```
#include <stdio.h>  
#include <conio.h>  
Int max(int a,b,c);  
int main() {  
    Int x,y,z;  
    Printf("nhap 3 so");  
    Scanf("%d%d%d",&x,&y,&z);  
    Printf('Max =%d',max(x,y,z));  
}
```

```

Int max(int a,b,c){
    Int t;
    T =a;
    If (t<b) t=b;
    If (t<c) t=c;
    Return(t);
}

```

3. Lời gọi hàm

Cú pháp

<tên hàm>([<ds tham số thực sự>]);

Trong đó < ds tham số thực sự > là các giá trị cụ thể được truyền vào cho hàm, chúng được ngăn cách nhau bởi dấu phẩy. Giữa tham số hình thức và tham số thực sự phải tương ứng nhau về kiểu và số lượng.

Có 3 cách truyền tham số khi gọi hàm

a. Truyền tham số theo giá trị

Cách truyền này không làm thay đổi giá trị truyền vào

Vd: chương trình sau không làm thay đổi giá trị 2 số

```

#include <stdio.h>
void swap(int, int);
int main()
{
    int x,y;
    printf("x="); scanf("%d",&x)
    printf("y="); scanf("%d",&y);
    swap(x,y);
    printf("\nx=%d",x);
    printf("\ny=%d",y);
    return 0 ;
}
void swap(int a, int b){
    int t;
    t=a;
    a=b;
    b=t
}

```

b. Truyền tham số địa chỉ

Để truyền tham số theo địa chỉ thì tham số hình thức phải khai báo kiểu con trỏ.

Con trỏ là biến chứa địa chỉ của biến khác

Cách khai báo: <kiểu> ds biến con trỏ

Vd: `int *p, a;`

P là biến con trỏ, a là biến thường.

để trỏ p vào a : `p=&a;`

lấy giá trị mà p trỏ tới : `*p`

Ví dụ : viết lại hàm swap để hoán đổi 2 số

```
#include <stdio.h>
void swap(int *a, int *b);
int main() {
    int x, y;
    printf("x="); scanf("%d", &x);
    printf("y="); scanf("%d", &y);
    swap(&x, &y);
    printf("\nx=%d", x);
    printf("\ny=%d", y);
    return 0;
}
void swap(int *a, int *b) {
    int t;
    t=*a;
    *a=*b;
    *b=t;
}
```

c. Truyền tham số theo tham chiếu

C++ đưa thêm cách truyền tham số gọi là truyền theo tham chiếu. Biến tham chiếu là bí danh của biến đã khai báo trước đó.

Cú pháp:

`<kiểu> &biến tham chiếu = biến;`

Vd: `int a, &tong=a;` → tong là bí danh của biến a. tong và a dùng chung một vùng nhớ.

Vd: hàm swap có thể viết lại như sau:

```
void swap(int &a, int &b) {
    int t;
    t=a;
    a=b;
    b=t;
}
```

Khi gọi

`swap(x, y) ;`

4. Một số ví dụ

VD1: Viết hàm tính $s=1+2+\dots+N$

```
long sum(int n){
    If (n=1) return (1)
    Else return (n+sum(n-1))
}
```

VD2: Viết hàm tính $N!$

```
long gt(int n){
    If (n=1) return (1)
    Else return (n*gt(n-1))
}
```

VD3:

Dạng không đệ quy;

```
int UCLN(int a, int b){
    int r;
    do
        r=a % b;
        a=b;
        b=r;
    while (r!=0)
    return(a);
}
```

Dạng đệ quy

```
int UCLN(int a, int b){
    If (a==b) return(a)
    Else if (a>b) return(UCLN(a-b,b)
    Else return (UCLN(a,b-a))
}
```