ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

TIN HỌC ĐẠI CƯƠNG

Phần II: LẬP TRÌNH C

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và xâu ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- Chương 8: Tệp dữ liệu

Chương 2: Kiểu dữ liệu và biểu thức trong C

2.1. Các kiểu dữ liệu chuẩn trong C

2.2. Biểu thức trong C

2.3. Các toán tử trong C

2.4. Một số toán tử đặc trưng

Chương 2: Kiểu dữ liệu và biểu thức trong C

2.1. Các kiểu dữ liệu cơ bản trong C

2.2. Biểu thức trong C

2.3. Các toán tử trong C

2.4. Một số toán tử đặc trưng

Các kiểu đơn

Tên kiểu	Ý nghĩa	Kích thước	Miền dữ liệu
char	Kí tự; Số nguyên có dấu	1 byte	-128 ÷ 127
short int	Số nguyên có dấu	2 byte	-32.768÷32.767
int	Số nguyên có dấu	2 hoặc **4 byte**	
long long int	Số nguyên có dấu	4 byte	-2,147,483,648 ÷ 2,147,483,647
float	Số thực dấu phẩy động, độ chính xác đơn	4 byte	± 3.4E-38 ÷ ± 3.4E+38
double	Số thực dấu phẩy động, độ chính xác kép	8 byte	±1.7E-308 ÷ ± 1.7E+308

Các kiểu kết hợp

Với số nguyên, thêm từ khóa unsigned để chỉ ra số không dấu

Kiểu dữ liệu	Ý nghĩa	Kích thước	Miền dữ liệu
unsigned char	Số nguyên không dấu	1 byte	0 ÷ 255
unsigned short	Số nguyên không dấu	2 byte	0÷65.535
unsigned int	Số nguyên không dấu	2 hoặc 4 byte	
unsigned long unsigned long int	Số nguyên không dấu	4 byte	0 ÷ 4,294,967,295
long double	Số thực dấu phẩy động,	10 byte	±3.4E-4932 ÷ ±1.1E+4932
void	Là kiểu đặc biệt, không có kích thước		

Biểu diễn hằng số (literal)

Kiểu dữ liệu	Ví dụ	Ý nghĩa
Số nguyên	123, -12	Số hệ 10
	012, 03777	Số hệ 8
	0x7F, 0x3fe15	Số hệ 16
	39u 0267u, 0xFFu	Số không dấu
Số nguyên lớn (kiểu long)	12L, 07723L 0xFFL, -10L 0xFFUL,0xFFLU	
Số thực	3.1415 -12.3, .327 10e-12, -15.3E12 3.1415F, -12.F	

Khai báo biến

- Một biến phải được khai báo trước khi sử dụng
- Cú pháp khai báo:

```
<KieuDuLieu> TenBien;
<KieuDuLieu> TenBien1, …, TenBien_N;
```

- Ví dụ:

```
//Khai báo biến x là một số nguyên 4 byte có dấu
        int x;
//Khai báo các biến y, z là các số thực 4 byte
        float  y,z;
//Sau khi khai báo, có thể sử dụng
        x = 3; y = x + 1;
```

Khai báo biến

int n; m = 2 * n; ⇒ m=?

- Sau khi khai báo, biến chưa có giá trị xác định.
  - Biến cần được gán giá trị trước khi sử dụng
- C cho phép kết hợp khai báo và khởi tạo biến

```
KieuDuLieu TenBien = GiaTriBanDau;
KieuDuLieu Bien1 = GiaTri1, BienN = Gia_TriN;
```

- Ví dụ:

```
//Khai báo biến nguyên a và khởi tạo gia tri bằng 3
int a = 3;
//Khai báo biến thực x,y và khởi tạo giá tri bằng 5.0 và 7.6
float x = 5.0, y = 7.6;
```

Khai báo hằng (dùng macro)

- Dùng chỉ thị #define
- Cú pháp:

```
# define  Tên_hằng  Giá_trị
```

Không có dấu ;

- Ví dụ:

```
#define  MAX_SINH_VIEN  50
#define  CNTT  “Cong nghe thong tin”
#define  DIEM_CHUAN  23.5
```

Khai báo hằng

- Dùng từ khóa const
- Cú pháp:

```
const  Kiểu  Tên_hằng = giá_trị;
```

- Ví dụ:

```
const int  MAX_SINH_VIEN  = 50;
const char  CNTT[20] = “Cong nghe thong tin”;
const float  DIEM_CHUAN = 23.5;
```

Khai báo hằng

- Chú ý:
- Giá trị của các hằng phải được xác định ngay khi khai báo.
- #define là chỉ thị tiền xử lý
  - Khi chương trinh được biên dịch, <Tên_hằng> sẽ được thay thế bằng <Giá_trị>
  - Dễ đọc, dễ thay đổi
  - Dễ chuyển đổi giữa các nền tảng phần cứng hơn

Chương 2: Kiểu dữ liệu và biểu thức trong C

2.1. Các kiểu dữ liệu cơ bản trong C

2.2. Biểu thức trong C

2.3. Các toán tử trong C

2.4. Một số toán tử đặc trưng

Mục đích sử dụng

- Làm vế phải của lệnh gán.
- Làm toán hạng trong các biểu thức khác.
- Làm tham số thực sự trong lời gọi hàm.
- Làm biểu thức kiểm tra trong các cấu trúc điều khiển
  - Cấu trúc lặp: for, while, do while.
  - Cấu trúc rẽ nhánh: if, switch.

Tính toán giá trị biểu thức

- Các toán hạng được thay thế bởi giá trị tương ứng
- Các phép tính được thực hiện theo thứ tự

Ví dụ (alpha = 10, beta = 81)

Biểu thức: alpha + sqrt(beta)

: alpha + sqrt(81)

: alpha + 9.0

: 10 + 9.0

: 19.0

Các loại biểu thức

- Biểu thức số học
- Biểu thức quan hệ
- Biểu thức logic

Biểu thức số học

- Là biểu thức mà giá trị của nó là các đại lượng số học (số nguyên, số thực).
  - Sử dụng các toán tử là các phép toán số học (cộng, trừ, nhân, chia…),
  - Các toán hạng là các đại lượng số học (hằng số, biến, biểu thức khác).
- Ví dụ: a, b, c là các biến thuộc kiểu số thực.
  - 3 * 3.7
  - 8 + 6/3
  - a + b – c
- Chú ý: phép chia số nguyên/số nguyên → số nguyên

Biểu thức quan hệ

- Là những biểu thức có sử dụng các toán tử quan hệ như lớn hơn, nhỏ hơn, khác nhau…
- Chỉ có thể trả về một trong 2 giá trị logic Đúng (TRUE) hoặc Sai (FALSE)
- Ví dụ

```
5 > 7          // có giá trị logic là sai, FALSE
9 != 10        // có giá trị logic là đúng, TRUE
2 >= 2         // có giá trị logic là đúng, TRUE
a > b          // giả sử a, b là 2 biến kiểu int
```

- Chú ý: Ngôn ngữ C coi các giá trị nguyên khác 0 là giá trị logic đúng (TRUE), giá trị 0 là giá trị logic sai (FALSE)

Biểu thức logic

- Là biểu thức trả về các giá trị logic Đúng/Sai
  - Các phép toán logic gồm có
    - AND VÀ logic, sử dụng toán tử &&
    - OR HOẶC logic, sử dụng toán tử ||
    - NOT PHỦ ĐỊNH, sử dụng toán tử !
  - Biểu thức quan hệ là trường hợp riêng của biểu thức logic.
  - Biểu thức logic cũng trả về một giá trị số học 0/1

Biểu thức logic →Ví dụ

(5 > 7)&&(9!=10)	// có giá trị logic là sai, FALSE
0 || 1	// có giá trị logic là đúng, TRUE
(5 > 7)||(9!=10)	// có giá trị logic là đúng, TRUE
0	// có giá trị logic là sai, FALSE
!0	// phủ định của 0, có giá trị logic là đúng, TRUE
3	// có giá trị logic là đúng, TRUE
!3	// phủ định của 3, có giá trị logic là sai, FALSE
(a > b)&&(a < b)	// Có giá trị sai, FALSE. Giả sử a, b là 2 biến kiểu int

Chương 2: Kiểu dữ liệu và biểu thức trong C

2.1. Các kiểu dữ liệu cơ bản trong C

2.2. Biểu thức trong C

2.3. Các toán tử trong C

2.4. Một số toán tử đặc trưng

Các toán tử chính

- Toán tử số học
- Toán tử quan hệ
- Toán tử logic
- Toán tử logic bit
- Toán tử gán

Các toán tử số học

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ (*int a = 12; float x=3.0*)
-	Đảo dấu	float, double, int, long,.. (*Số nguyên hoặc thực*)	-12, -12.34, - a, - x
+	Cộng	float, double, int, long,..	12 + x
-	Trừ	float, double, int, long,..	12.0 - x
*	Nhân	float, double, int, long,.. (*Số nguyên hoặc thực*)	12 * 3.0 12 * 3
/	Chia	Nếu có ít nhất 1 toán hạng là số thực	17.0/3.0 → 5.666667 17/3.0 → 5.666667 17.0/3 → 5.666667
/	Chia lấy thương	**Số nguyên** int, long,..	17/3 → 5
%	Chia lấy số dư	**Số nguyên:** int, long,..	17%3 → 2

Các toán tử quan hệ

<, >, <=, >=, ==, !=

- Dùng cho phép so sánh giá trị 2 toán hạng
- Kết quả phép so sánh là một số nguyên
  - 1 nếu quan hệ có kết quả là đúng,
  - 0 nếu quan hệ có kết quả sai
- Ví dụ:
  - 6 > 4 → Trả về giá trị 1
  - 6 < 4 → Trả về giá trị 0
  - int b = (x !=y);

    Nếu x và y khác nhau, biểu thức đúng và b mang giá trị 1. Ngược lại biểu thức sai và b mang giá trị 0

Các toán tử logic

- Sử dụng để xây dựng các biểu thức logic
- Biểu thức logic có kết quả logic đúng
  - $\rightarrow$ Trả về giá trị 1
- Biểu thức logic có kết quả logic sai
  - $\rightarrow$ Trả về giá trị 0

Các toán tử logic (tiếp)

- Và logic (&&) :
  - Cho kết quả đúng (trả về giá trị 1) khi cả 2 toán hạng đều đúng (khác 0)
  - Ví dụ: 3 < 5 && 4 < 6 → 1; 3 < 5 && 5 > 6 → 0
- Hoặc logic (||):
  - Cho kết quả sai (trả về giá trị 0) chỉ khi cả 2 toán hạng đều sai (bằng 0)
  - Ví dụ: 4 || 5 < 3 → 1; 5 < 5 || 2 > 6 → 0
- Phủ định logic (!):
  - Cho kết quả đúng (1) hoặc sai (0) khi toán hạng là sai (0) hoặc đúng (khác 0)
  - Ví dụ: !3 → 0; !(2 > 3) → 1;

Toán tử trên bit

- Toán tử trên bit (bitwise operator) được sử dụng với kiểu số nguyên

  Và nhị phân: Op1 & Op2

  Hoặc nhị phân : Op1 | Op2

  Hoặc có loại trừ nhị phân: Op1 ^ Op2

  Đảo bít nhị phân : ~ Op

  Dịch trái n bit: Op << n

  Dịch phải n bit: Op >> n

Toán tử trên bit (tiếp)

char Op1 = 83, Op2 = -38, Op = 3;

char r = Op1 & Op2;

0 1 0 1 0 0 1 1

1 1 0 1 1 0 1 0

r = 0 1 0 1 0 0 1 0 → (82)

char r = Op1 | Op2;

0 1 0 1 0 0 1 1

1 1 0 1 1 0 1 0

r = 1 1 0 1 1 0 1 1 → (-37)

char r = Op1 ^ Op2;

0 1 0 1 0 0 1 1

1 1 0 1 1 0 1 0

r = 1 0 0 0 1 0 0 1 → (-119)

char r = ~ Op2;

1 1 0 1 1 0 1 0

r = 0 0 1 0 0 1 0 1 → (37)

unsigned char r = Op1 | Op2; r = 1 1 0 1 1 0 1 1 → 219

Toán tử trên bit (tiếp)

char Op1 = 83, Op2 = -38, Op = 3;

char r = Op1 >> Op;

0 1 0 1 0 0 1 1

r = 0 0 0 0 1 0 1 0 → (10)

char r = Op2 >> Op;

1 1 0 1 1 0 1 0

r =1 1 1 1 1 0 1 1 → (-5)

unsigned char Op =218;

unsigned char r =Op >> 3;

1 1 0 1 1 0 1 0

r =0 0 0 1 1 0 1 1 → (27)

char r = Op2 << 2;

r = 0 1 1 0 1 0 0 0 →(104)

(unsigned) int r = Op2<<2 →?

Toán tử gán

```
Biến = Biểu_thức;
```

- Là toán tử được sử dụng thường xuyên
  - Biểu thức bên phải dấu bằng được tính toán
  - Giá trị của biểu_thức được gán cho biến ở vế trái
- Ví dụ:
  - int a, b, c;
  - a = 3;
  - b = a + 5;
  - c = a * b;

Toán tử gán

- Biểu thức gán là biểu thức nên cũng có giá trị.
  - Giá trị của biểu thức gán bằng giá trị của biểu_thức bên phải toán tử
  - Có thể gán giá trị của biểu thức gán cho một biến khác
  - Có thể sử dụng như một biểu thức bình thường
- Ví dụ:
  - int a, b, c;
  - a = b = 2007;
  - c = (a = 20) * (b = 30);// c→600

Toán tử gán dạng kết hợp

- Toán tử số học:

+= -= *= /= %=

Ví dụ: a *= b // a= a * b

Toán tử trên bit:

&= |= ^= <<= >>=

Ví dụ:

x &= 0x3F // x = x & 0x3F

s <<= 4 // s = s << 4

Chương 2: Kiểu dữ liệu và biểu thức trong C

2.1. Các kiểu dữ liệu cơ bản trong C

2.2. Biểu thức trong C

2.3. Các toán tử trong C

2.4. Một số toán tử khác

Các toán tử

- Tăng/giảm tự động một đơn vị
- Lấy địa chỉ
- Biểu thức điều kiện
- Toán tử phẩy

Tăng giảm tự động một đơn vị

++	Tăng tự động	++Var, Var++
--	Giảm tự động	--Var, Var--

- Tiền tố (hậu tố): biến được tăng(++)/giảm(--) trước (sau) khi sử dụng để tính toán biểu thức
- Ví dụ:

```
int     a = 5, b, c, d, e;
b = a++;            // b = 5 sau đó a = 6
c = ++a;            // a = 7 rồi tới c = 7
d = a--;            // d = 7 rồi tới a = 6
e = --a;            // a = 5 sau đó e= 5
```

Toán tử lấy địa chỉ

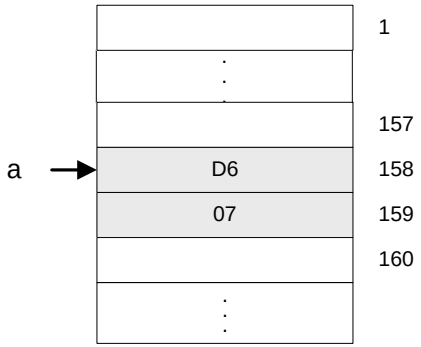
`&Tên_biến`

- Biến thực chất là một vùng nhớ của máy tính được đặt tên → tên của biến
- Mọi ô nhớ trên bộ nhớ máy tính đều được đánh địa chỉ.

  → Mọi biến đều có địa chỉ
- Ví dụ:

```
short int a = 2006;
```

&a là địa chỉ của ô nhớ chứa giá trị biến a



Biểu thức điều kiện

```
exp1 ? exp 2 : exp3
```

- Nếu exp1 ≠ 0 (giá trị đúng), biểu thức điều kiện trả về giá trị của exp2
- Nếu exp1 = 0 (giá trị sai) biểu thức điều kiện trả về giá trị của exp3
- Ví dụ:

```
float x= 5.2, y = 3.8, z;
z = (x < y) ? x : y;
```

→ z = 3.8 // z min{x, y}

⇔ if (x < y) z = x; else z = y;

Toán tử phẩy (, comma)

```
biểu_thức_1, biểu thức_2,..
```

- Toán tử phẩy (,) cho phép sử dụng nhiều biểu thức tại nơi chỉ cho phép viết một biểu thức
- Các biểu thức được tính toán từ trái qua phải
- Giá trị và kiểu của biểu thức là giá trị và kiểu của biểu thức cuối cùng, bên phải
- Ví dụ:

```
if (i = 0, a !=b)…
for(i = 0, j = 0; i < 100; i++, j++)….
```

Chuyển kiểu

```
(Kiểu) biểu thức
```

- Chuyển kiểu tự động
  - Chương trình dịch tự động chuyển đổi từ kiểu có phạm vi biểu diễn thấp tới kiểu có phạm vi biểu diễn cao

  char → int → long int → float → double → long double
- Ép kiểu
  - Bằng câu lệnh tường minh trong chương trình
  - Được sử dụng khi muốn tự chuyển kiểu dữ liệu, hoặc thao tác chuyển kiểu tự động không được hỗ trợ

Ví dụ

```
#include <stdio.h>
#include <conio.h>
int main(){
    long L = 0xABCDEF;
    float f = 123.456;
    int i;
    i = (int) L;
    printf("\n L = %ld; i = %d(%X)", L, i, i);
    i = (int) f; L = (long) f;
    printf("\n f = %f; L = %ld;  i = %d", f, L, i);
}
```

Thứ tự ưu tiên các toán tử

Mức	Toán tử	Chức năng	Chiều	
1	-> . [ ] () ++ hậu tố --hậu tố	Lựa chọn, chỉ số…	→	Chiều kết hợp với các toán hạng
2	++ -- ~ ! + - * & () sizeof	Toán tử 1 ngôi, ép kiểu,…	←	
3	* / %	Toán tử số học lớp nhân	→	
4	+ -	Toán tử số học lớp cộng	→	
5	>> <<	Dịch bit	→	
6	< <= > >=	Toán tử quan hệ	→	
7	== !=	Bằng, khác	→	
8	&	AND nhị phân	→	
9	^	XOR nhị phân	→	
10	|	OR nhị phân	→	
11	&&	AND logic	→	
12	||	OR logic	→	
13	? :	Toán tử phỏng điều kiện	←	
14	= *= += <<= &= ...	Toán tử gán	←	

Thứ tự ưu tiên các toán tử

- Nguyên tắc
- Biểu thức con trong ngoặc được tính toán trước
- Phép toán một ngôi đứng bên trái toán hạng được kết hợp với toán hạng đi liền nó.
- Toán hạng đứng cạnh hai toán tử
  - Nếu hai toán tử có độ ưu tiên khác nhau thì toán tử nào có độ ưu tiên cao hơn sẽ kết hợp với toán hạng
  - Nếu hai toán tử cùng độ ưu tiên thì dựa vào trật tự kết hợp của các toán tử để xác định toán tử được kết hợp với toán hạng.
- Ví dụ

  a < 10 && 2 * b < c ≡ (a < 10) && ((2 * b) < c)

  Chú ý: int x = 5, a = 5 * x++; → a = 25, x = 6

Ví dụ

```
const int N=10;
float S= 0.0;
int b;
S = N/3 +1;
b=(S>4);

S= ?    b = ?
```

```
int a= 3, b=4, c;
c = a++ * ++b;

a= ?   b= ?   c= ?
```

```
int k, num=30;
k = num>5 ? (num <=10 ? 100 : 200): 500;
k=?
```

Ví dụ

```
const int N=10;
float S= 0.0;
int b;
S = N/3 +1;
b=(S>4);
```

S= 4 b = 0

```
int a= 3, b=4, c;
c = a++ * ++b;
```

a=4 b=5 c=15

```
int k, num=30;
k = num>5 ? (num <=10 ? 100 : 200): 500;
```

k=200

Tóm tắt chương 2

- Kiểu dữ liệu
  - Nguyên: char, unsigned char, int, long, unsigned int, unsigned long
  - Thực: float, double, long double
- Giá trị logic
  - Đúng/TRUE : 1 (*Khác 0*)
  - Sai/FALSE : 0
- Toán tử
  - Một ngôi : + -; ++ --; ~ !; &; ();
  - Hai ngôi : + - * / %; == != < <= > >=; << >>; &, ^, |, && ||; = *= +=…
  - 3 ngôi : ? :