TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN KHOA HỌC MÁY TÍNH

CƠ SỞ LẬP TRÌNH

Mã học phần :841020

Email: quocpt@sgu.edu.vn
Website: https://sites.google.com/site/phantanquoc

CHƯƠNG TRÌNH CON

Khái niệm chương trình con

- Để thuận tiện cho việc viết chương trình người ta thường tách chương trình đó thành các đoạn chương trình nhỏ hơn - mỗi đoạn chương trình nhỏ này giải quyết *trọn vẹn* một công đoạn cụ thể của bài toán; mỗi đoạn chương trình nhỏ đó được gọi là một **chương trình con**.
- Chương trình con trong *C*/*C++* được gọi chung là **hàm**. Hàm trong *C*/*C++* có hai dạng là hàm có giá trị trả về và hàm không có giá trị trả về.
- Từ khái niệm chương trình con, cần lưu ý rằng mỗi hàm chỉ nên thiết kế để giải quyết một chức năng cụ thể của vấn đề bài toán.

Lợi ích của việc sử dụng chương trình con

- Làm cho việc phân tích thiết kế chương trình được thuận lợi.
- Giúp thuận lợi trong việc kiểm thử, nâng cấp, viết tài liệu cho chương trình, rút ngắn thời gian lập trình, tránh lặp lại các đoạn lệnh tương tự nhau,…
- Bản thân ngôn ngữ lập trình *C*/*C*++ đã được hỗ trợ một số hàm; các hàm này được gọi là các hàm chuẩn.
- Tùy theo nhu cầu thực tế mà người lập trình có thể thiết kế thêm các hàm khác; các hàm này được gọi là các hàm tự tạo

Tham số hình thức biến, tham số hình thức trị

- Các biến liệt kê trong các dòng *tiêu đề hàm* thường được gọi là các **tham số**. Hàm có thể không có tham số hoặc có nhiều tham số.
- Các tham số mà ngay trước nó không có chỉ dẫn & thì được hiểu là tham số đó được truyền theo kiểu **tham trị**,
- Các biến mà ngay trước nó có chỉ dẫn & thì được hiểu là tham số đó được truyền theo kiểu **tham biến**.
- Danh sách các tham số ở *lời gọi hàm* được gọi là **tham số hình thức**.

Phân biệt tham số hình thức trị/biến

- Các tham số được truyền theo kiểu tham trị thì những thay đổi trong thân hàm chứa nó sẽ không được giữ lại khi ra khỏi hàm chứa nó;
- Ngược lại, nếu tham số được truyền theo kiểu tham biến thì mọi sự thay đổi trong thân hàm đó sẽ được giữ lại khi ra khỏi hàm đó.

Ví dụ 3-1

```
1. #include<iostream.h> // khai báo thư viện
2. void test1(int a, int b, int &c);// dòng tiêu đề hàm, kết
   thúc bằng dấu ;
3. int main()
4. {
5.    int a=1,b=2,c=3;
6.    test1(a,b,c);// lời gọi hàm
7.    cout<<a<<"     "<<b<<"   "<<c;
8. }
9. void test1(int a, int b, int &c) //dòng tiêu đề hàm, kết
   thúc không có dấu ;
10.   {
11.         a=a+1;
12.         b=b+1;
13.         c=c+a+b;
14.   }
```

void test1(int *a*, int *b*, int &*c*);

- hàm test1 có ba tham số là *a, b, c*; trong đó *a, b* là các tham trị, *c* là tham biến.
- Khi thực hiện ví dụ trên thì kết quả là : 1 2 8

Biến toàn cục và biến địa phương

- Các biến được khai báo trong các hàm - kể cả hàm int main () được gọi là các **biến cục bộ** (biến địa phương) và nó chỉ có ảnh hưởng trong phạm vi thân hàm chứa nó. Khi hàm chứa nó kết thúc thì các biến này cũng mất tác dụng theo.
- Ngược lại, các biến được khai báo ngoài được gọi là các **biến toàn cục**; biến toàn cục ảnh hưởng đến cả chương trình.
- Khi lập trình chúng ta cần tránh tối đa việc sử dụng các biến toàn cục.

Ví dụ 3-2

```
1. #include<iostream.h>
2. void test2(int a, int b);
3. int c;// biến toàn cục
4. int main()
5. {
6.    int a=1,b=7,d=5; // các biến cục bộ
7.    c=3;
8.    test2(a,b);
9.    cout<<a<<"     "<<b<<"   "<<c<<"   "<<d;
10. }
11. void test2(int a, int b)
12. {
13.    c=a+b;
14.    int d=a+b;
15.    a=a+1;
16.    b=b+1;
17. }
```

- Chương trình trên có sử dụng bốn biến *a, b, c, d*; trong đó *c* là biến toàn cục; còn *a, b, d* là các biến cục bộ.
- Khi thực hiện ví dụ trên thì kết quả là: 1 7 8 5

Hàm không có giá trị trả về, hàm có giá trị trả về

- Hàm không có giá trị trả về (trả về kiểu *void*), hàm có giá trị trả về (trả về kiểu khác *void*).
- Nếu hàm cần thiết kế chỉ có một giá trị trả về thì có thể thiết kế hàm theo dạng hàm không có giá trị trả hoặc có giá trị trả về; khi đó ta cần thêm thông tin sau để quyết định việc lựa chọn một trong hai cách trên:
  - Nếu tên hàm không được sử dụng trong các hàm hoặc biểu thức khác nữa thì nên thiết kế theo dạng **hàm không có giá trị trả về**,
  - ngược lại thì nên thiết kế theo dạng **hàm có giá trị trả về**.
  - Hàm được thiết kế theo dạng có giá trị trả về thì trong thân hàm đó phải có lệnh return để trả về giá trị là kết quả cần trả về của hàm đó.

Khi cần thiết kế một hàm có nhiều hơn một giá trị thì có thể giải quyết theo ba cách sau:

- *thứ nhất*, phân rã chức năng của hàm thành nhiều chức năng nhỏ hơn nữa, để mỗi hàm có đúng một chức năng riêng biệt và nó sẽ trả về đúng một giá trị;
- *thứ hai*, trả về nhiều giá trị thông qua kỹ thuật truyền tham biến – cần trả về bao nhiêu giá trị thì sử dụng thêm bấy nhiêu tham biến;
- *thứ ba*, sử dụng biến toàn cục – dùng biến toàn cục để ghi nhận giá trị cần trả về.

Ví dụ 3-3

```
1. #include<iostream.h>
2. int test3 (int a, int b, int c);
3. void test4 (int a, int b, int c);
4. void test5 (int &a, int &b, int &c);
5. int main()
6. {
7.    int a=10,b=20,c=30;
8.    cout<< test3(a,b,c)<<endl;//trả về giá trị 63
9.    test4(a,b,c); // trả về giá trị 63
10.   test5(a,b,c);
11.   cout<<a+b+c<<endl;// trả về giá trị 63; a,b,c
   là các tham biến của test 5
12. }
```

```
13. int test3 (int a, int b, int c)
14. {
15.   a=a+1;
16.   b=b+1;
17.   c=c+1;
18.   return a+b+c;
19. }
```

```
20. void test4 (int a, int b, int c)
21. {
22.   a=a+1;
23.   b=b+1;
24.   c=c+1;
25.   cout<<a+b+c<<endl;
26. }
```

```
27. void test5 (int &a, int &b, int &c)
28. {
29.   a=a+1;
30.   b=b+1;
31.   c=c+1;
32. }
```

Ví dụ 3-4

Viết chương trình tính diện tích của hình vuông khi biết độ dài cạnh, hình chữ nhật khi biết độ dài 2 cạnh, hình tam giác khi biết độ dài 3 cạnh, hình tròn khi biết độ dài bán kính.

Trong ví dụ 3-4, chúng ta sẽ xem xét ba kiểu thiết kế chương trình con: Chương trình con mà các tiêu đề hàm không có tham số nào (Ví dụ 3-4a),

```
1.#include <iostream.h>
2.void dthinhvuong();
3.void dthinhchunhat();
4.void dthinhtamgiac();
5.void dthinhtron();
6.int main()
7.{
8.   dthinhvuong();
9.   dthinhchunhat();
10.  dthinhtamgiac();
11.  dthinhtron();
12.}
```

```
13.void dthinhvuong()
14.{
15. int a;
16. cout<<"\nChuong trinh tinh dien tich
  hinh vuong";
17. cout<<"\nNhap chieu dai canh hinh
  vuong : ";
18. cin>>a;
19. int s=a*a;
20. cout<<"dien tich la : "<<s<<endl;
21.}
```

```
22.void dthinhchunhat()
23.{
24.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
25.}
26.void dthinhtamgiac()
27.{
28.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
29.}
30.void dthinhtron()
31.{
32.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
33.}
```

chương trình con không có giá trị trả về (Ví dụ 3-4b),

```
1.#include <iostream.h>
2.void dthinhvuong(int a);
3.void dthinhchunhat(int a, int b);
4.void dthinhtamgiac(int a, int b, int c);
5.void dthinhtron(int r);
6.int main()
7.{
8.   int a,b,c,r;
9.   dthinhvuong(a);
10.  dthinhchunhat(a,b);
11.  dthinhtamgiac(a,b,c);
12.  dthinhtron(r);
13.}
```

```
14.void dthinhvuong(int a)
15.{
16.cout<<"\nChuong trinh tinh dien tich hinh vuong";
17.cout<<"\nNhap chieu dai canh hinh vuong:";
18.cin>>a;
19.int s=a*a;
20.cout<<"dien tich la : "<<s<<endl;
21.}
```

```
22.void dthinhchunhat(int a, int b)
23.{
24.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
25.}
26.void dthinhtamgiac(int a, int b, int c)
27.{
28.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
29.}
30.void dthinhtron(int r)
31.{
32.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
33.}
```

chương trình con có giá trị trả về (Ví dụ 3-4c).

```
1.#include <iostream.h>
2.int dthinhvuong(int a);
3.int dthinhchunhat(int a, int b);
4.float dthinhtamgiac(int a, int b, int c);
5.float dthinhtron(int r);
```

```
6. int main()
7. {
8.   int a,b,c,r;
9.   cout<<"\nChuong trinh tinh dien tich hinh vuong";
10.  cout<<"\nNhap chieu dai canh hinh vuong: ";
11.  cin>>a;
12.  cout<<dthinhvuong(a);
13.  cout<<dthinhchunhat(a,b);
14.  cout<<dthinhtamgiac(a,b,c);
15.  cout<<dthinhtron(r);
16.}
```

```
17.int dthinhvuong(int a)
18.{
19. return a*a;
20.}
```

```
21.int dthinhchunhat(int a, int b)
22.{
23.//bạn đọc dễ dàng hoàn chỉnh đoạn chương trình này
24.}
25.float dthinhtamgiac(int a, int b, int c)
26.{
27.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình
   này
28.}
29.float dthinhtron(int r)
30.{
31.  //bạn đọc dễ dàng hoàn chỉnh đoạn chương trình
   này
32.}
```

Ví dụ 3-5

Viết chương trình in các số nguyên tố nhỏ hơn hoặc bằng n. Với n là số nguyên dương cho trước.

```
1. #include<iostream.h>
2. #include<math.h> // thư viện chứa các hàm toán
   học
3. void nhap(int &n);
4. void insonguyento(int n);
5. int main()
6. {
7.    int n;
8.    nhap(n);
9.    insonguyento(n);
10. }
```

Ví dụ 3-6

Nhập 3 giá trị ngày, tháng, năm. Hãy viết chương trình tìm ngày kề sau của ngày vừa nhập.

```
1. #include <iostream.h>
2. void ngaymai(int &ngay, int &thang, int &nam);
3. int main()
4. {
5.    int ngay,thang,nam;
6.    cin>>ngay>>thang>>nam;
7.    ngaymai(ngay,thang,nam);
8.    cout<<ngay<<" "<<thang<<" "<<nam;
9. }
```

Ví dụ 3-7

Viết chương trình nhập vào hai phân số a/b và c/d.

Hãy tính tổng của hai phân số; yêu cầu phân số kết quả phải ở dạng tối giản. Ví dụ: 1/2 + 1/6 có kết quả là 2/3.

```
1.  #include <iostream.h>
2.  void   nhapps(int &tu, int &mau);
3.  void   xuatps(int tu, int mau);
4.  void   congps(int  tu1,  int  mau1,int  tu2,  int
    mau2, int &tu, int &mau);
5.  int    uscln(int a, int b);
6.  int  main()
7.  {
8.     int tu1,mau1,tu2,mau2,tu,mau;
9.     nhapps(tu1,mau1);
10.    nhapps(tu2,mau2);
11.    congps(tu1,mau1,tu2,mau2,tu,mau);
12.    xuatps(tu,mau);
13. }
```

BÀI TẬP

BT3-1.

Đếm xem từ 1 đến 1 tỷ có bao nhiêu số nguyên tố? Bao nhiêu số chính phương? Bao nhiêu số hoàn chỉnh? (số hoàn chỉnh là số bằng tổng các ước thực sự của nó, Ví dụ 6, 28 là các số hoàn chỉnh).

BT3-2.

Tìm tất cả các số nhỏ hơn 1 tỷ sao cho nó vừa là số nguyên tố và là số đối xứng. Ví dụ 7, 131, 10001 là các số đối xứng.

BT3-3.

Liệt kê tất cả các số tự nhiên *k* thỏa mãn đồng thời ba điều kiện: k là số có 5 chữ số; *k* là số nguyên tố, *k* là số đối xứng.

BT3-4.

Phân tích số tự nhiên *n* thành tích các số nguyên tố. Ví dụ 90 = 2 * 3 * 3 * 5.

BT3-5.

a.Cho biết một ngày có dạng ngày/tháng/năm là ngày thứ bao nhiêu trong năm?

Ví dụ: Ngày 3/2/2012 là ngày thứ 34 trong năm.

b.Tính xem từ ngày 1/1/1 đến ngày 31/12/2017 có bao nhiêu ngày ?

BT3-6.

Viết chương trình nhập vào vào số nguyên dương n.

a.Tìm chữ số có giá trị lớn nhất của số n.

b.Hãy tìm số nguyên tố gần nó nhất.

BT3-7.

Nhập vào số nguyên dương n.

a.Đếm số lượng số nguyên tố nhỏ hơn hoặc bằng n.

b.Tìm số nguyên tố thứ n (dãy các số nguyên tố: 2,3,5,7,11,...)

BT3-8.

Nhập số tự nhiên *n*, hãy in ra chữ số thứ *n* của dãy vô hạn các số chính phương 149162536496481100121….

BT3-9.

Trong biểu thức: ((((a1?a2)?a3)?a4)?a5) = a6. Hãy điền vào các vị trí dấu ? bởi một trong 4 phép tóan +, - , * , / sao cho giá trị của biểu thức đã cho hai vế là bằng nhau.

Hãy tìm tất cả các lời giải.

BT3-10.

Hãy tìm các số p,q ($0<p<q<n$) sao cho tổng các ước số thực sự của p bằng q và tổng các ước số thực sự của q bằng p.