

- Bản chất
- Ý nghĩa sử dụng
- Ví dụ minh họa

Truyền mạng vào hàm



Bản chất

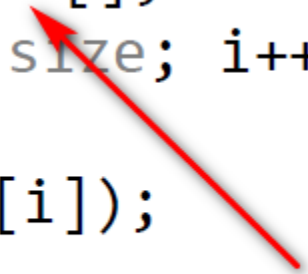
- Bản chất của tên mảng là một con trỏ cố định luôn trỏ đến phần tử đầu tiên trong mảng
- Khi truyền mảng vào hàm, chương trình sẽ truyền tham chiếu chứ không phải truyền giá trị

Cú pháp

- Hàm nhận tham số là biến kiểu mảng sẽ có cặp móc [] sau tên, trong [] không nhất thiết phải có tham số
- Hàm này thường có thêm một tham số để nhận số phần tử thực tế của mảng

• Ví dụ:

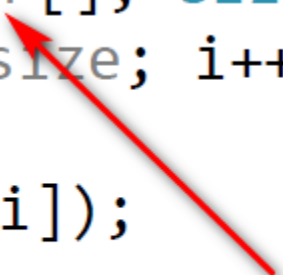
```
void showArrayElements(int arr[], size_t size) {  
    for (size_t i = 0; i < size; i++)  
    {  
        printf("%d ", arr[i]);  
    }  
}
```



Gọi hàm

- Khi gọi hàm để truyền mảng đi, ta cung cấp tên mảng và số phần tử thực tế tương ứng

```
void showArrayElements(int arr[], size_t size) {  
    for (size_t i = 0; i < size; i++)  
    {  
        printf("%d ", arr[i]);  
    }  
}
```



- Lời gọi hàm:

```
puts("\nAll array elements: ");  
showArrayElements(arr, size);
```

Ý nghĩa sử dụng

- Việc truyền mảng vào hàm sẽ giúp ta thực hiện việc modul hóa chương trình, phân tách chức năng độc lập và có thể tái sử dụng dễ dàng
- Do mảng truyền vào hàm theo kiểu tham chiếu nên việc gọi hàm đơn giản không tốn nhiều chi phí
- Nếu muốn giới hạn chức năng của hàm ở mức chỉ đọc, ta dùng keyword const:

```
void showArrElements(const int arr[]) {  
    for (size_t i = 0; i < SIZE; i++)  
    {  
        printf("%5d", arr[i]);  
    }  
    arr[0] = 120; // error!  
}
```

Tiếp theo

Sắp xếp các phần tử trong mảng

