Chương 6: Hàm (function)

Agenda

- Khái niệm
- Dạng tổng quát của hàm
- Tham số

1. Khái niệm

- Hàm là một khối lệnh được đặt tên và thực hiện một tác vụ nào đó. Hàm được thực thi (execute) khi nó được gọi từ 1 điểm nào đó trong chương trình.
- Hàm còn được gọi là chương trình con (subroutine)
- Hàm có thể có giá trị trả về hoặc không. Nếu hàm không có giá trị trả về, hàm có thể được gọi là thủ tục (procedure).

1. Khái niệm

- Có 2 loại hàm
  - Hàm thư viện: là những hàm đã được xây dựng sẵn. Muốn sử dụng các hàm thư viện phải khai báo thư viện chứa nó trong phần khai báo #include.
    - Ví dụ: sqrt(), pow(), getch(), …
  - Hàm do người dùng định nghĩa.

2. Dạng tổng quát của hàm

- Hàm do người dùng định nghĩa có dạng tổng quát như sau:

```
returnType functionName (parameterList)
{
    body of the function
}
```

  - **returnType**: Kiểu dữ liệu của kết quả trả về của hàm. Nếu hàm không trả về giá trị thì returnType = void.

2. Dạng tổng quát của hàm

- Ví dụ 1:

```
#include <iostream>
int addition (int a, int b)
{
    return (a+b);
}
int main ()
{
    int z;
    z = addition (5,3);
    cout << "The result is " << z;
    return 0;
}
```

2. Dạng tổng quát của hàm

- Ví dụ 1:

```
#include <iostream>
int addition (int a, int b) ; // prototype of function

int main ()
{
	int z;
	z = addition (5,3);
	cout << "The result is " << z;
	return 0;
}
int addition (int a, int b)
{
	int r;
	r=a+b;
	return (r);
}
```

2. Dạng tổng quát của hàm

- Ví dụ 2: Hàm không trả về giá trị

```
#include <iostream>
void  printMsg ( ) ; // prototype of the function

int main ()
{
      printMsg();
      return 0;
}
void  printMsg ( )
{
      cout<<“Hello world !!!”;
}
```

2. Dạng tổng quát của hàm

- Ví dụ:
  - Viết chương trình nhập điểm toán, lý, hóa của học sinh. Viết hàm tính điểm trung bình.

2. Tham số hình thức & tham số thực

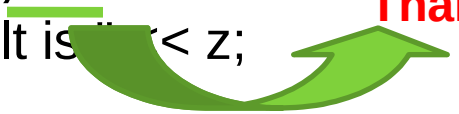
- Tham số hình thức:
  - Khi hàm cần nhận đối số (*arguments) để* thực thi thì khi khai báo hàm cần khai báo **danh sách các tham số** để nhận giá trị từ chương trình gọi. Các tham số này được gọi là ***tham số hình thức.***

```
int addition (int a, int b)
{
    int r;
    r=a+b;
    return (r);
}
```

3. Tham số hình thức & tham số thực

- Tham số thực:
  - Khi gọi hàm, ta cung cấp các giá trị thật, các giá trị này sẽ được sao chép vào các tham số hình thức và các giá trị thật được gọi là **tham số thực**

```
int addition (int a, int b) ; // prototype of function
int main ()
{
      int z;
      z = addition (5,3);
      cout << "The result is " << z;
      return 0;
}
int addition (int a, int b)
{
      return (a+b);
}
```



4. Truyền tham số trong hàm

- Có 2 cách truyền tham số trong hàm: Truyền tham trị & truyền tham biến.
- **4.1 Truyền tham trị:**
  - Truyền giá trị của đối số vào tham số hình thức.
  - Những thay đổi của tham số không làm ảnh hưởng đến đối số.
  - Tham số hình thức được khai báo trong phần khai báo hàm là biến thông thường, không phải là biến con

4.1 Truyền tham trị

- Ví dụ

```
#include <iostream.h>
void doubleNum(int a); //prototype
void main()
{
      int a=40;
      doubleNum(a);
      cout << “Inside main function:” << endl;
      cout << “a = “ << a << endl;
}
void doubleNum(int a)
{
      a = a*2;
      cout << “Inside doubleNum function. a = “ <<
      a;
}
```

4.1 Truyền tham chiếu

- Truyền địa chỉ của đối số vào tham số hình thức
- Những thay đổi của tham số sẽ ảnh hưởng đến đối số.
- Ví dụ 1

```
#include <iostream.h>
void doubleNum(int *b); //prototype
void main()
{
	int a=40;
	doubleNum(&a);
	cout << “Inside main function:” << endl;
	cout << “a = “ << a << endl;
}
void doubleNum(int *b)
{
	*b = *b + *b;
	cout << “Inside doubleNum function.a = “ << *b;
}
```

4.1 Truyền tham chiếu

- Truyền địa chỉ của đối số vào tham số hình thức
- Những thay đổi của tham số sẽ ảnh hưởng đến đối số.
- Ví dụ 2

```
#include <iostream.h>
void doubleNum(int &x); //prototype
void main()
{
	int a=40;
	doubleNum(a);
	cout << “Inside main function:” << endl;
	cout << “a = “ << a << endl;
}
void doubleNum(int &x)
{
	x = x + x;
	cout << “Inside doubleNum function. a = “ <<x ;
}
```

5. Truyền mảng vào hàm

- Khi mảng được dùng như 1 đối số để truyền cho hàm, **địa chỉ của mảng được vào tham số hình thức của hàm.**
- Truyền mảng vào hàm mặc định là truyền tham chiếu ➔ Những thay đổi đến giá trị của các phần tử mảng trong thân hàm sẽ ảnh hưởng đến mảng gốc

5. Truyền mảng vào hàm

- Ví dụ: Viết chương trình thay đổi các phần tử trong mảng: nếu giá trị >=0 thì thay bằng 1, ngược lại thay bằng 0.

```
#include <iostream.h>
void change(int a[], int elements); //prototype
void main()
{
	int arr[] = {5, -5, -3, 3, 7, -7};
	change(arr,6);

	for(int i=0 ; i<6 ; i++)
	cout << “arr[“ <<i<< “] = “ << arr[i] << endl;
}
void change(int a[], int elements)
{
	for(int i=0 ; i<elements ; i++)
	if(a[i] < 0) a [i]=0;
	else a[i] = 1;
}
```

Ví dụ:

- 1. Viết hàm kiểm tra 1 số có phải là số nguyên tố hay ko.
- 2. Tìm ước số của 1 số nguyên n.
- 3. Nhập vào 1 mảng số nguyên và 1 số x. Tìm xem x xuất hiện trong mảng bao nhiêu lần.
- 4. Đếm số chữ số trong số nguyên n.
- 5. Nhập vào 1 mảng, xuất ra các số nguyên tố trong mảng.
- 6. Nhập vào 1 mảng. Viết chương trình đảo ngược các phần tử trong mảng.

6. Đối số của hàm main

- Hàm main là điểm bắt đầu của mọi chương trình C/C++. Trong 1 vài trường hợp, ta cần truyền thông tin vào hàm main. Những thông tin này ta gọi là đối số dòng lệnh (command line arguments).
- Hàm main() có 2 tham số:
  - **int argv:** là một biến nguyên giữ số đối số có trong dòng lệnh
  - **char *argc[]:** là một mảng con trỏ char. Mỗi phần tử của mảng này trỏ đến một đối số dòng lệnh. Tất cả đối số dòng lệnh là chuỗi (string).

```
int main(int argc, char *argv[])
{ }
```

6. Đối số của hàm main

- **Ví dụ:**

```
int main(int argc, char *argv[])
{
    if (argc!=2)
    {
        cout << “Hello, “ << argv[1];
        exit(1);
    }
    return 0;
}
```

7. Lệnh return

- Lệnh **return** có 2 cách dùng
  - Cách 1: Kết thúc ngay lập tức hàm chứa nó và trả điều khiển về cho chương trình gọi hàm.
  - Cách 2: Trả giá trị về cho hàm

7.1 Lệnh return (cách 1)

- Có 2 cách để hàm kết thúc sự thực thi
  - Cách 1: Kết thúc bình thường – hàm được thực hiện từ đầu đến cuối.
  - Cách 2: Hàm thực hiện câu lệnh **return**

7.1 Lệnh return (cách 1)

- Ví dụ: In các phân tử của mảng. Khi gặp phần tử có giá trị âm thì dừng lại.

```
#nclude <iostream.h>
void main()
{
      int a[] = {3,2,1,0,-1,-2,-3];
     for(int i=0 ; i<7 ; i++)
     {
          if(a[i] < 0) return;
          cout << setw(5) << a[i];
     }
}
```

7.2 Lệnh return (cách 2)

- Lệnh “return” được dùng để trả về giá trị cho hàm.

```
#include <iostream.h>
int sum(int a,int b)
{
    return (a+b);
}
void main()
{
    int a,b;
    cout<< “Nhap a, b”;
    cin>>a>>b;
    cout<<a  <<“+” <<b<< “ = ”<<sum(a,b);
}
```

8. Đệ qui

- 1 hàm được gọi là đệ qui khi bên trong hàm có 1 lệnh gọi đến chính hàm đó.
- Ví dụ: Viết chương trình tính n! với n! = 1* 2 * 3 * ... * (n-1)