II. HÀM VÀ CẤU TRÚC CHƯƠNG TRÌNH

TỔ CHỨC CHƯƠNG TRÌNH THÀNH CÁC HÀM

XÂY DỰNG HÀM VÀ SỬ DỤNG HÀM

CON TRỎ VÀ ĐỊA CHỈ

CON TRỎ VÀ MẢNG MỘT CHIỀU

CON TRỎ VÀ MẢNG NHIỀU CHIỀU

KIỂU CON TRỎ, KIỂU ĐỊA CHỈ, CÁC PHÉP TOÁN TRÊN CON TRỎ

MẢNG CON TRỎ

CON TRỎ TỚI HÀM

ĐỆ QUY

ĐỐI DÒNG LỆNH

Hàm và chương trình

- Chương trình (program):

+ Một chương trình bao gồm một hoặc nhiều hàm

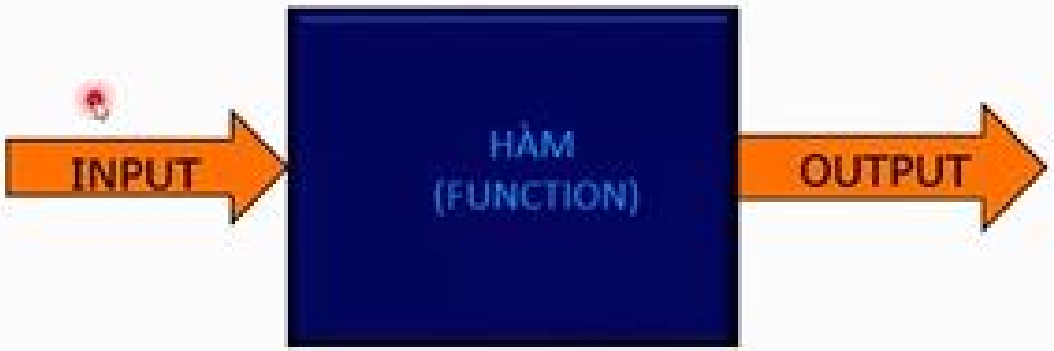
+ Hàm main() là hàm bắt buộc của chương trình

+ Chương trình luôn bắt đầu thực hiện từ câu lệnh đầu tiên của hàm main() và thường kết thúc khi gặp dấu } cuối cùng của nó.

- Hàm (Function): là một đoạn chương trình độc lập thực hiện trọn vẹn một công việc nhất định, nó thường trả về cho chương trình gọi nó một giá trị.

Note: Không cho phép xây dựng một hàm bên trong một hàm khác.

Hàm



Tổ chức chương trình

.....

hàm 1

.....

hàm 2

.....

hàm n

1- ....: #include, #define, định nghĩa các kiểu dữ liệu, khai báo biến ngoài

2- Việc truyền dữ liệu từ hàm này sang hàm khác?

=>

- Sử dụng biến toàn bộ, biến ngoài
- Sử dụng đối/giá trị trả về của hàm

Xây dựng hàm

Để viết một hàm, trước hết ta phải xác định mục đích của hàm là dùng để làm gì, từ đó xác định các thành phần:

- Nguyên mẫu hàm
- Kiểu giá trị trả về của hàm (kiểu hàm)
- Tên hàm
- Các tham số (đối số) của hàm
- Nội dung hàm

Xây dựng hàm – Nguyên mẫu hàm (Prototype)

< kiểu dữ liệu của hàm> <**tên hàm**(*danh sách các tham số*)>;

- Có thể viết hoặc không viết nguyên mẫu

- Các nguyên mẫu hàm (nếu có) cần đặt trước hàm main()

Ví dụ:

double exp(double x);

double pow(double x, double y);

*char *strcat(char *des, const char *source);*

Xây dựng hàm – Kiểu giá trị của hàm

- Giá trị của hàm được xác định dựa vào mục đích của hàm
- Trong thân hàm trả về (câu lệnh return) phải trả về đúng kiểu hàm đã định
- Các hàm không trả về giá trị ta sử dụng kiểu hàm là: void

Xây dựng hàm – Tên hàm

- Theo quy định đặt tên (định danh – identifier)
- Nên ngắn gọn và phản ánh phần nào mục đích của hàm
- Tên hàm trong nguyên mẫu, khai báo và sử dụng phải giống nhau

Xây dựng hàm – Tham số của hàm

- Các tham số mà ta ghi trong nguyên mẫu hay lúc khai báo hàm gọi là ***tham số hình thức***
- Các giá trị, biến mà ta ghi sau tên hàm được gọi để thực hiện hàm đó gọi là ***tham số thực***
- Tham số của một hàm có hai công dụng:
  - *+ Cung cấp các giá trị cho hàm khi ta gọi nó thực hiện*
  - *+ Lưu các kết quả tính toán được trong quá trình hàm hoạt động*

Như vậy ta có thể chia thành:

+ Tham số vào: cung cấp giá trị cho hàm
+ Tham số ra: Lưu kết quả tính toán được trong hàm
+ Tham số vừa vào, vừa ra.

Xây dựng hàm – Thân hàm

- Thân hàm là nội dung chính của hàm bắt đầu bằng dấu { và kết thúc bởi dấu }.

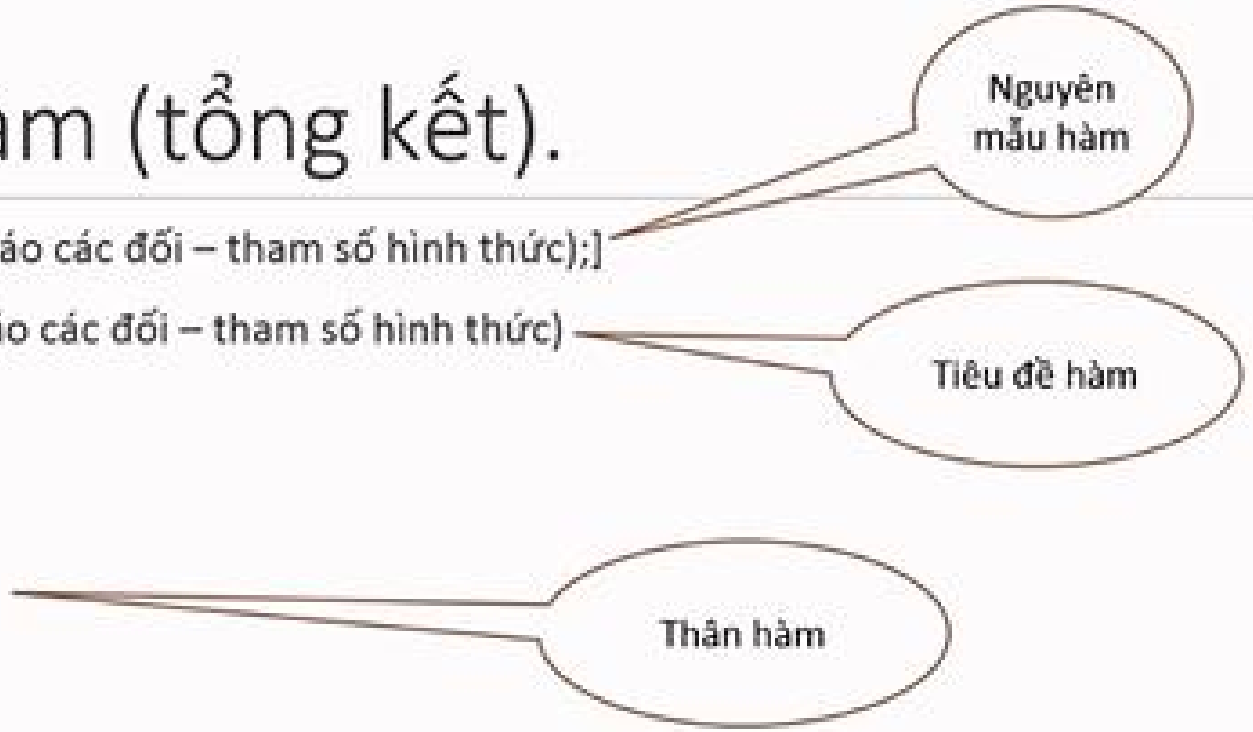
-Trong thân hàm chứa các câu lệnh cần thiết để thực hiện một yêu cầu nào đó đã đề ra cho hàm

- Trong thân hàm có thể sử dụng một câu lệnh return, có thể dùng nhiều câu lệnh return ở những chỗ khác nhau và cũng có thể không sử dụng câu lệnh này. Dạng tổng quát của nó là:

```
return [ biểu thức ];
```

Giá trị của biểu thức trong câu lệnh return sẽ được gán cho hàm.

Xây dựng hàm (tổng kết).



[kiểu_hàm tên_hàm (khai báo các đối – tham số hình thức);]

kiểu_hàm tên_hàm (khai báo các đối – tham số hình thức)

{

khai báo các biến cục bộ

các câu lệnh

[return [biểu thức];]

}

Sử dụng hàm

Hàm được sử dụng thông qua lời gọi tới nó. Cách viết một lời gọi hàm như sau:

tên_hàm ([danh sách tham số thực])

Một điều cần nhớ khi viết lời gọi hàm là:

+ số lượng tham số thực phải bằng số lượng tham số hình thức (đối)

+ mỗi tham số thực phải có cùng kiểu giá trị như kiểu giá trị của đối tương ứng với nó.

Ví dụ:

sqrt(100)

sqrt(3*x)

pow(d, 1.0/k)

Một số lưu ý về biến cục bộ (biến trong hàm) và đối

- Do đối và biến cục bộ đều có phạm vi hoạt động trong cùng một hàm nên đối và biến cục bộ cần có tên khác nhau.

- Đối và biến cục bộ đều là các biến tự động. Chúng được cấp bộ nhớ khi hàm được xét đến và chúng sẽ lập tức bị xoá trước khi ra khỏi hàm. Như vậy, không thể mang giá trị của đối ra khỏi hàm.

- Đối hoặc biến cục bộ có thể trùng tên với bất kỳ đại lượng nào ở ngoài hàm mà không gây ra một nhầm lẫn nào cả

Ví dụ: xây dựng, sử dụng hàm, tổ chức chương trình

Bài 1: Xây dựng hàm tính n! và sử dụng tính một số giai thừa

Bài 2: Xây dựng hàm nhận vào hai số nguyên dương a và b, nhập một trong các phép toán pt: +, -,* , /. ^. Hàm trả ra một số là kết quả của biểu thức a pt b.

Sử dụng hàm trong chương trình, cho phép nhập 2 số nguyên bất kỳ cùng với 1 phép toán, tính và in ra màn hình giá trị của phép toán giữa và b.

Bài 3: Xây dựng hàm tính giá trị lớn nhất của 3 số thực? Trong chương trình chính sử dụng hàm để in ra giá trị lớn nhất của 3 số thực bất kỳ?

Bài 4: Xây dựng hàm tính khoảng cách giữa 2 điểm trong xOy. Sử dụng hàm trong chương trình chính để tính khoảng cách giữa 2 điểm bất kỳ?

Bài 5: Xây dựng hàm tính diện tích của tam giác khi biết tọa độ 3 đỉnh dựa trên hàm tính khoảng cách ở bài 4. Sử dụng trong chương trình để tính diện tích tam giác cho 3 điểm?

Bài 6: Xây dựng hàm giải phương trình bậc 2? Sử dụng để giải phương trình bậc 2 bất kỳ?

Dùng dẫn hướng #define để định nghĩa hàm đơn giản

Ví dụ 1:

```
#include <stdio.h>

#define tong(x, y) x + y

main()
{
   int a = 5, b = 8;
   printf("%d + %d = %d",a, b, tong(a, b));
}
```

Tổ chức hàm thành "thư viện"

Ta có thể viết các hàm trên các file khác với file chương trình

Trong chương trình #include "tên file"

Quy tắc hoạt động của hàm

Khi gặp một lời gọi hàm thì hàm bắt đầu được thực hiện. Nói cách khác, khi máy gặp một lời gọi hàm ở một chỗ nào đó của chương trình, thì máy sẽ tạm rời chỗ đó và chuyển đến hàm tương ứng.

Quá trình thực hiện hàm sẽ diễn ra theo trình tự 4 bước như sau:

a/ Cấp phát bộ nhớ cho các đối và các biến cục bộ.

b/ Gán giá trị của các tham số thực cho các đối tương ứng.

c/ Thực hiện các câu lệnh trong thân hàm.

d/ Khi gặp câu lệnh return hoặc dấu } cuối cùng của thân hàm thì máy sẽ xoá các đối, các biến cục bộ (giải phóng bộ nhớ của các đối, biến cục bộ) và thoát khỏi hàm.

Nếu trở về từ một câu lệnh return có chứa biểu thức thì giá trị của biểu thức được gán cho hàm. Giá trị của hàm sẽ được sử dụng trong các biểu thức chứa nó.

Nguyen Duc Du 13:49

Bài 1: Xây dựng hàm tính n! và sử dụng tính một số giai thừa

Bài 2: Xây dựng hàm nhận vào hai số nguyên dương a và b, nhập một trong các phép toán pt: +, -,* , /. ^. Hàm trả ra một số là kết quả của biểu thức a pt b.

Sử dụng hàm trong chương trình, cho phép nhập 2 số nguyên bất kỳ cùng với 1 phép toán, tính và in ra màn hình giá trị của phép toán giữa và b.

Bài 3: Xây dựng hàm tính giá trị lớn nhất của 3 số thực? Trong chương trình chính sử dụng hàm để in ra giá trị lớn nhất của 3 số thực bất

Bài 4: Xây dựng hàm tính khoảng cách giữa 2 điểm trong xOy. Sử dụng hàm trong chương trình chính để tính khoảng cách giữa 2 điểm bất kỳ?

Bài 5: Xây dựng hàm tính diện tích của tam giác khi biết tọa độ 3 đỉnh dựa trên hàm tính khoảng cách ở bài 4. Sử dụng trong chương trình để tính diện tích tam giác cho 3 điểm?

Bài 6: Xây dựng hàm giải phương trình bậc 2? Sử dụng để giải phương trình bậc 2 bất kỳ?

Bài 7: Xây dựng hàm kiểm tra một số n có là số nguyên tố hay không? Sử dụng trong chương trình

Bài 8: Viết hàm kiểm tra số hoàn hảo.

Nhập vào một dãy n số và thông báo có phải số nguyên tố, số hoàn hảo hay không?

Bài 9: Xây dựng hàm tính số Fibonacci

Bài 10: Viết các hàm cho các bài tập 1.

Bài 11: Xây dựng hàm tính căn bậc 2