KỸ THUẬT LẬP TRÌNH C

Chương 5: Hàm

1. Khái niệm
2. Khai báo hàm
3. Lời gọi hàm
4. Nguyên tắc hoạt động của hàm
5. Biến cục bộ và biến toàn cục
6. Hàm kiểu void
7. Các cách truyền tham số
8. Nguyên mẫu hàm

1. Khái niệm

Hàm là một đoạn chương trình độc lập nhằm thực hiện trọn vẹn một công việc nhất định. Chúng thực chất là những đoạn chương trình nhỏ giúp chia nhỏ một vấn đề lớn.

2. Khai báo hàm

Cú pháp:

<Kiểu hàm> <Tên hàm> (<Danh sách tham số hình thức>) {

    <Khai báo thêm các biến>

    <Các câu lệnh>

    return <biểu thức>;

}

Trong đó:

- <Kiểu hàm> có thể là một kiểu dữ liệu nào đó (char, int, float, double, . . .) hoặc là kiểu void.
- <Tên hàm> bắt buộc phải có đối với mọi hàm.
- <Danh sách tham số hình thức> có thể có hoặc không tùy thuộc ta định dùng hàm đó làm gì.

- Phần bao trong cặp dấu ngoặc {} được gọi là thân hàm, dấu {} là bắt buộc.
- Khi cần thêm một số biến thì phải <Khai báo thêm các biến>, các biến này gọi là **biến cục bộ của hàm**.
- <Các câu lệnh> là phần thực hiện nhiệm vụ của hàm.
- Câu lệnh return <biểu thức> có thể có hoặc không, khi kiểu hàm không phải là void thì nó bắt buộc phải có. Câu lệnh này nhằm trả về giá trị cho nơi gọi hàm.

3. Lời gọi hàm

Hàm được sử dụng thông qua lời gọi tới nó. **Số lượng và kiểu tham số thực** trong lời gọi hàm phải tương ứng với **số lượng và kiểu tham số hình thức** trong khai báo hàm.

Cú pháp:

<Tên hàm> (<Danh sách tham số thực>)

4. Nguyên tắc hoạt động của hàm

Bước 1: Cấp phát bộ nhớ cho các tham số hình thức và các biến cục bộ (nếu có).

Bước 2: Truyền giá trị các tham số thực cho các tham số hình thức tương ứng.

Bước 3: Thực hiện các câu lệnh trong thân hàm.

Bước 4: Khi gặp câu lệnh return hoặc dấu } cuối cùng của thân hàm thì máy sẽ giải phóng các tham số hình thức, các biến cục bộ và thoát khỏi hàm

Ví dụ:

```
/*Hàm trả về giá trị max của hai số thực*/
float Max(float x, float y) {
        float ret; //Khai báo biến cục bộ ret
       ret = (x > y ? x : y);
       return ret;
}
 void main() {
        float a = 3.6, b = 7.2;
        float kq = Max(a, b);  /*Lời gọi hàm Max() với hai
                                  tham số thực a và b*/
       <xuất kq> }
```

5. Biến cục bộ và biến toàn cục

- Biến cục bộ là biến được khai báo bên trong một hàm. Biến này chỉ có thể được truy cập bên trong hàm mà nó khai báo và chỉ tồn tại khi hàm được gọi tới.
- Biến toàn cục là biến được khai báo bên ngoài các hàm. Biến này có thể được truy cập ở mọi nơi trong CT và tồn tại trong suốt thời gian CT thực hiện.

Ví dụ:

```
float a, b; //Khai báo 2 biến toàn cục a và b
/*Hàm trả về giá trị lớn nhất của hai số thực (hàm không có tham số hình thức - đối số)*/
 float Max() {
      float ret; //Khai báo biến cục bộ ret
      ret = (a > b ? a:b);
      return ret;}
void main() {
       float kq;
      a = 3.6; b = 7.2;
      kq = Max();  //Lời gọi hàm Max() không tham số thực
      printf("max (%0.2f, %0.2f) = %0.2f\n", a, b, kq);
}
```

6. Hàm kiểu void

Khi một hàm không trả về một giá trị nào, hàm đó được gọi là hàm kiểu void.

Ví dụ:

```
void Xuat(int x, int y) {
    printf("%d, %d\n", x, y);
}
void main() {
    int i;
    for(i = 1; i <= 10; i++)
        Xuat(i, 2*i); /*Lời gọi hàm Xuat() với hai tham
                        số thực i và 2*i */
}
```

7. Truyền tham số cho hàm

a. Truyền tham trị

- **Giá trị** của các tham số thực được truyền cho các tham số hình thức tương ứng của hàm.
- Mọi thay đổi trong hàm trên các tham số hình thức sẽ không ảnh hưởng tới giá trị ban đầu của của các tham số thực.

Ví dụ:

```
//Hàm hoán vị hai số nguyên
void HoanVi(int a, int b) {
        int tam; //khai báo biến cục bộ tam
        tam = a; a = b; b = tam;
}
void main() {
        int x = 3, y = 7;
        HoanVi(x, y); /*Lời gọi hàm HoanVi với hai tham số
                                thực x và y*/
        <In x, y>; //x và y không thay đổi
}
```

b. Truyền tham trỏ

- **Địa chỉ** của các tham số thực (có dấu & đặt trước) được truyền cho các tham số hình thức tương ứng của hàm và các tham số hình thức phải được khai báo dưới dạng **con trỏ** (có dấu * đặt trước).
- Bên trong hàm có thể truy nhập trực tiếp đến vùng nhớ các tham số thực thông qua các con trỏ này.

Ví dụ:

```
/*Hàm hoán vị hai số nguyên */
void HoanVi(int *a, int *b) {
        int tam;
        tam = *a; *a = *b; *b = tam;
}
void main() {
        int x = 3, y = 7;
        HoanVi(&x, &y); /*Lời gọi hàm HoanVi() với hai
                                tham số thực &x và &y*/
        <In x, y>; //x và y đã bị thay đổi
}
```

8. Nguyên mẫu hàm

- Về nguyên tắc khi gọi một hàm thì hàm đó phải được khai báo trước, nếu không chương trình sẽ báo lỗi. Tuy nhiên cũng có thể gọi một hàm chưa được khai báo trước bằng cách khai báo trước nguyên mẫu hàm.
- Nguyên mẫu hàm thực chất là dòng đầu của hàm có thêm dấu chấm phẩy.

Ví dụ:

```
void HoanVi(int *a, int *b); /*Khai báo nguyên
                                mẫu hàm*/
void main() {
      int x = 3, y = 7;
      HoanVi(&x, &y); /*Gọi hàm HoanVi() được khai báo
                        sau*/
      <In x, y>
}
void HoanVi(int *a, int *b) {
      int tam;
      tam = *a; *a = *b; *b = tam;
}
```

Hết