

Tin học cho lớp 11 chuyên Tin
Năm học 2009-2010
Phần II. NHẬP MÔN LẬP TRÌNH THEO NGÔN NGỮ C

Bài 5. Chương trình con: Hàm

- C/C++, Pascal,... là những ngôn ngữ lập trình có cấu trúc: Chương trình có thể chia sẻ ra thành nhiều modul lập trình riêng biệt lắp ghép lại thành một phần mềm hoàn chỉnh.
- Trong Pascal là những Thủ tục và Hàm, chúng không cần biết đến dữ liệu thực sự, mà đều là hình thức cả.
- Còn trong C/C++ chỉ có Hàm!

Trong C/C++ thì chỉ có Hàm, do vậy đôi khi Hàm cũng làm nhiệm vụ như một thủ tục/lệnh. Trong Pascal ta cũng có thể bắt chước cách làm này của C/C++ (!).

1. Nguyên tắc cơ bản xây dựng một Hàm trong C/C++

```
<Kiểu kết quả> <Tên Hàm>(Danh sách các đối số)
{
Khai báo biến cục bộ;
Các câu lệnh làm nhiệm vụ của Hàm;
Return (Biểu thức);
}
```

Chú ý:

Nếu không có kiểu kết quả thì thay vào đó là từ khóa **void**. Nếu không có đối số thì để trống hoặc từ khóa **void**. Thân chương trình chính cũng là một Hàm. Trong C/C++ **không được có Hàm con** trong mỗi Hàm như trong Pascal, nhưng vẫn có Hàm đệ quy!

2. Cấu trúc mã nguồn C/C++ có Hàm

Cách 1 :

- Khai báo các tệp bao Hàm
- Khai báo dữ liệu ngoài, tức là các biến toàn cục;
- Khai báo nguyên mẫu các Hàm (Danh sách tiêu đề các Hàm sẽ sử dụng sau Hàm chính); {Ý nghĩa như Forward ở Pascal}
- Hàm chính main()
- Chi tiết các Hàm khác

Cách 2 (giống Pascal):

- Khai báo các tệp bao Hàm
- Khai báo dữ liệu ngoài, tức là các biến toàn cục;
- Khai báo chi tiết các Hàm
- Hàm chính main()

Ví dụ 1:

Tính diện tích hình chữ nhật theo hai cạnh a và b nhập từ bàn phím. **Hàm đơn trị (tức là chỉ trả lại đúng một giá trị).**

```
0) #include <conio.h>
1) #include <stdio.h>
2) float a,b,s;
3) float DienTich(float,float);
4) void main(void)
5) {
6) clrscr();
```

```

7) printf("Nhap kích thước hình chữ nhật = ");
8) scanf("%f%f", &a, &b);
9) s=DienTich(a,b);
10) printf("Đáp số: Diện tích = %f",s);
11) getch();
12) }
13) float DienTich(float x, float y)
14) {
15) Return (x*y)
16) }

```

Chú ý:

Dòng 2 là các biến toàn cục

Dòng 3 là nguyên mẫu Hàm **DienTich**, tên các biến có thể bỏ đi, phải kết thúc bằng dấu chấm phẩy;

Dòng 9 là Hàm chính gọi Hàm con **DienTich**.

Dòng 13-14-15 là xây dựng Hàm **DienTich**.

Chữ **void** nghĩa là không có giá trị! Nếu trong Hàm main không có kiểu void thì cuối Hàm đó phải có lệnh **return (0);** hay **return 0;**

Ví dụ 1 cho ta hình mẫu rất đơn giản về một Hàm và cả mã nguồn có mặt nó. Hàm này là Hàm trả lại cho ta một giá trị kiểu số thực. Vì trả lại 1 giá trị, nên gọi là Hàm **Đơn trị**. Tuy nhiên đôi khi ta lại muốn bắt nó trả lại nhiều giá trị hơn, chẳng hạn nhập 3 cạnh của tam giác mà lại cho cả chu vi, diện tích, độ dài đường cao, trung tuyến, phân giác, bán kính đường tròn nội, ngoại và bàng tiếp,, Hàm trả lại nhiều giá trị như vậy gọi là Hàm **Đa trị**. Sau đây là cách xây dựng một Hàm đa trị: Nghiên cứu kỹ ví dụ sau:

Ví dụ 2:

Tính chu vi và diện tích của hình chữ nhật có kích thước là a,b. **Hàm đa trị (tức là trả lại đúng nhiều hơn một giá trị).**

```

#include <conio.h>
#include <stdio.h>
float a,b,p,s;
void ChuViVaDienTich(float,float,float *,float *);
main()
{
clrscr();
printf("Nhap kích thước hình chữ nhật = ");
scanf("%f%f", &a, &b);
ChuViVaDienTich(a,b, &p, &s)
printf("Đáp số: Chu vi = %f, Diện tích = %f",p,s);
getch();
return 0;
}
void ChuViVaDienTich(float x,float y,float *cv,float *dt)
{
*cv=2*(x+y);
*dt=x*y;
}

```

Chú ý:

Với Hàm đa trị thì không trả lại 1 giá trị, nên dùng **void** cho kiểu giá trị. Nó sẽ trả lại giá trị ở trong bộ nhớ tại các địa chỉ do các con trỏ trở vào. Hàm có bao nhiêu "trị" thì có bấy nhiêu con trỏ tương ứng. Không gán nó cho một biến nào!

So với Pascal thì các con trỏ này ví như các **tham biến** trong việc xây dựng chương trình con còn lại như tham trị... Vậy đó!

Ví dụ 3:

Đổi chỗ hai giá trị của hai biến x và y.

```
#include <conio.h>
#include <stdio.h>
float a,b;
void DoiCho(float *,float *);
main()
{
clrscr();
printf("Nhap hai so thuc a, b = ");
scanf("%f%f",&a,&b);
DoiCho(&a,&b);
printf("Sau khi doi cho a = $f, b= %f",a,b);
getch();
return 0;
}
void DoiCho(float *x,float *y)
{
float phu;
phu=*x;
*x=*y;
*y=phu;
}
```

Kiểu của Hàm đơn trị không những là các số mà còn có thể là các kiểu khác nữa, thậm chí còn là kiểu con trỏ nữa. Ví dụ sau đây minh họa bài toán: Tìm xem trong một dãy số nguyên cho trước có số nào bằng số x nhập từ bàn phím hay không?

Ví dụ 4:

Tìm xem trong một dãy số nguyên a cho trước có số nào bằng số x?

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
int a[10],x; /*Ta lay ví dụ mang co 10 phan tu*/
int *TimKiem(int *,int,int);
main()
{
int i,*p;
clrscr();
randomize();
for (i=0;i<10;i++)
a[i]=random(100);
printf("Day so nhap tu dong là:\n");
for (i=0;i<10;i++)
printf("a[%d]=%d, ",i,a[i]);
printf("\n");
printf("Nhap so x = ");
scanf("%d",&x);
p=TimKiem(a,10,x); /*Tìm trong 10 so hang cua mang a co so x khong?*/
if (p!=NULL)printf("Co thay phan tu x trong day.");
else printf("Khong co phan tu nào trong day bang x ca!");
getch();
return 0;
}
int *TimKiem(int *a,int n,int x)
/* Tìm trong mảng a, ở n phần tử đầu tiên, xem có phần tử nào có giá
```

```

trị bằng x không*/
{
int i=0, thay=0, *q;
q=a;
while (i<n && thay==0)
{
if (*q==x)
{
thay=1;
break;
}
q++;
i++
}
if (thay==1) return (q);
else return (NULL);
}

```

Chú ý:

NULL tương tự như nil của Pascal, không trở vào đâu cả!

3. Một số Chú ý khi sử dụng C++

- Dòng chú thích: có thêm dấu // nhưng chỉ trên 1 dòng và ở cuối dòng.
- Ép kiểu linh hoạt hơn: Trong C phải là (<kiểu><biến>, thì trong C++ có thể là <kiểu>(<biến>).
- Phải có Tiền bao Hàm đầy đủ, nếu không có thể sai kết quả, mặc dù không báo lỗi.
- Xuất dữ liệu có thể dùng

```
cout << biểu thức1 << biểu thức 2 << .. << biểu thức;
```

e) Nhập cũng vậy:

```
cin >> biến 1 >> biến 2 >> ... >> biến n;
```

Sau khi nhập như vậy, nó tự động xuống dòng. Nếu muốn không xuống dòng thì dùng lệnh:

```
cin.ignore(1);
```

Muốn nhập n kí tự vào biến xâu kí tự ta có thể dùng lệnh

```
cin.get(<biến xâu>,n);
```

Tuy nhiên nhớ phải có tiền báo Hàm

```
#include <iostream.h>
```

Muốn viết k chữ số sau dấu phẩy thập phân của số thực, ta dùng hai lệnh:

```
cout << setiosflags(ios::showpoint) << setprecision(k);
```

Muốn viết tổng số k vị trí của biến, ta dùng:

```
cout << setw(k) << biến 1 << Biến 2;
```

f) Việc khai báo các biến địa phương có thể đặt ở bất cứ đâu, trước khi sử dụng chúng.

4. Bài tập:

Trong các Bài tập sau phải lập chương trình có các Hàm phục vụ cho Hàm chính

- Tìm USCLN và BSCNN của 2 số a,b
- Rút gọn căn thức bậc hai, nhờ các Hàm trên.
- Rút gọn một phân số.
- Giải gần đúng phương trình $\sin(x)=x-3$;

Như trong Ví dụ về cộng các số lớn dạng xâu kí tự ta thấy đối số của Hàm cũng có thể là mảng 1 chiều, khi đó hiển nhiên nó có kiểu con trỏ rồi., không cần phải có * gì cả!.

Đôi số của Hàm cũng có thể là mảng 2 chiều. Sau đây là một vài ví dụ đơn giản:

Ví dụ 5:

Nhập mảng 2 chiều rồi tìm phần tử max, min của nó. Không cần tìm tất cả, và không cần đưa ra chỉ số của các phần tử đó.

```
#include <stdio.h>
#include <conio.h>
int b[10][20],m,n,max,min;
void MaxMin(int a[10][20],int m,int n,int *max,int *min)
{
int i,j;
*max=*min=a[0][0];
for (i=0;i<m;i++)
    for (j=0;j<n;j++)
    {
    if (a[i][j]>*max) *max=a[i][j];
    if (a[i][j]<*min) *min=a[i][j];
    }
}
void NhapMang(int a[10][20],int *m,int *n)
{
int i,j;
printf("Nhap mang:\n");
printf("So hang va so cot: m,n=");
scanf("%d%d",m,n);
for (i=0;i<*m;i++)
    for (j=0;j<*n;j++)
    {
    printf("a[%d][%d]=",i,j);
    scanf("%d",&a[i][j]);
    }
}
void XuatMang(int a[10][20],int m,int n)
{
int i,j;
printf("Mang hien thoi la:\n");
for (i=0;i<m;i++)
    {
    for (j=0;j<n;j++)
    printf("%6d",a[i][j]);
    printf("\n");
    }
}
void main()
{
do
    {
    clrscr();
    NhapMang(b,&m,&n);
    XuatMang(b,m,n);
    MaxMin(b,m,n,&max,&min);
    printf("Ds: Max=%d, Min=%d",max,min);
    }
while (getch()!=27);
}
```

Chú ý:

Trở lại vấn đề mảng hai chiều $a[i][j]$, ngoài việc khai báo kiểu “truyền thống”

```
int a[20][30],m,n; //m=20 là số hàng, n=30 là số cột
```

Ta cũng có thể khai báo mảng hai chiều một cách “hiện đại”, chỉ dùng 1 con trỏ:

```
int *p,m,n;
```

Khi đó $a+i$ là con trỏ nhảy qua i bước, mỗi bước có độ dài (số Bytes) bằng cỡ của khiểu thành phần. Cụ thể là, ta có thể đồng nhất

```
a[i][j] = *(p+i*n+j)
```

và cũng vậy

```
&a[i][j] = p+i*n+j.
```

Thế thì bài toán trên có thể viết lại hoàn toàn mới **không hề có biến mảng** như sau:

Ví dụ 6:

```
#include <stdio.h>
#include <conio.h>
int *p,m,n,max,min;
void MaxMin(int *q,int m,int n,int *max,int *min)
{
    int i,j;
    *max=*min=*q;
    for (i=0;i<m;i++)
        for (j=0;j<n;j++)
        {
            if (*(q+n*i+j)>*max) *max=*(q+n*i+j);
            if (*(q+n*i+j)<*min) *min=*(q+n*i+j);
        }
}
void NhapMang(int *p,int *m,int *n)
{
    int i,j;
    printf("Nhap mang:\n");
    printf("So hang va so cot: m,n=");
    scanf("%d%d",m,n);
    for (i=0;i<*m;i++)
        for (j=0;j<*n;j++)
        {
            printf("a[%d,%d]=",i,j); //vẫn giả vờ viết a[i,j] như thường
            scanf("%d",&p+n*i+j);
        }
}
void XuatMang(int *p,int m,int n)
{
    int i,j;
    printf("Mang hien thoi la:\n");
    for (i=0;i<m;i++)
    {
```

```

    for (j=0;j<n;j++)
printf("%6d",*(p+n*i+j));
    printf("\n");
}
}
void main()
{
do
{
clrscr();
NhapMang(p,&m,&n); //Không viết &p vì p đã là con trỏ, là địa chỉ
XuatMang(p,m,n);
MaxMin(p,m,n,&max,&min);
printf("Ds: Max=%d, Min=%d",max,min);
}
while (getch() !=27);
}

```

Nhìn lại xâu kí tự, với bản chất là mảng một chiều của các kí tự tận cùng bằng kí tự có mã là 0, ta có thể thao tác đơn giản hơn khi lập trình

Ví dụ 7:

Nhập năm sinh âm lịch, tính ra Can-Chi

```

#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
int nam;
char *can,*chi;
do
{
clrscr();
printf("Vao nam = ");
scanf("%d",&nam);
switch((nam-4)%10)
{
case 0:can="Giap"; break;
case 1:can="At"; break;
case 2:can="Binh"; break;
case 3:can="Dinh"; break;
case 4:can="Mau"; break;
case 5:can="Ky"; break;
case 6:can="Canh"; break;
case 7:can="Tan"; break;
case 8:can="Nham"; break;
case 9:can="Quy"; break;
}
switch((nam-4)%12)
{
case 0:chi="Ti"; break;
case 1:chi="Suu"; break;
case 2:chi="Dan"; break;
case 3:chi="Mao"; break;
case 4:chi="Thin"; break;
case 5:chi="Ty"; break;
}
}
}

```

```

    case 6:chi="Ngo"; break;
    case 7:chi="Mui"; break;
    case 8:chi="Than"; break;
    case 9:chi="Dau"; break;
    case 10:chi="Tuat"; break;
    case 11:chi="Hoi"; break;
}
printf("Do la nam %s-%s\n",can,chi);
}
while (getch() !=27);
}

```

Sau đây lại thêm một ví dụ nữa về mảng các chuỗi ký tự, dùng các Hàm có đối số là mảng 2 chiều:

Ví dụ 8:

Nhập một dãy chuỗi ký tự, rồi sắp xếp chúng lại theo thứ tự từ điển

```

#include <stdio.h>
#include <conio.h>
#include <string.h>
char s[15][25];
int m;
void NhapMangXau(char s[][25],int *);
void SapXep(char s[][25],int); //sap xep m xau theo thutu Tu dien
void XuatMangXau(char s[][25],int);
main()
{
int i;
do
{
clrscr();
NhapMangXau(s,&m);
SapXep(s,m); //sap xep m xau theo thutu Tu dien
printf("\n");
XuatMangXau(s,m);
}
while (getch() !=27);
return 0;
}
void NhapMangXau(char s[][25],int *m)
{
printf("So cac xau=");
scanf("%d*c",m);
for (int i=0;i<*m;i++)
{
printf("Xau thu %d la: ",i);
scanf("%s",s[i]);
}
}
void SapXep(char s[][25],int m) //sap xep m xau theo thutu Tu dien
{
char phu[25];
int i,j;
for (i=0;i<m-1;i++)
for (j=i+1;j<m;j++)
if (strcmp(s[i],s[j])>0)
{

```

```

    strcpy(phu, s[i]);
    strcpy(s[i], s[j]);
    strcpy(s[j], phu);
}
}
void XuatMangXau(char s[][25], int m)
{
printf("Mang xau hien thoi la:\n");
for (int i=0; i<m; i++)
{
printf("Xau thu %d la: %s\n", i, s[i]);
}
}

```

Cuối bài này, xin được phổ biến thêm về các phép toán trên bit, được tiến hành trên từng bit một...

Đó là các phép:

& (và) $2 \& 3 = 0010 \& 0011 = 0010 = 2$
 | (hoặc) $2 | 3 = 0010 | 0011 = 0011 = 3$
 ^ (hoặc loại trừ) $2 \wedge 3 = 0010 \wedge 0011 = 0001 = 1$
 ~ (lấy phần bù) $\sim 3 = \sim 0011 = 1100 = 12$
 >> (dịch chuyển phải) $15 \gg 3 = 1111 \gg 3 = 0001 = 1$ (như $15/2^3$)
 << (dịch chuyển trái) $15 \ll 3 = 1111 \ll 3 = 1111000 = 120$ (như $15 * 2^3$)

Chú ý: Các phép && và || là phép and và or của hai mệnh đề.

Ví dụ 9

Vẽ chữ A cơ to, khi coi A là một ma trận điểm 8x8, hay mảng 8 Byte, mỗi Byte gồm 8 bit:

```

00001000
00010100
00100010
00100010
00111110
00100010
00100010
00100010

```

6. Một số bài tập

Bài 1.

Nhập một mảng 2 chiều các số nguyên ngẫu nhiên trong khoảng từ -10 đến +10, xuất mảng dưới dạng cân đối, đẹp mắt, tìm max và liệt kê các phần tử bằng max, tìm min và xuất ra các phần tử bằng min.

Bài 2.

Nhập một mảng 2 chiều các số nguyên ngẫu nhiên trong khoảng từ -10 đến +10, xuất mảng dưới dạng cân đối, đẹp mắt, tìm các điểm yên ngựa và tọa độ của mỗi điểm đó.

Bài 3.

Nhập một mảng 2 chiều các số nguyên ngẫu nhiên trong khoảng từ -10 đến +10, xuất mảng dưới dạng cân đối, đẹp mắt. Đảo lại mảng hàng thành cột và cột thành hàng. Đảo lại thứ tự các hàng. Tính tổng các phần tử mỗi cột... Tìm max của mỗi hàng.