HVIT CLAN JSC

C Programming

Xây dựng hàm

Nội dung slide: Đi sâu vào xây dựng hàm

Thuyết minh: Nguyễn Đồng Khánh

HVIT CLAN 2020

NỘI DUNG CHÍNH (3 nội dung)

Truyền giá trị hay địa chỉ?

Truyền mảng cho hàm

Xây dựng nguyên mẫu

Phần 1

Truyền địa chỉ hay giá trị ?

Một ví dụ về địa chỉ và giá trị

Xét ví dụ dưới đây:

```
1  #include <stdio.h>
2  #include <conio.h>
3  void Tang(int so)
4  {
5      so = so + 1;
6  }
7  void main()
8  {
9      int so = 10;
10     Tang(so);
11     Tang(so);
12     printf("%d", so);
13 }
```

Mục đích xây dựng hàm là dùng để tăng một giá trị cho biến số nguyên truyền vào. Và kết quả in ra trong hàm main vẫn là 10 !

*Biến **so** khai báo trong **hàm main** nằm ở ô thứ 8 trong bộ nhớ*

*Biến **so** khai báo trong **hàm Tang** nằm ở ô thứ 18 trong bộ nhớ*

bộ nhớ RAM					
ô1	ô2	ô3	ô4	ô5	ô6
ô7	ô8 10	ô9	ô10	ô11	ô12
ô13	ô14	ô15	ô16	ô17	ô18 10 →11
ô19	ô29	ô21	ô22	ô23	ô24
ô25	ô26	ô27	ô28	ô29	ô30

Biến con trỏ

Khai báo biến con trỏ:

kiểu* tên_biến = new kiểu ;

Cách sử dụng:

Biến A	Lấy Giá Trị	Lấy Địa Chỉ
Là Biến Con Trỏ	*A	A
Là Biến Bình Thường	A	&A

Một số ví dụ:

```
7   void main()
8   {
9       int* a = new int;
10      int b = 20;
11      a = &b;
12      *a = 30;
13      printf("%d", b);
14  }
```

```
1   #include <stdio.h>
2   #include <conio.h>
3   void Tang(int* so)
4   {
5       *so = *so + 1;
6   }
7   void main()
8   {
9       int so = 10;
10      Tang(&so);
11      printf("%d", so);
12  }
```

```
1   #include <stdio.h>
2   #include <conio.h>
3   void Tang(int& so)
4   {
5       so = so + 1;
6   }
7   void main()
8   {
9       int so = 10;
10      Tang(so);
11      printf("%d", so);
12  }
```

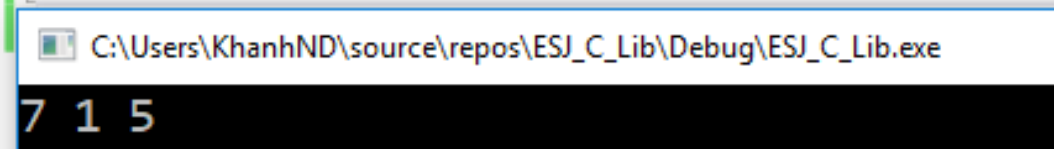
Phần 2

Truyền mảng cho hàm

Truyền mảng cho hàm

Về bản chất, mảng chính là một con trỏ ở đầu của một dãy số. Ta truy cập đến các vị trí còn lại thông qua việc gọi đến các địa chỉ tiếp theo: *(địa chỉ + số) hoặc cách viết đơn giản hơn : tên mảng[số]

```c
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[] = { 7,4,1,2,5,8 };
    printf("%d %d %d", *a, *(a + 2), *(a + 4));
    _getch();
}
```



Do vậy, để truyền mảng vào cho hàm, ta chỉ việc truyền vào con trỏ đầu vào cho mảng. Ví dụ như sau:

```c
#include <stdio.h>
#include <conio.h>
void Hien(int* a, int n)
{
    for (int i = 0; i < n; i++)
        printf("%d", a[i]);
}
void main()
{
    int a[] = { 7,4,1,2,5,8 };
    Hien(a, 6);
    _getch();
}
```

Phần 3

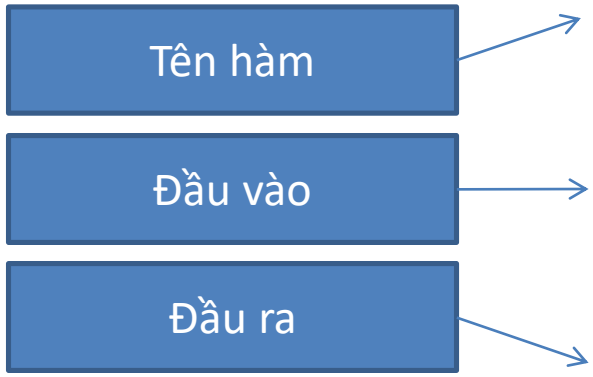
Xây dựng nguyên mẫu

Xây dựng nguyên mẫu

Xây dựng nguyên mẫu hàm:

Là việc quan trọng để xác định trước tên hàm, đầu vào, đầu ra của hàm. Giúp các hàm nắm bắt trước được sự tồn tại của nhau. Đối với người học, việc này còn có ý nghĩa quan trọng khác nhằm nâng cao tối đa việc xây dựng một hàm sao cho đúng.

Các bước xây dựng:



Tên hàm cần bắt đầu bằng một động từ. Các từ tiếp theo có thể mô tả thêm về hàm.
Tên hàm nên gợi ý hoặc giúp người sử dụng hàm hiểu hàm dùng để làm gì?

Xác định CHÍNH XÁC và ĐỦ chứ không được THỪA hay không THIẾU biến truyền vào.
Nếu thừa, thiếu hoặc sai thì khi triển khai nội dung hàm sẽ lủng củng và khó khăn.

Giá trị trả về của hàm có thể là một giá trị đơn lẻ hoặc nhiều giá trị hoặc không trả về.
Cần phải xác định rõ điều này từ yêu cầu viết hàm.

Ví dụ:

1. Xây dựng hàm hiển thị một số có phải số nguyên tố hay không? → *void HienThiNguyenTo(int so);*
2. Xây dựng hàm xác định một số có phải số nguyên tố hay không? → *int XacDinhNguyenTo(int so);*
3. Xây dựng hàm tính tổng của một dãy số thực n phần tử? → *double TinhTong(double* dayso, int n);*
4.Xây dựng hàm đếm số ký tự viết hoa của chuỗi ký tự ? → *int DemVietHoa(char* chuoi);*
5. Xây dựng hàm đảo giá trị của hai biến ký tự? → *void DaoGiaTri(char& a, char& b);*

TRÂN TRỌNG
CẢM ƠN