



TỔ CHỨC MÃ NGUỒN

NHẬP MÔN LẬP TRÌNH

GVHD: Trương Toàn Thịnh

NỘI DUNG

- Giới thiệu
- Cơ chế truyền tham số
- Các vấn đề về hàm trong C/C++
 - Hàm trùng tên
 - Hàm có tham số mặc định
 - Hàm có tham số là hàm
- Tổ chức hàm với nhiều tập tin mã nguồn
- Phạm vi hàm và biến toàn cục với nhiều tập tin mã nguồn

GIỚI THIỆU

- Mã nguồn của phần mềm được phân chia ra các thành phần như sau
 - Các gói mã nguồn: gồm nhiều tập tin phối hợp để thiết lập một hệ thống con của phần mềm (các tập tin cùng gói sẽ lưu cùng thư mục)
 - Các tập tin mã nguồn; gồm một hay nhiều chương trình con để giải quyết một nhóm những công việc có liên quan
 - Các chương trình con: có nhiệm vụ thực hiện một công việc cụ thể độc lập

GIỚI THIỆU

- Ví dụ xây dựng hàm ‘làm tròn số’
- Thư viện `<math.h>` cung cấp hàm `floor()` và `ceil()` nhưng chưa cung cấp hàm làm tròn.

Giá trị x	<code>floor(x)</code>	<code>ceil(x)</code>	Số làm tròn
3.2	3	4	3
3.7	3	4	4
-3.2	-4	-3	-3
-3.8	-4	-3	-4

GIỚI THIỆU

- Ví dụ

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>#include <math.h></code>
3	<code>double round(double);</code>
4	<code>double round(double x){</code>
5	<code>double kq;</code>
6	<code>if(x >= 0)</code>
7	<code>kq = floor(x + 0.5);</code>
8	<code>else</code>
9	<code>kq = -floor(-x + 0.5);</code>
10	<code>return kq;</code>
11	<code>}</code>

Dòng	Mô tả
12	<code>void main(){</code>
13	<code>double a, y;</code>
14	<code>printf("Nhập a: ");</code>
15	<code>scanf("%lf", &a);</code>
16	<code>y = round(a);</code>
17	<code>printf("round(a) = %lf", y);</code>
18	<code>}</code>

GIỚI THIỆU

- Phân tích ví dụ
 - Khai báo hàm: làm theo mẫu

<Kiểu dữ liệu trả về>	<Tên hàm>	<danh sách tham số>
double	round	(double)

- Cài đặt hàm (định nghĩa hàm): thực hiện các lệnh bên trong thân hàm
- Gọi hàm: việc gọi hàm hay sử dụng hàm được thực hiện ở hàm main() hay bất kì hàm nào có nhu cầu.

NỘI DUNG

- Giới thiệu
- Cơ chế truyền tham số
- Các vấn đề về hàm trong C/C++
 - Hàm trùng tên
 - Hàm có tham số mặc định
 - Hàm có tham số là hàm
- Tổ chức hàm với nhiều tập tin mã nguồn
- Phạm vi hàm và biến toàn cục với nhiều tập tin mã nguồn

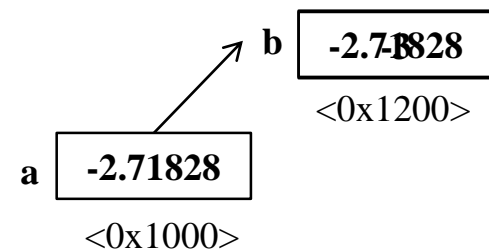
CƠ CHẾ TRUYỀN THAM SỐ

- Tham số của hàm: được gọi là tham số hình thức
- Khi hàm được gọi, thì nơi sử dụng sẽ truyền tham số thực sự vào hàm
- Tham số hình thức và tham số thực có thể trùng tên hay khác tên.
- Việc gửi các tham số thực vào vị trí của tham số hình thức khi sử dụng hàm gọi là cơ chế truyền tham số.

CƠ CHẾ TRUYỀN THAM SỐ

- Cơ chế truyền tham số giá trị (tham trị)
 - Khi gọi hàm thì giá trị tham số thực được sao chép vào tham số hình thức
 - Mọi sự tác động trong hàm lên tham số hình thức (nếu có tác động) sẽ không ảnh hưởng tới giá trị trong tham số thực

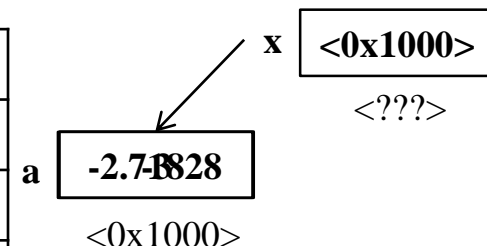
Dòng	Mô tả		
1	<code>void main(){</code>	7	<code>double round(double b){</code>
2	<code>double a = -2.71828, y;</code>	8	<code>double res;</code>
3	<code>y = round(a);</code>	9	<code>if(x>=0) res = floor(b + 0.5);</code>
4	<code>printf("Round is = %lf\n", y);</code>	10	<code>else res = -floor(-x + 0,5);</code>
5	<code>printf("a = %lf\n", a);</code>	11	<code>return res;</code>
6	<code>}</code>	12	<code>}</code>



CƠ CHẾ TRUYỀN THAM SỐ

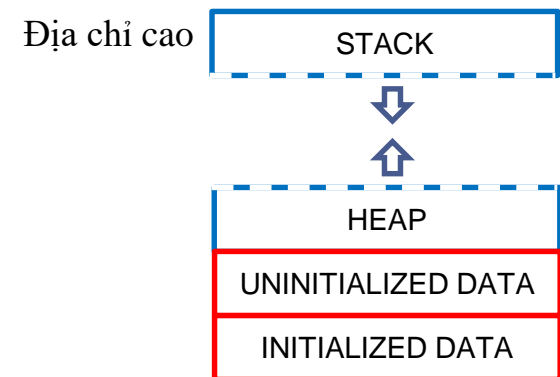
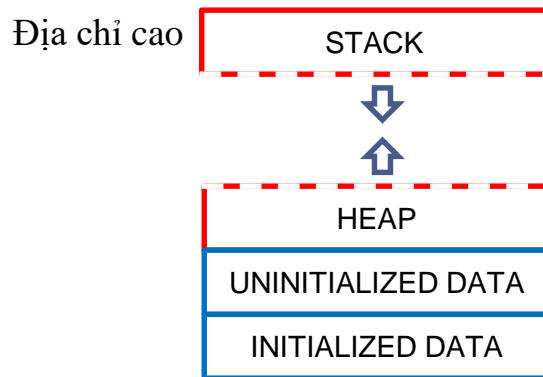
- Cơ chế truyền tham số tham chiếu (tham chiếu)
 - Khi gọi hàm thì tham số thực và tham số hình thức là một
 - Mọi tác động trong hàm lên tham số hình thức (nếu có) đều ảnh hưởng tới tham số thực

Dòng	Mô tả		
1	<code>void round(double);</code>	6	<code>void main(){</code>
2	<code>void round(double &x){</code>	7	<code>double a = -2.71828;</code>
3	<code>if(x >= 0) kq = floor(x + 0.5);</code>	8	<code>round(a);</code>
4	<code>else kq = -floor(-x + 0.5);</code>	9	<code>printf("Round is = %lf\n", a);</code>
5	<code>}</code>	10	<code>}</code>



CƠ CHẾ TRUYỀN THAM SỐ

- Biến cục bộ
 - Được khai báo bên trong của một hàm
 - Dùng để lưu các giá trị tạm thời trong quá trình tính toán của hàm
 - Không cần lưu khi quá trình tính toán kết thúc
 - Khi một hàm được gọi, sẽ có vùng nhớ tạm để cấp phát cho các biến cục bộ



- Biến cục bộ tĩnh:
 - Được cấp trong những vùng nhớ cố định
 - Khai báo bên trong hàm, tuy nhiên giá trị chỉ bị hủy khi **chương trình kết thúc**.

CƠ CHẾ TRUYỀN THAM SỐ

- Ví dụ biến cục bộ tĩnh

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>double Accumulator(double number){</code>
3	<code>static double sum = 0;</code> chỉ gọi 1 lần
4	<code>sum += number;</code>
5	<code>return sum;</code>
6	<code>}</code>
7	<code>void main(){</code>
8	<code>double kq;</code>
9	<code>Accumulator(1);</code>
10	<code>Accumulator(2);</code>
11	<code>kq = Accumulator(3);</code>
12	<code>printf("kq = %lf\n", kq);</code>
13	<code>}</code>

NỘI DUNG

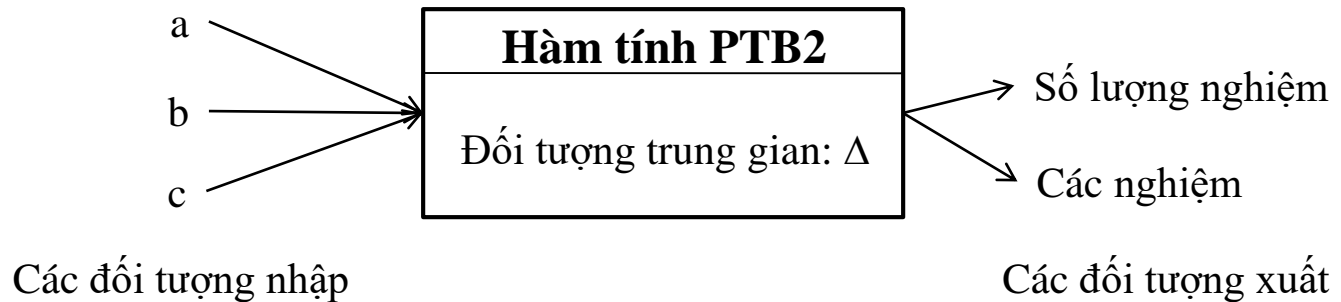
- Giới thiệu
- Cơ chế truyền tham số
- Các vấn đề về hàm trong C/C++
 - Hàm trùng tên
 - Hàm có tham số mặc định
 - Hàm có tham số là hàm
- Tổ chức hàm với nhiều tập tin mã nguồn
- Phạm vi hàm và biến toàn cục với nhiều tập tin mã nguồn

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Dữ liệu liên quan tới hàm được chia làm ba loại
 - Những đối tượng nhập: những đối tượng mà hàm có sẵn
 - Những đối tượng kết xuất: những đối tượng cần xác định hay tính toán
 - Những đối tượng trung gian: sử dụng để lưu các giá trị tạm thời
- Cách cài đặt
 - Đối tượng nhập: thường dùng cơ chế truyền tham trị
 - Đối tượng xuất: có thể dùng cơ chế truyền tham chiếu hay trả về (dùng **return**)
 - Đối tượng trung gian: thường dùng biến cục bộ

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ: viết hàm tính nghiệm phương trình bậc hai một ẩn $ax^2 + bx + c = 0$



CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 1 tính nghiệm phương trình bậc hai

Dòng	Mô tả
1	<code>void EquaDeg2(double a, double b, double c, double& x1, double& x2, int& sn){</code>
2	<code>double delta, sqrtDelta;</code>
3	<code>delta = b*b - 4*a*c;</code>
4	<code>if(delta > 0){</code>
5	<code>sn = 2; sqrtDelta = sqrt(delta);</code>
6	<code>x1 = (-b + sqrtDelta)/(2*a); x2 = (-b - sqrtDelta)/(2*a);</code>
7	<code>}</code>
8	<code>else if(delta == 0){</code>
9	<code>sn = 1; x1 = x2 = (-b)/(2*a);</code>
10	<code>}</code>
11	<code>else {sn = 0;}</code>
12	<code>}</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 2 tính nghiệm phương trình bậc hai

Dòng	Mô tả
1	<code>int EquaDeg2(double a, double b, double c, double& x1, double& x2){</code>
2	<code>double delta, sqrtDelta; int sn ;</code>
3	<code>delta = b*b - 4*a*c;</code>
4	<code>if(delta > 0){</code>
5	<code>sn = 2; sqrtDelta = sqrt(delta);</code>
6	<code>x1 = (-b + sqrtDelta)/(2*a); x2 = (-b - sqrtDelta)/(2*a);</code>
7	<code>}</code>
8	<code>else if(delta == 0){</code>
9	<code>sn = 1; x1 = x2 = (-b)/(2*a);</code>
10	<code>}</code>
11	<code>else {sn = 0;}</code>
12	<code>return sn;</code>
13	<code>}</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 3 kiểm tra số nguyên tố

Dòng	Mô tả
1	<code>int isPrime(long n){</code>
2	<code>int Prime;</code>
3	<code>if(n < 0) n = -n;</code>
4	<code>if(n == 0) Prime = 1;</code>
5	<code>else if(n == 1) Prime = 0;</code>
6	<code>else{</code>
7	<code>long i = 2; Prime = 1;</code>
8	<code>while(i <= sqrt(n)){ tmp =</code>
9	<code>if(n % i == 0) { Prime = 0; break; }</code>
10	<code>}</code>
11	<code>i++;</code>
12	<code>}</code>
13	<code>return Prime;}</code>

n →
(Đối tượng nhập)

Hàm isPrime

Đối tượng trung gian:
sqrt(n) và các biến đếm

↓
Prime (**true** or **false**)
(Đối tượng xuất)

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 4 liệt kê các số nguyên tố nhỏ hơn n


Dòng	Mô tả
1	<code>void PrimeListing(long n){</code>
2	<code>for(long i = 2; i <= n; i++){</code>
3	<code>if(isPrime(i)) printf("%ld\n", i);</code>
4	<code>}}</code>

- Ví dụ 5 in ra số nguyên tố thứ n

Dòng	Mô tả
1	<code>void GetPrime(long n){</code>
2	<code>long p = 2, c = 1, nextNum = 3;</code>
3	<code>while(c < n){</code>
4	<code>if(isPrime(nextNum)){p = nextNum; c++; }</code>
5	<code>nextNum+=2;</code>
6	<code>}}</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 6 xét phương trình $ax + b = 0$

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>#include <math.h></code>
3	<code>#define NoSolution 0</code> 
4	<code>#define Undetermined -1</code>
5	<code>int EqualDeg1(double a, double b, double& x){</code>
6	<code>int nSolution;</code>
7	<code>if(a!=0){ x = -b/a; nSolution = 1; }</code>
8	<code>else{</code>
9	<code>x = 0;</code>
10	<code>if(b == 0) nSolution = Undetermined;</code>
11	<code>else nSolution = NoSolution;</code>
12	<code>}</code>
13	<code>return nSolution; }</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 7 xét phương trình $ax^2 + bx + c = 0$

Dòng	Mô tả
1	...
2	<code>int EqualDeg2(double a, double b, double c, double& x1, double& x2){</code>
3	<code>int nSolution; x1 = x2 = 0;</code>
4	<code>if(a==0) nSolution = EqualDeg1(b, c, x1);</code>
5	<code>else{</code>
6	<code>double delta = b*b - 4*a*c, two_a = 2*a;</code>
7	<code>if(delta < 0) nSolution = NoSolution;</code>
8	<code>else if(delta == 0) { x1 = x2 = -b/two_a; nSolution = 1;}</code>
9	<code>else{</code>
10	<code>double sqrtDelta = sqrt(delta);</code>
11	<code>x1 = (-b-sqrtDelta)/two_a; x2 = (-b+sqrtDelta)/two_a;</code>
12	<code>nSolution = 2; } }</code>
13	<code>return nSolution;}</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 8 xét phương trình $ax^4+bx^2+c=0$ (1)
 - Đặt $y = x^2 \Rightarrow ay^2 + by + c = 0$ (2)
 - Nếu (2) vô nghiệm thì (1) vô nghiệm
 - Nếu (2) có 1 nghiệm $y = y_1 = x^2 \Leftrightarrow x^2 - y_1 = 0$ (3): Giải (3) tìm x_1 & x_2
 - Nếu (2) có 2 nghiệm
 - $y = y_1 = x^2 \Leftrightarrow x^2 - y_1 = 0$ (4): Giải (4) tìm x
 - $y = y_2 = x^2 \Leftrightarrow x^2 - y_2 = 0$ (5): Giải (5) tìm x
- Lưu ý: không cần kiểm tra điều kiện $y \geq 0$ vì việc đó sẽ do EqualDeg2 xử lý

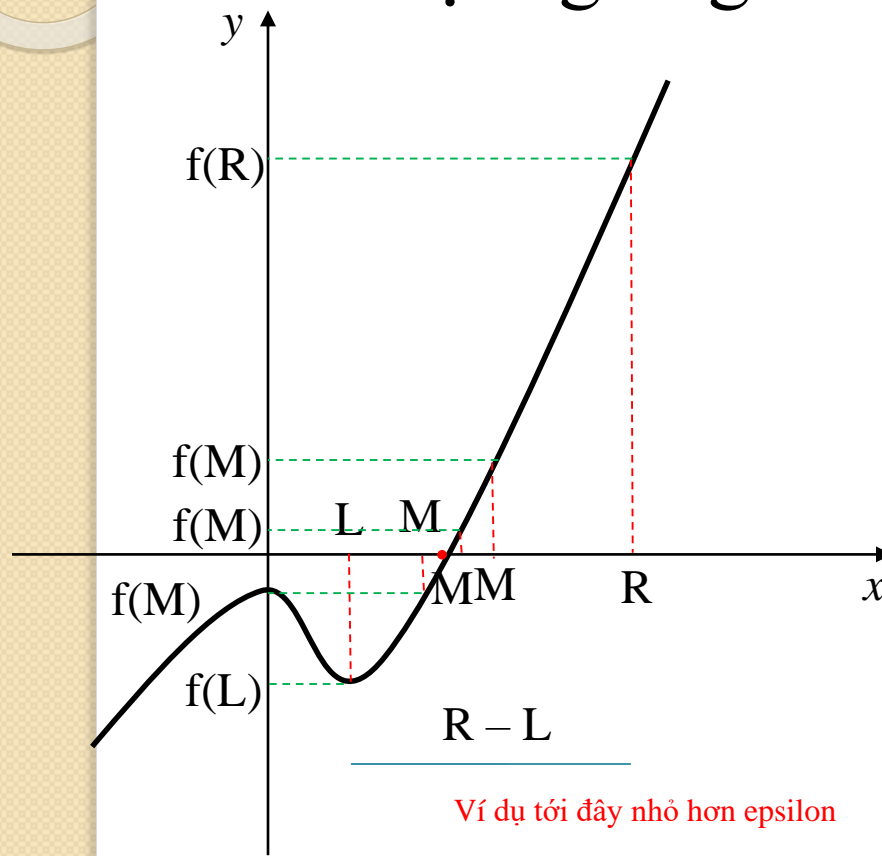
CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 8 xét phương trình $ax^4 + bx^2 + c = 0$

Dòng	Mô tả
1	...
2	<code>int EqualQuartic(double a, double b, double c, double& x1, double& x2, double& x3, double& x4){</code>
3	<code>int nSolution, nSol1, nSol2; double y1, y2; x1 = x2 = x3 = x4 = 0;</code>
4	<code>nSol1 = EqualDeg2(a, b, c, y1, y2);</code>
5	<code>switch(nSol1){</code>
6	<code>case NoSolution: case Undetermined: nSolution = nSol1; break;</code>
7	<code>case 1: nSolution = EqualDeg2(1, 0, -y1, x1, x2); break;</code>
8	<code>case 2: nSol2 = EqualDeg2(1, 0, -y1, x1, x2);</code> → (4)
9	<code>switch(nSol2){</code>
10	<code>case NoSolution: nSolution = EqualDeg2(1, 0, -y2, x1, x2); break;</code> → (5)
11	<code>case 1: nSolution = ① + EqualDeg2(1, 0, -y2, x2, x3); break;</code> (5)
12	<code>case 2: nSolution = ② + EqualDeg2(1, 0, -y2, x3, x4); break;</code>
13	<code>}} return nSolution; }</code>

CÁC VẤN ĐỀ HÀM TRONG C/C++

- Ví dụ 9 giải gần đúng phương trình $f(x)=0$



Ví dụ tới đây nhỏ hơn epsilon

Điều kiện để tồn tại nghiệm:

- 1) Hàm số đơn điệu trên (L, R)
- 2) Tích $f(L)*f(R) < 0$

Dòng	Mô tả
1	<code>double f(double x){</code>
2	<code>return pow(x, 9) + x + 1; //f(x)=x⁹+x+1</code>
3	<code>}</code>
4	<code>void Solve(int& x){ //Tìm x để f(x) = 0</code>
5	<code>const double epsilon = 0.000000001;</code>
6	<code>double left = -1, right = 0; // a=-1, b=0</code>
7	<code>while(right - left > epsilon){</code> → (0 ss bằng)
8	<code>double mid = (left + right)/2;</code>
9	<code>if(f(left)*f(mid) < 0) right = mid;</code>
10	<code>else left = mid;</code>
11	<code>}</code>
12	<code>x = (left + right)/2;</code>
13	<code>}</code>

HÀM TRÙNG TÊN

- Là các hàm có tên giống nhau, phân biệt bằng danh sách tham số đầu vào và kiểu dữ liệu trả về
- Ví dụ hàm tròn số
 - `double round(double)`: trả về số nguyên được làm tròn từ số thực theo nguyên tắc đến 0.5
 - Ví dụ: `round(1.9) → 2`
 - `double round(double, int)`: trả về số nguyên được làm tròn tới n chữ số lẻ
 - Ví dụ: `round(1.879, 2) → 1.88`

HÀM TRÙNG TÊN

- Ví dụ

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>#include <math.h></code>
3	<code>double round(double);</code>
4	<code>double round(double, int);</code>
5	<code>double round(double x){</code>
6	<code>double kq;</code>
7	<code>if(x >= 0) kq = floor(x + 0.5);</code>
8	<code>else kq = -floor(-x + 0.5);</code>
9	<code>return kq;</code>
10	<code>}</code>

Dòng	Mô tả
11	<code>double round(double x, int n){</code>
12	<code>double kq, s = pow(10, n);</code>
13	<code>x*= s;</code>
14	<code>if(x >= 0) kq = floor(x + 0.5)/s;</code>
15	<code>else kq = -floor(-x + 0.5)/s;</code>
16	<code>return kq;</code>
17	<code>}</code>

HÀM CÓ THAM SỐ MẶC ĐỊNH

- Mục tiêu nhằm giảm lược các hàm trùng tên (nếu cần thiết)
- Hàm round
 - Thay vì viết hai hàm ta dùng một hàm có tham số mặc định
 - Tham số mặc định là tham số nếu ta không truyền giá trị nó sẽ có giá trị mặc định trước
- Ví dụ
 - `double round(double, int = 0)`

tham số mặc định bắt buộc nằm bên trục phải

HÀM CÓ THAM SỐ MẶC ĐỊNH

- Ví dụ

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>#include <math.h></code>
3	<code>double round(double, int=0);</code>
4	<code>double round(double x, int n){</code>
5	<code>double kq, s = pow(10, n);</code>
6	<code>x*=s;</code>
7	<code>if(x >= 0) kq = floor(x + 0.5)/s;</code>
8	<code>else kq = -floor(-x + 0.5)/s;</code>
9	<code>return kq;</code>
10	<code>}</code>

Dòng	Mô tả
11	<code>void main(){</code>
12	<code>double a = 10.237;</code>
14	<code>double kq1 = round(a, 2);</code>
15	<code>double kq2 = round(a);</code>
16	<code>printf("kq1 = %lf", kq1);</code>
17	<code>printf("kq2 = %lf", kq2);</code>
18	<code>}</code>

HÀM CÓ THAM SỐ KIỂU

- Mục tiêu nhằm hỗ trợ viết các hàm độc lập kiểu dữ liệu
- Hàm swap:
 - Hàm hay sử dụng
 - Cần viết chồng nhiều hàm khi thay đổi kiểu dữ liệu
- Ví dụ
 - `void swap(double& a, double& b) {`
 - `double c = a; a = b; b = c;`
 - `}`
 - `void swap(int& a, int& b) {`
 - `int c = a; a = b; b = c;`
 - `}`
 - `void swap(long& a, long& b){`
 - `long c = a; a = b; b = c;`
 - `}`
- Giải pháp
 - `template <class T>`
 - `void swap(T& a, T& b) {T c = a; a = b; b = c;}`

HÀM CÓ THAM SỐ LÀ HÀM

- Ngôn ngữ C/C++ cho phép ta cài đặt một hàm có tham số là hàm
- Điều này làm tăng tính linh hoạt và tổ chức hơn cho chương trình
- Xét ví dụ viết hàm đếm theo yêu cầu
 - DemTheoYeuCau(long, int KiemTra(int)): sẽ đếm xem các kí số có thỏa hàm KiemTra hay không:
 - Ví dụ:
 - 1239 có 2 kí số nguyên tố nếu KiemTra là hàm KiemTraSNT
 - 1239 có 3 kí số lẻ nếu KiemTra là hàm KiemTraSoLe

HÀM CÓ THAM SỐ LÀ HÀM

- Ví dụ

Dòng	Mô tả
1	<code>#include <stdio.h></code>
2	<code>int Dem(int, int KT(int));</code>
3	<code>int KTSNT(int);</code>
4	<code>int KTSNT(int n){</code>
5	<code>if(n == 1 n == 0) return 0;</code>
6	<code>for(int i = 2; i < n; i++)</code>
7	<code>if(n % i == 0) return 0;</code>
8	<code>return 1;</code>
9	<code>}</code>

Dòng	Mô tả
10	<code>int Dem(int a, int KT(int)){</code>
11	<code>int tmp, count = 0;</code>
12	<code>do{</code>
13	<code>tmp = a%10; a = a/10;</code>
14	<code>if(KT(tmp) == 1)count++;</code>
15	<code>}while(a!=0);</code>
16	<code>return count;</code>
17	<code>}</code>
18	<code>void main(){</code>
19	<code>int a = 1239; // Dem(a, KTSNT)</code>
20	<code>int d = Dem(a, KTSNT);</code>
21	<code>printf("d = %d\n", d);</code>
22	<code>}</code>

HÀM CÓ THAM SỐ LÀ HÀM

- Xét lại ví dụ tính gần đúng

Dòng	Mô tả	
1	<code>double f(double x){</code>	<code>double g(double x){</code>
2	<code>return pow(x, 9) + x + 1; //f(x)=x⁹+x+1</code>	<code>return pow(x, 5) + 7*x + 1; //g(x)=x⁵+7x+1</code>
3	<code>}</code> <i>chỉ giải đc f(x)</i>	<code>}</code> <i>tổng quát hóa hàm Solve</i>
4	<code>void Solve(int& x) { //Tìm x để f(x) = 0</code>	<code>double Solve(double F(double x), double a, double b){</code>
5	<code>const double epsilon = 0.000000001;</code>	<code>const double epsilon = 0.000000001;</code>
6	<code>double left = -1, right = 0; // a=-1, b=0</code>	<code>double left = a, right = b;</code>
7	<code>while(right - left > epsilon){</code>	<code>while(right - left > epsilon){</code>
8	<code>double mid = (left + right)/2;</code>	<code>double mid = (left + right)/2;</code>
9	<code>if(f(left)*f(mid) < 0) right = mid;</code>	<code>if(F(left)*F(mid) < 0) right = mid;</code>
10	<code>else left = mid;</code>	<code>else left = mid;</code>
11	<code>}</code>	<code>}</code>
12	<code>x = (left + right)/2;</code>	<code>return (left + right)/2;</code>
13	<code>}</code>	<code>}</code>

HÀM CÓ THAM SỐ LÀ HÀM

- Xét lại ví dụ tính gần đúng

Dòng	Mô tả	14	<code>void main(){</code>
1	<code>double g(double x){</code>	15	<code>double x = Solve(f, -1, 0);</code>
2	<code>return pow(x, 5) + 7*x + 1; //g(x)=x⁵+7x+1</code>	16	<code>printf("%ld\n", x);</code>
3	<code>}</code>	17	<code>x = Solve(g, 2, 5);</code>
4	<code>double Solve(double F(double x), double a, double b){</code>	18	<code>printf("%ld\n", x);</code>
5	<code>const double epsilon = 0.000000001;</code>	19	<code>}</code>
6	<code>double left = a, right = b;</code>		
7	<code>while(right - left > epsilon){</code>		
8	<code>double mid = (left + right)/2;</code>		
9	<code>if(F(left)*F(mid) < 0) right = mid;</code>		
10	<code>else left = mid;</code>		
11	<code>}</code>		
12	<code>return (left + right)/2;</code>		
13	<code>}</code>		

NỘI DUNG

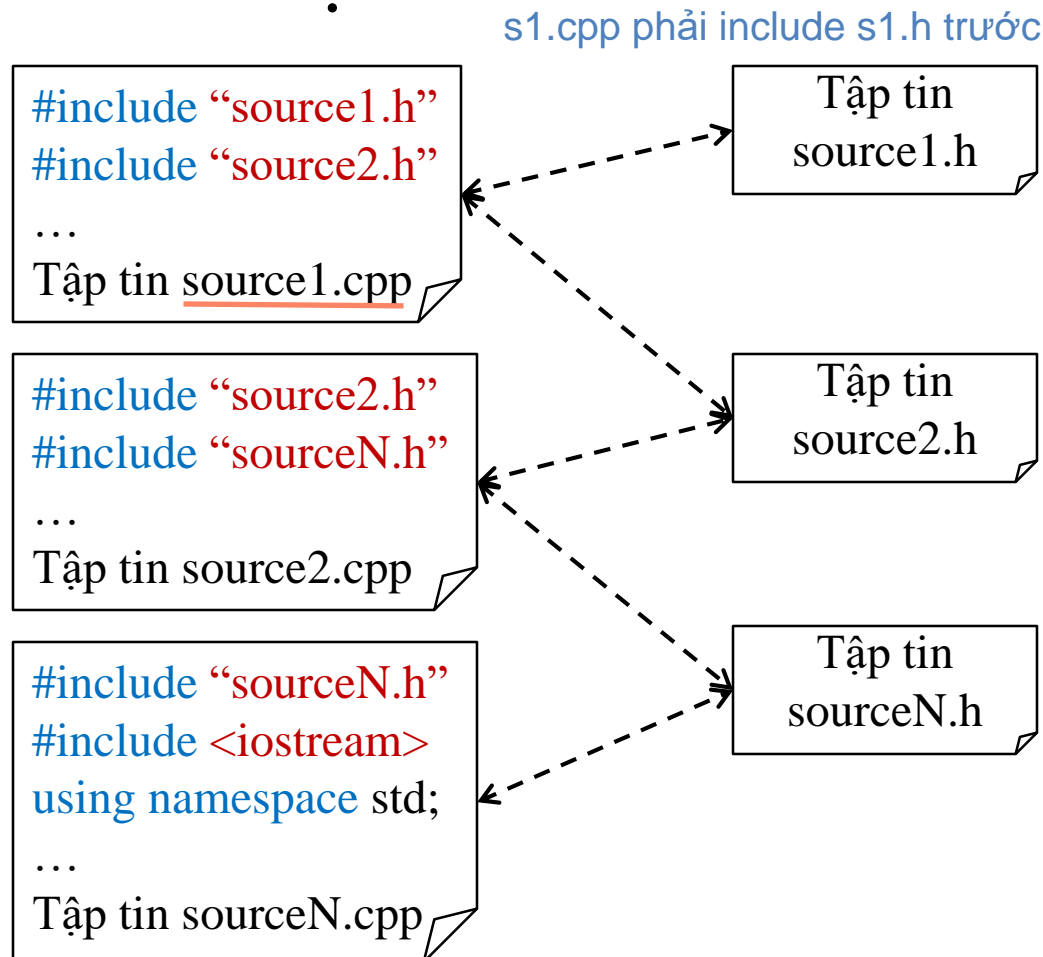
- Giới thiệu
- Cơ chế truyền tham số
- Các vấn đề về hàm trong C/C++
 - Hàm trùng tên
 - Hàm có tham số mặc định
 - Hàm có tham số là hàm
- Tổ chức hàm với nhiều tập tin mã nguồn
- Phạm vi hàm và biến toàn cục với nhiều tập tin mã nguồn

TỔ CHỨC HÀM VỚI NHIỀU TẬP TIN MÃ NGUỒN

- Một phần mềm viết bằng C/C++ bao gồm nhiều tập tin mã nguồn
 - Các tập tin có phần mở rộng `‘.h’`: mô tả giao tiếp lập trình, chứa các lệnh khai báo hàm và các hằng số
 - Các tập tin có phần mở rộng `‘.cpp’` hay `‘.c’`: cài đặt mã nguồn, chứa chi tiết đoạn mã của các hàm đã khai báo trong phần mô tả giao tiếp. Cần dùng từ khóa `‘include’` để tham chiếu tới phần mô tả giao tiếp

TỔ CHỨC HÀM VỚI NHIỀU TẬP TIN MÃ NGUỒN

- Sơ đồ ví dụ:



TỔ CHỨC HÀM VỚI NHIỀU TẬP TIN MÃ NGUỒN

- Ví dụ giải các phương trình

```
#include <math.h>
#include "Equation.h"
//...
//Tập tin Equation.cpp
int EquaDeg1(double a, double b,
double& x){
    //...
}

int EquaDeg2(double a, double b,
double c, double& x, double& y){
    //...
}
```

```
//Tập tin Equation.h
#ifndef EQUATION_H_
#define EQUATION_H_    khai báo
#define NoSolution 0
#define Underermined -1
int EquaDeg1(double, double, double&);
int EquaDeg2(double, double, double,
double&, double&);
//...
#endif
```

TỔ CHỨC HÀM VỚI NHIỀU TẬP TIN MÃ NGUỒN

- Ví dụ giải các phương trình

```
#include "EquationIO.h" ←
#include "Equation.h"
//...
//Tập tin EquationIO.c ←
void SolutionPrint(int n, double x,
double y = 0){
    switch(n){
        case NoSolution:
            //...
    }
}
void EquaDisplay(double a,
double b){
    //...
}
//...
```

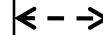
```
//Tập tin EquationIO.h
#ifndef _EQUATIONIO_H_
#define _EQUATIONIO_H_

void SolutionPrint(int, double, double=0);
void EquaDisplay(double, double);
void EquaDisplay(double, double, double);
void EquationInput(double&, double&);
void EquationInput(double&, double&,
double&);

//...
#endif
```

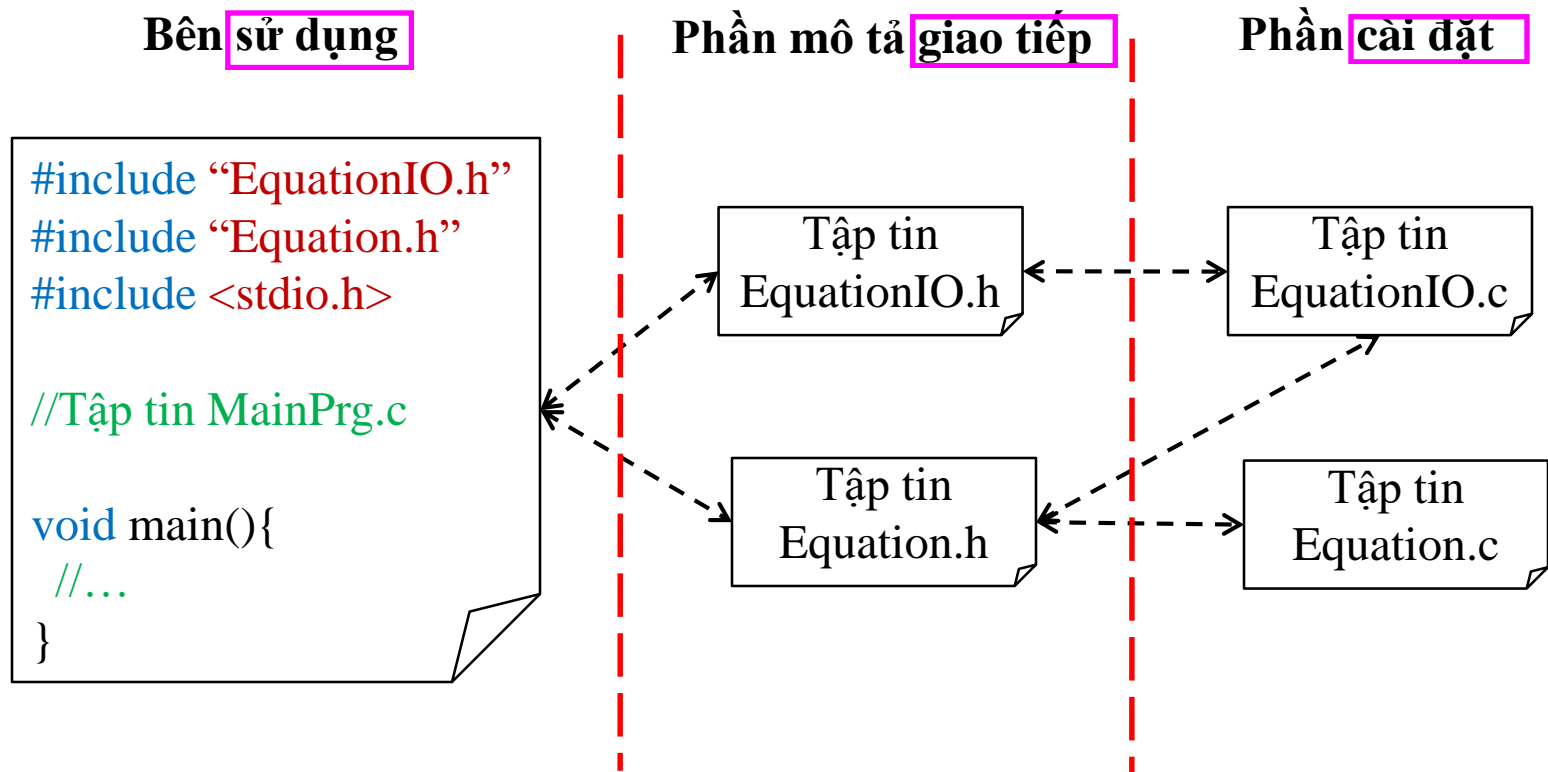
Tập tin
Equation.h

Tập tin
Equation.c



TỔ CHỨC HÀM VỚI NHIỀU TẬP TIN MÃ NGUỒN

- Ví dụ giải các phương trình



NỘI DUNG

- Giới thiệu
- Cơ chế truyền tham số
- Các vấn đề về hàm trong C/C++
 - Hàm trùng tên
 - Hàm có tham số mặc định
 - Hàm có tham số là hàm
- Tổ chức hàm với nhiều tập tin mã nguồn
- Phạm vi hàm và biến toàn cục với nhiều tập tin mã nguồn

PHẠM VI HÀM & BIẾN TOÀN CỤC

- Một chương trình có nhiều tập tin mã nguồn sẽ xuất hiện một số tình huống
 - Tập tin A muốn dùng hàm trong tập tin B (giải pháp: dùng các tập tin giao tiếp `.h`)
 - Biến toàn cục hay hàm trong tập tin A được giữ kín sao cho các tập tin khác không được dùng tới (giải pháp: thêm từ khóa `static` trước khai báo biến toàn cục hay hàm)
 - Biến toàn cục trong tập tin A và các tập tin khác ĐƯỢC dùng tới (giải pháp: thêm từ khóa `extern` tại các tập tin cần dùng)

PHẠM VI HÀM & BIẾN TOÀN CỤC

- Ví dụ

```
static int nItem;  
extern int nCounter; <-  
  
void Func(){  
    //...  
}
```

```
static int nItem;  
int nCounter; <-  
  
static void Func(){  
    //...  
}
```

```
extern int nCounter; <-  
  
void main(){  
    //...  
}
```