1.8 PHÉP TOÁN, BIỂU THỨC VÀ CÂU LỆNH

1.8.1 Phép toán

1.8.1.1 Các phép toán số học: +, -, *, /, %

- Các phép toán + (cộng), - (trừ), * (nhân) được hiểu theo nghĩa thông thường trong số học.

- Phép toán a / b (chia) được thực hiện theo kiểu của các toán hạng, tức nếu cả hai toán hạng là số nguyên thì kết quả của phép chia chỉ lấy phần nguyên, ngược lại nếu 1 trong 2 toán hạng là số thực thì kết quả là số thực.

Ví dụ:

```
13 / 5 = 2                              // vì 13 và 5 là 2 số nguyên
13.0 / 5 = 13 / 5.0 = 13.0 / 5.0 = 2.6  // vì có ít nhất 1 toán hạng là thực
```

- Phép toán a % b (lấy phần dư) trả lại phần dư của phép chia a / b, trong đó a và b là 2 số nguyên.

Ví dụ:

```
13 % 5 = 3    // phần dư của 13/5
5 % 13 = 5    // phần dư của 5/13
```

1.8.1.2 Các phép toán tự tăng, tự giảm: i++, ++i, i--, --i

- Phép toán ++i và i++ sẽ cùng tăng i lên 1 đơn vị tức tương đương với câu lệnh i = i+1. Tuy nhiên nếu 2 phép toán này nằm trong câu lệnh hoặc biểu thức thì ++i khác với i++. Cụ thể ++i sẽ tăng i, sau đó i mới được tham gia vào tính toán trong biểu thức, ngược lại i++ sẽ tăng i sau khi biểu thức được tính toán xong (với giá trị i cũ). Điểm khác biệt này được minh họa thông qua ví dụ sau với giả sử i = 3, j = 15.

Phép toán	**Tương đương**	**Kết quả**
i = ++j ; // tăng trước	j = j + 1; i = j	i = 16 , j = 16
i= j++ ; // tăng sau	i = j; j = j + 1	i = 15 , j = 16
j = ++i + 5 ;	i = i + 1 ; j = i + 5;	i = 4, j = 9
j = i++ + 5 ;	j = i + 5; i = i + 1;	i = 4, j = 8

1.8.1.3 Các phép toán so sánh và logic

Đây là các phép toán mà giá trị trả lại là đúng (true) hoặc sai (false). Mở rộng hơn C++ quan niệm một giá trị bất kỳ khác 0 là "đúng" và giá trị 0 là "sai".

- *Các phép toán so sánh:* == (bằng nhau), != (khác nhau), > (lớn hơn), < (nhỏ hơn), >= (lớn hơn hoặc bằng), <= (nhỏ hơn hoặc bằng).

Hai toán hạng của các phép toán này phải cùng kiểu.

- *Các phép toán logic:* && (và), || (hoặc), ! (không, phủ định)

Giá trị trả lại của phép toán là 1 (true) hoặc 0 (false) và được cho trong *bảng* sau:

A	B	A && B	A || B	! A
1	1	1	1	0
1	0	0	1	0
0	1	0	1	1
0	0	0	0	1

Ví dụ: Cho i = 2, j = 3, xét 2 biểu thức sau đây:

x = (++i < 4 && ++j < 5) cho kết quả x = 1 , i = 3 , j = 4

y = (i++ > 2 || ++j < 4) cho kết quả y = 0 , i = 3 , j = 4

1.8.2 Các phép gán

- Phép gán có điều kiện:

biến = (Điều_kiện) ? a : b ;

Điều_kiện là một biểu thức logic, a, b là các biểu thức bất kỳ cùng kiểu với kiểu của biến. Phép toán này gán giá trị a cho biến nếu điều kiện đúng và gán giá trị b cho biến nếu ngược lại.

Ví dụ:

```
x = (3 + 4 < 7) ? 10 : 20      // x = 20 vì 3 + 4 < 7 là sai
x = (3 + 4) ? 10 : 20          // x = 10 vì 3 + 4 khác 0, tức điều kiện đúng
x = (a > b) ? a : b            // x = số lớn hơn trong 2 số a, b.
```

- Cách viết gọn của phép gán: Một phép gán dạng x = x @ a ; có thể được viết gọn dưới dạng x @= a trong đó @ là các phép toán số học, xử lý bit ...

Ví dụ:

Thay cho viết x = x + 2 có thể viết x += 2;

hoặc x = x / 2; x = x * 2 có thể được viết lại như x /= 2; x *= 2;

1.8.3 Biểu thức

Biểu thức (expression) là dãy ký hiệu kết hợp giữa các toán hạng, phép toán và cặp dấu () theo một qui tắc nhất định. Các toán hạng có thể là hằng, biến, hàm.

Ví dụ:

(x + y) * 2 - 4

3 - x + sqrt(y)

(-b + sqrt(delta)) / (2*a)

1.8.3.1 Thứ tự ưu tiên của các phép toán

Để tính giá trị của một biểu thức cần có một trật tự tính toán cụ thể và thống nhất.

Ví dụ:

Xét biểu thức x = 3 + 4 * 2 + 7

- Nếu tính theo đúng trật tự từ trái sang phải, thì x = ((3 + 4) * 2) + 7 = 21
- Nếu ưu tiên dấu + được thực hiện trước dấu *, thì x = (3 + 4) * (2 + 7) = 63
- Nếu ưu tiên dấu * được thực hiện trước dấu +, thì x = 3 + (4 * 2) + 7 = 18

C++ qui định trật tự tính toán theo các mức độ ưu tiên như sau:

1. Các biểu thức trong cặp dấu ngoặc ()
2. Các phép toán 1 ngôi (tự tăng, tự giảm, lấy địa chỉ, lấy nội dung con trỏ, ...)
3. Các phép toán số học
4. Các phép toán so sánh, logic
5. Các phép gán

Nếu có nhiều cặp ngoặc lồng nhau thì cặp trong cùng (sâu nhất) được tính trước. Các phép toán trong cùng một lớp có độ ưu tiên theo thứ tự: lớp nhân (*, /, &&), lớp cộng (+, -, ||). Nếu các phép toán có cùng thứ tự ưu tiên thì chương trình sẽ thực hiện từ trái sang phải.

1.8.3.2 Phép chuyển đổi kiểu

- Chuyển kiểu tự động:

char <-> int <-> long long <-> double

Ví dụ:

```
int i = 3;
double f = i + 2;
```

Trong ví dụ trên i có kiểu nguyên và vì vậy i + 2 cũng có kiểu nguyên trong khi f có kiểu thực. Tuy nhiên phép toán gán này là hợp lệ vì chương trình sẽ tự động chuyển kiểu của i + 2 (bằng 5) sang kiểu thực (bằng 5.0) rồi mới gán cho f.

- *Ép kiểu:* Trong chuyển kiểu tự động, chương trình chuyển các kiểu từ thấp lên cao, tuy nhiên chiều ngược lại không thể thực hiện được vì nó có thể gây mất dữ liệu. Do đó nếu cần thiết phải ra lệnh cho chương trình chuyển kiểu từ cao xuống thấp.

Ví dụ:

```
int i;
double f = 3 ;  // tự động chuyển 3 thành 3.0 và gán cho f
i = f + 2 ;        // sai (báo lỗi) mặc dù f + 2 = 5 nhưng không gán được cho i
```

Trong ví dụ trên để câu lệnh i = f + 2 thực hiện được (không báo lỗi) ta phải ép kiểu của biểu thức f + 2 về thành kiểu nguyên. Cú pháp tổng quát như sau:

(tên_kiểu)biểu_thức

hoặc:

tên_kiểu(biểu_thức)

lưu ý: không nên có dấu cách trước "(", sau dấu ")"

ví dụ:

int a = 7, b = 2;

cout << a / b << endl; //kết quả là: 3

cout << double(a) / b; //kết quả là: 3.5

1.8.4 Câu lệnh và khối lệnh

Các ví dụ nhập/xuất hoặc các phép gán tạo thành những câu lệnh đơn giản như:

```
cin >> x >> y ;
x = x + 1 ;
y = sqrt(x) ;
cout << x << endl;
cout << y << endl;
```

Một số câu lệnh được gọi là lệnh có cấu trúc, tức là bên trong nó lại chứa dãy lệnh khác. Dãy lệnh này phải được bao giữa cặp dấu ngoặc {} và được gọi là khối lệnh.

1.9 MỘT SỐ HÀM THÔNG DỤNG

1.9.1 Nhóm hàm toán học

Các hàm này đều được khai báo trong file nguyên mẫu ***math.h***.

- abs(x), labs(x), fabs(x) : trả lại giá trị tuyệt đối của một số nguyên, số nguyên dài và số thực.
- pow(x, y) : hàm mũ, trả lại giá trị x lũy thừa y (x^y).
- exp(x) : hàm mũ, trả lại giá trị e mũ x (e^x).
- log(x), log10(x) : trả lại lôgarit cơ số e và logarit thập phân của x (lnx, logx) .
- sqrt(x) : trả lại căn bậc 2 của x.
- sin(x), cos(x), tan(x), asin(x), acos(x), atan(x) : trả lại các giá trị sinx, cosx, tgx, arcsinx, arccosx, arctgx.

1.9.2 Nhóm hàm kiểm tra ký tự

Các hàm này đều được khai báo trong file nguyên mẫu ***ctype.h***.

- isalnum(kt) : kiểm tra ký tự kt có phải là chữ cái hoặc chữ số hay không.
- isalpha(kt) : kiểm tra ký tự kt có phải là chữ cái hay không.
- isdigit(kt) : kiểm tra ký tự kt có phải là chữ số hay không.
- islower(kt) : kiểm tra ký tự kt có phải là chữ cái thường (a .. z) hay không.
- isupper(kt) : kiểm tra ký tự kt có phải là chữ cái hoa (A .. Z) hay không.
- isspace(kt) : kiểm tra ký tự kt có phải là ký tự trống hay không.

1.9.3 Nhóm hàm chuyển đổi dữ liệu

Các hàm này đều được khai báo trong file nguyên mẫu ***ctype.h***.

- toupper(c) : chuyển từ chữ thường sang chữ hoa.
- tolower(c) : chuyển từ chữ hoa sang chữ thường.
- stof(s): chuyển xâu s sang giá trị float.
- stod(s): chuyển xau s sang giá trị double.
- stoi(s): chuyển chuỗi s sang giá trị int.
- stoll(s): chuyển chuỗi s sang giá trị long long.
- to_string(n): chuyển số n về xâu (n là số thực hoặc nguyên)