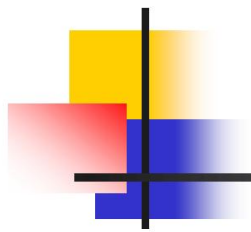




# Toán tử và Biểu thức

Chỉ dành cho Trung tâm Aptech sử dụng

## Phiên 3



# Mục tiêu

---

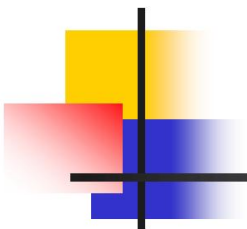
Giải thích về toán tử gán

Hiểu biểu thức số học Giải thích

về toán tử quan hệ và logic Chỉ dành cho Trung tâm Aptech sử dụng

toán tử và biểu thức logic bitwise Giải thích về ép kiểu Hiểu

về thứ tự ưu tiên của toán tử

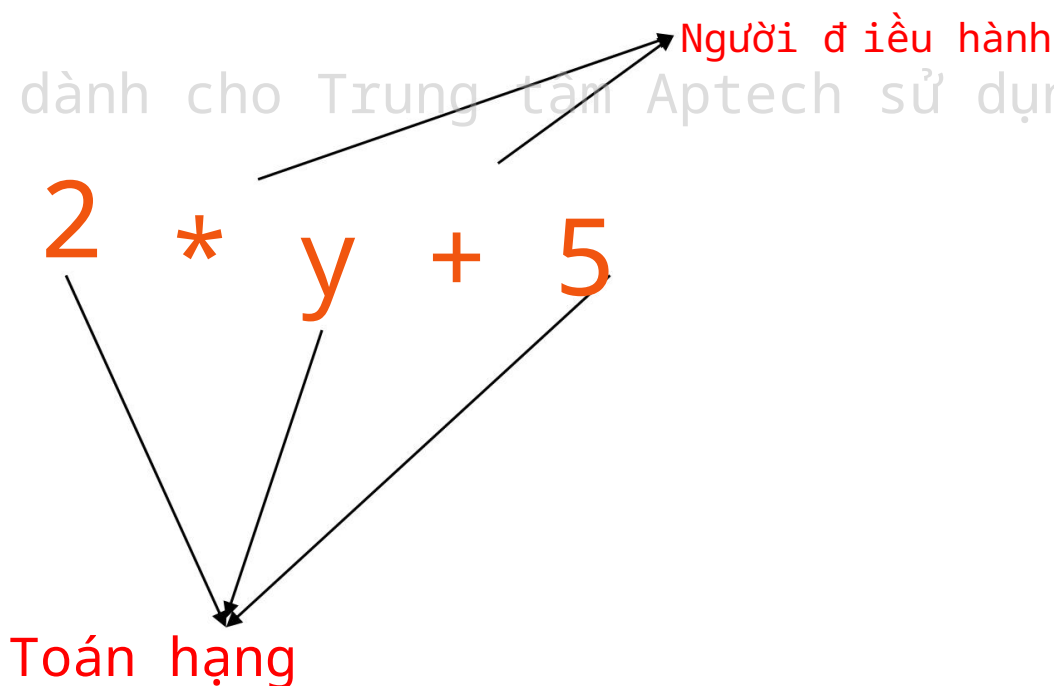


# Biểu thức

Sự kết hợp của các toán tử và toán hạng

Chỉ dành cho Trung tâm Aptech sử dụng

Ví dụ

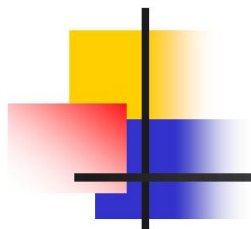


# Toán tử gán

Toán tử gán (=) có thể được sử dụng với bất kỳ biểu thức C hợp lệ nào



# Nhiều bài tập



Nhiều biến có thể được gán cùng một giá trị trong một câu lệnh duy nhất

Chỉ dành cho Trung tâm Artech sử dụng

```
a = b = c = 10;
```

Tuy nhiên, bạn không thể làm điều này:

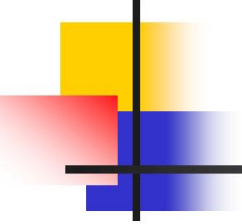
```
int a = int b = int b = int c = 10
```

**X**

# Người điều hành

## 4 loại

---



Số học

Hợp lý

Quan hệ

Bitwise



# Biểu thức số học

---

Biểu thức toán học có thể được thể hiện trong C bằng cách sử dụng các toán tử số học

Ví dụ Chỉ dành cho Trung tâm Aptech sử dụng

`++tôi % 7`

`5 + (c = 3 + 8)`

`Một * (b + c/d)22`

# Quan hệ & Logic

## Toán tử-1

Đã từng..

Kiểm tra mối quan hệ giữa hai biến hoặc giữa một biến và một hằng số

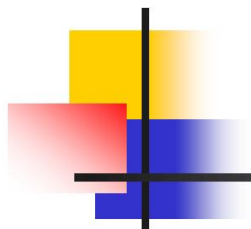
Chỉ dành cho Trung tâm Aptech sử dụng

Toán tử quan hệ

Operator	Relational Operators Action
>	Greater than lon hơn
>=	Greater than or equal
<	Less than
<=	Less than or equal
==	Equal bang
!=	Not equal không bằng



# Quan hệ & Logic Toán tử-2



Toán tử logic là các ký hiệu được sử dụng để kết hợp hoặc phủ định các biểu thức chứa toán tử quan hệ

Chỉ dành cho Trung tâm Công nghệ Sử dụng

Operator	Logical Operators Action
&&	AND
	OR
!	NOT

Ví dụ: nếu  $(a > 10) \ \&\& \ (a < 20)$

Biểu thức sử dụng toán tử logic trả về số không cho sai và 1 cho đúng

# Toán tử logic bitwise-1

Xử lý dữ liệu sau khi chuyển đổi số sang dạng nhị phân tương đương. (Biểu diễn từng bit)

VÀ ( SỐ 1 & SỐ 2 )	Trả về 1 nếu cả hai toán hạng đều là 1
HOẶC ( SỐ 1   SỐ 2 )	Trả về 1 nếu bit của bất kỳ toán hạng nào là 1
KHÔNG ( ~ SỐ 1 )	Đảo ngược các bit của toán hạng của nó (từ 0 đến 1 và từ 1 đến 0)
XOR ( SỐ1 ^ SỐ2 )	Trả về 1 nếu một trong hai bit trong toán hạng là 1 nhưng không phải cả hai

0 → 0  
0 → 0  
OR → 0

th mô rong của OR



# Toán tử logic bitwise-2

---

## Ví dụ

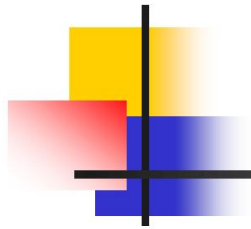
10 & 15    1010 & 1111    1010    10

Chỉ dành cho Trung tâm Aptech sử dụng

10 | 15    1010 | 1111    1111    15

10^15    1010^1111    0101    5

~ 10    ~1010    1011    -11



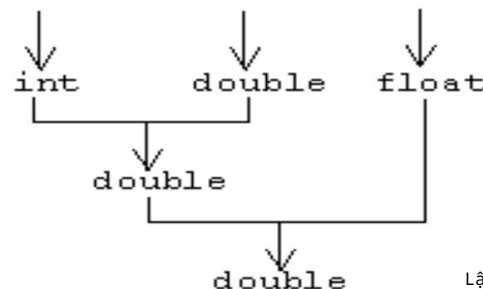
# Chuyển đổi loại

The **automatic type conversions** for evaluating an expression are tabulated below.

- char** and **short** are converted to **int** and **float** is converted to **double**.
- If either operand is **double**, the other is converted to **double**, and the result is **double**.
- If either operand is **long**, the other is converted to **long** the result is **double**.
- If either operand is **unsigned**, the other is also converted to **unsigned** and the result is also **unsigned**.
- Otherwise all that are left are the operands of type **int**, and the result is **int**.

Ví dụ

```
char ch;
int i;
float f;
double d;
result = (ch/i) + (f*d) - (f+i);
```





# Diễn viên

Một biểu thức có thể bị ép buộc phải có một kiểu nhất định bằng cách sử dụng ép kiểu.  
Cú pháp chung của ép kiểu:

(loại) **đ** **úc**

loại bất kỳ kiểu dữ liệu C hợp lệ nào

Ví dụ:

số thực x, f;

f = 3,14159;

x = (int) f; , giá trị của x sẽ là 3 (số nguyên)

Giá trị số nguyên được trả về bởi (int)f  
được chuyển đổi trở lại thành dấu chấm động  
khi nó vượt qua toán tử gán.  
Giá trị của f không thay đổi.



# Thứ tự ưu tiên của các toán tử-1

Thứ tự ưu tiên thiết lập thứ bậc của một tập hợp toán tử so với tập hợp khác khi một biểu thức số học được đánh giá

Nó đề cập đến thứ tự mà C đánh giá các toán tử

Thứ tự ưu tiên của các toán tử có thể được thay đổi bằng cách bao quanh các biểu thức trong ngoặc đơn

Lớp toán tử	Người điều hành	Tính liên kết
Đơn vị	- ++ --	Từ phải sang trái
nhị phân	$\wedge$	Từ trái sang phải
nhị phân	* / %	Từ trái sang phải
nhị phân	+ -	Từ trái sang phải
nhị phân	=	Từ phải sang trái



# Thứ tự ưu tiên của các toán tử-2

---

**Example :**  $-8 * 4 \% 2 - 3$

Chỉ dành cho Trung tâm Aptech sử dụng

Sequence	Operation done	Result
1.	- 8 (unary minus)	negative of 8
2.	- 8 * 4	- 32
3.	- 32 % 2	16
4.	16-3	13

# Ưu tiên giữa so sánh

Người điều hành

Luôn luôn được đánh giá từ trái sang phải

Chỉ dành cho Trung tâm Aptech sử dụng





# Ưu tiên cho Logic Toán tử-1



Ưu tiên	Người điều hành
1	KHÔNG
2	VÀ
3	HOẶC

Khi nhiều trường hợp của một toán tử logic được sử dụng trong một điều kiện, chúng được đánh giá từ phải sang trái

# Ưu tiên cho Logic Toán tử-2

Hãy xem xét biểu thức sau  
Sai HOẶC Đúng VÀ KHÔNG Sai VÀ Đúng

Tình trạng này được đánh giá như sau:

Sai HOẶC Đúng VÀ [KHÔNG Sai] VÀ Đúng

NOT có mức độ ưu tiên cao nhất.

Sai HOẶC Đúng VÀ [Đúng VÀ Đúng]

AND là toán tử có độ ưu tiên cao nhất và các toán tử có độ ưu tiên giống nhau được đánh giá từ phải sang trái

Sai HOẶC [Đúng VÀ Đúng]

[Sai HOẶC Đúng]

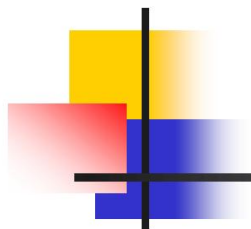
ĐÚNG VẬY

# Ưu tiên giữa Toán tử-1

Khi một phương trình sử dụng nhiều hơn một loại toán tử thì thứ tự ưu tiên phải được thiết lập với các loại toán tử khác nhau

Chỉ dành cho Trung tâm Aptech sử dụng	
Loại ưu tiên của	
	Người điều hành
1	Số học
2	So sánh
3	Hợp lý

# Ưu tiên giữa Toán tử-2



Hãy xem xét ví dụ sau:

$$2*3+4/2 > 3 \text{ VÀ } 3<5 \text{ HOẶC } 10<9$$

Đánh giá đ ược thể hiện như sau:

$$[2*3+4/2] > 3 \text{ VÀ } 3<5 \text{ HOẶC } 10<9$$

Đầu tiên các toán tử số học đ ược xử lý

$$[[2*3]+[4/2]] > 3 \text{ VÀ } 3<5 \text{ HOẶC } 10<9$$

Vì

$$[6+2] > 3 \text{ VÀ } 3<5 \text{ HOẶC } 10<9$$

$$[8 > 3] \text{ VÀ } [3 < 5] \text{ HOẶC } [10 < 9]$$

# Ưu tiên giữa Toán tử-3

Tiếp theo được đánh giá là các toán tử so sánh, tất cả đều có cùng thứ tự ưu tiên và do đó được đánh giá từ

trái sang phải

Chỉ dành cho Trung tâm Aptech sử dụng

Đúng VÀ Đúng HOẶC Sai

Toán tử logic là toán tử cuối cùng được đánh giá.

VÀ được ưu tiên hơn HOẶC

[Đúng VÀ Đúng] HOẶC Sai

Đúng hay Sai



# Thay đổi thứ tự ưu tiên-1

---

Dấu ngoặc đơn ( ) có mức độ ưu tiên cao nhất

Thứ tự ưu tiên của các toán tử có thể được sửa đổi bằng cách sử dụng dấu ngoặc đơn ( )

Toán tử có mức độ ưu tiên thấp hơn với dấu ngoặc đơn  
giả định quyền ưu tiên cao nhất và được thực hiện đầu tiên

Trong trường hợp ngoặc đơn lồng nhau ( ( ( ) ) ) thì  
ngoặc đơn trong cùng được đánh giá trước

Một biểu thức bao gồm nhiều cặp dấu ngoặc đơn được xử lý  
từ trái sang phải



# Thay đổi thứ tự ưu tiên-2

---

Hãy xem xét ví dụ sau:

$$5+9*3^2-4 > 10 \text{ VÀ } (2+2^4-8/4 > 6 \text{ HOẶC } (2<6 \text{ VÀ } 10>11))$$

Giải pháp là: 1.

$$5+9*3^2-4 > 10 \text{ VÀ } (2+2^4-8/4 > 6 \text{ HOẶC } (\text{Đúng VÀ Sai}))$$

Chỉ dành cho Trung tâm Aptech sử dụng

Dấu ngoặc đơn bên trong được ưu tiên hơn tất cả các toán tử khác và việc đánh giá bên trong dấu ngoặc đơn này tuân theo các quy ước thông thường

$$2. \quad 5+9*3^2-4 > 10 \text{ VÀ } (2+2^4-8/4 > 6 \text{ HOẶC Sai})$$



# Thay đổi thứ tự ưu tiên-3

---

3.  $5+9*3^2-4 > 10$  VÀ  $(2+16-8/4 > 6$  HOẶC Sai)

Tiếp theo, dấu ngoặc đơn bên ngoài được đánh giá

Chỉ dành cho Trung tâm Aptech sử dụng

4.  $5+9*3^2-4 > 10$  VÀ  $(2+16-2 > 6$  HOẶC Sai)

5.  $5+9*3^2-4 > 10$  VÀ  $(18-2 > 6$  HOẶC Sai)

6.  $5+9*3^2-4 > 10$  VÀ  $(16 > 6$  HOẶC Sai)

7.  $5+9*3^2-4 > 10$  VÀ (Đúng HOẶC Sai)

8.  $5+9*3^2-4 > 10$  VÀ Đúng





# Thay đổi thứ tự ưu tiên-4

---

9.  $5+9*9-4>10$  AND True Biểu thức

bên trái được đánh giá theo các quy ước 10.  $5+81-4>10$  AND True

11.  $86-4>10$  AND

True 12.  $82>10$  AND True 13. True

AND True 14. True