

Hàm - function

- Khái niệm, mục đích sử dụng
- Cú pháp tổng quát
- Hàm trả về, không trả về
- Tham số, đối số
- Ví dụ minh họa

Khái niệm

- Các sản phẩm phần mềm thường có hàng ngàn dòng code hoặc hơn => khó bảo trì nâng cấp nếu không chia nhỏ thành từng phần để dễ quản lý
- Hàm là tập hợp của một nhóm các câu lệnh cùng thực hiện trọn vẹn một chức năng, được đặt cho một cái tên chung và có dấu hiệu nhận biết cụ thể
- Mỗi hàm có kiểu, tên hàm, các tham số và phần thân hàm
- Ví dụ hàm `main()`, `printf()`, `scanf()`, `puts()`

Mục đích sử dụng

- Có thể modul hóa chương trình
- Tái sử dụng lại mã nguồn(code)
- Dễ dàng bảo trì
- Dễ dàng mở rộng, nâng cấp
- Quản lý chương trình hiệu quả

Cú pháp tổng quát

```
data_type function_name(parameter_list) {  
    // statements  
}
```

Trong đó:

- Data_type: kiểu trả về của hàm, có thể là bất kì kiểu dữ liệu hợp lệ nào trong ngôn ngữ C
- Function_name: tên hàm, thường là một hành động thể hiện chức năng mà hàm đảm nhiệm, vd: add, subtract, speak, takeDamage

Cú pháp tổng quát

```
data_type function_name(parameter_list) {  
    // statements  
}
```

- Sau tên hàm là cặp () chứa tham số bên trong
- Parameter_list: danh sách tham số của hàm, một hàm có thể có 0, 1 hoặc nhiều tham số phân tách nhau bằng dấu phẩy
- Phần thân hàm trong {} chứa các câu lệnh thực hiện nhiệm vụ của hàm

Hàm trả về vs hàm không trả về

- Khi nói về hàm ta có hai thao tác:
1 là định nghĩa hàm
2 là gọi hàm, sử dụng hàm
- Trả về tức là việc đáp lại một lời gọi hàm nào đó
- Hàm trả về là hàm có kiểu khác void
- Sau khi thực hiện nhiệm vụ, hàm trả về sẽ gửi cho nơi gọi nó một giá trị cùng kiểu với kiểu của hàm thông qua keyword **return**
- Ví dụ:

```
// ham tra ve ket qua tong a va b
int add(int a, int b) {
    return a + b;
}
```

Hàm trả về vs hàm không trả về

- Hàm không trả về là các hàm có kiểu là void
- Các hàm này sau khi thực hiện nhiệm vụ thì không cần thông báo kết quả cho nơi gọi nó
- Sử dụng hàm trả về khi kết quả của hàm sẽ là đầu vào để tiếp tục thực hiện các chức năng khác trong chương trình
- Sử dụng hàm không trả về khi kết quả thực hiện của hàm không tiếp tục được sử dụng nữa
- Ví dụ:

```
void hienKetQua(int kq) {  
    printf("KQ = %d\n", kq);  
}
```

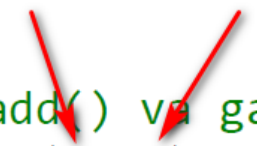
Tham số, đối số

- Tham số là các biến được khai báo trong cặp () định nghĩa hàm, ví dụ:

```
// ham tra ve ket qua tong a va b
int add(int a, int b) {
    return a + b;
}
```

- Đối số là các giá trị truyền vào hàm khi thực hiện lời gọi hàm, ví dụ:

```
int a = 5;
int b = 6;
// goi ham add() va gan kq nhan dc cho sum
int sum = add(a, b);
```



Tham số, đối số

- Tham số luôn là các biến vì nó có nhiệm vụ nhận giá trị được truyền vào hàm ở lời gọi hàm

```
// ham tra ve ket qua tong a va b
int add(int a, int b) {
    return a + b;
}
```

- Đối số có thể là các biến, hằng số hoặc giá trị cụ thể của một kiểu nào đó

```
int a = 5;
int b = 6;
// goi ham add() va gan kq nhan dc cho sum
int sum = add(a, b);
```

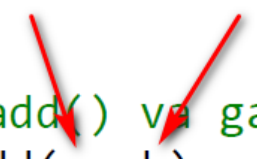
Tham số, đối số

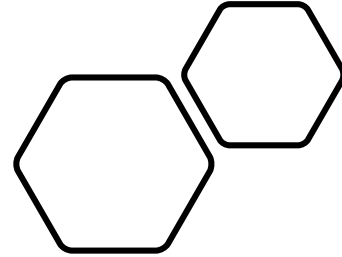
- Số lượng, thứ tự và kiểu của tham số và đối số phải trùng nhau. Kiểu của đối số có thể nhỏ hơn hoặc bằng kiểu của tham số
- Tên của tham số và đối số không cần phải trùng nhau

- Ví dụ:

```
// ham tra ve ket qua tong a va b
int add(int a, int b) {
    return a + b;
}
```
- Lời gọi:

```
int a = 5;
int b = 6;
// goi ham add() va gan kq nhan dc cho sum
int sum = add(a, b);
```





Tiếp theo

Giới thiệu thư viện `<math.h>`