

Toán tử trong Java

Java cung cấp rất nhiều toán tử đa dạng để thao tác với các biến. Chúng ta có thể chia tất cả các toán tử trong Java thành các nhóm sau:

- Toán tử số học
- Toán tử quan hệ
- Toán tử thao tác bit
- Toán tử logic
- Toán tử gán
- Và các toán tử hỗn hợp

Toán tử số học trong Java

Các toán tử số học được sử dụng trong các biểu thức toán học theo cách tương tự như chúng được sử dụng trong đại số học. Bảng sau liệt kê các toán tử số học trong Java.

Giả sử biến A giữ giá trị 10, biến B giữ giá trị 20, thì:

Ví dụ

Toán tử	Miêu tả	Ví dụ
+	Phép cộng	A + B sẽ cho kết quả 30
-	Phép trừ: trừ toán hạng trái cho toán hạng phải	A - B sẽ cho kết quả -10
*	Phép nhân	A * B sẽ cho kết quả 200
/	Phép chia: chia toán hạng trái cho toán hạng phải	B / A sẽ cho kết quả 2

%	Phép chia lấy phần dư: Lấy phần dư của phép chia toán hạng trái cho toán hạng phải	B % A sẽ cho kết quả 0
++	Phép lượng gia: lượng gia giá trị toán hạng thêm 1	B++ sẽ cho kết quả 21
--	Phép lượng giảm: lượng giảm giá trị toán hạng đi 1	B-- sẽ cho kết quả 19

Toán tử quan hệ trong Java

Bảng dưới liệt kê các toán tử quan hệ được hỗ trợ bởi Java.

Giả sử biến A giữ giá trị 10, biến B giữ giá trị 20, thì:

Ví dụ

Toán tử	Miêu tả	Ví dụ
==	Kiểm tra nếu giá trị của hai toán hạng có cân bằng hay không, nếu có thì điều kiện là true.	(A == B) là không true.
!=	Kiểm tra nếu giá trị hai toán hạng là cân bằng hay không, nếu không cân bằng, thì điều kiện là true	(A != B) là true.
>	Kiểm tra nếu toán hạng trái có lớn hơn toán hạng phải hay không, nếu có thì điều kiện là true	(A > B) là không true.
<	Kiểm tra nếu toán hạng phải có lớn hơn toán hạng trái hay không, nếu có thì điều kiện là true	(A < B) là true.
>=	Kiểm tra nếu toán hạng trái có lớn hơn hoặc bằng toán hạng phải hay không, nếu có thì điều kiện là true	(A >= B) là không true.
<=	Kiểm tra nếu toán hạng phải có lớn hơn hoặc bằng toán hạng trái hay không, nếu có thì điều kiện là true	(A <= B) là true.

Toán tử thao tác bit trong Java

Java định nghĩa một số toán tử thao tác bit có thể được áp dụng cho các kiểu giá trị integer, long, int, short, char, và byte.

Toán tử thao tác bit làm việc trên các bit. Giả sử nếu a = 60 và b = 13, thì trong định dạng nhị phân chúng sẽ như sau:

a = 0011 1100

b = 0000 1101

---

a&b = 0000 1100

a|b = 0011 1101

a^b = 0011 0001

~a = 1100 0011

Bảng dưới đây liệt kê các toán tử bit được hỗ trợ trong Java:

Giả sử biến A giữ giá trị 60 và biến B giữ 13 thì khi đó:

Ví dụ

Toán tử	Miêu tả	Ví dụ
&	Toán tử Và nhị phân sao chép một bit tới kết quả nếu nó tồn tại trong cả hai toán hạng	(A & B) sẽ cho kết quả 12, hay là 0000 1100
|	Toán tử Hoặc nhị phân sao chép một bit tới kết quả nếu nó tồn tại trong một hoặc hai toán hạng	(A | B) sẽ cho kết quả 61, hay là 0011 1101

^	Toán tử Hoặc loại trừ nhị phân sao chép bit nếu nó được thiết lập trong một toán hạng nhưng không phải trong cả hai	(A ^ B) sẽ cho kết quả 49, hay là 0011 0001
~	Toán tử đảo bit là toán tử một ngôi. Đảo bít 1 thành 0 và ngược lại	(~A) sẽ cho kết quả -61, hay là 1100 0011
<<	Toán tử dịch trái. Giá trị toán hạng trái được dịch chuyển sang trái bởi số các bit được xác định bởi toán hạng bên phải.	A << 2 sẽ cho kết quả 240, hay là 1111 0000
>>	Toán tử dịch phải. Giá trị toán hạng trái được dịch chuyển sang phải bởi số các bit được xác định bởi toán hạng bên phải	A >> 2 sẽ cho kết quả 15, hay là 1111
>>>	Toán tử dịch phải và điền 0 vào chỗ trống	A >>>2 sẽ cho kết quả 15, hay là 0000 1111

Toán tử logic trong Java

Bảng dưới liệt kê đầy đủ các toán tử logic trong Java:

Giả sử biến A giữ true và biến B giữ false thì khi đó:

Ví du

Toán tử	Miêu tả	Ví dụ
&&	Toán tử Và logic. Nếu cả hai toán hạng là khác không, thì khi đó điều kiện là true	(A && B) là false.
||	Toán tử Hoặc logic. Nếu một trong hai toán tử khác 0, thì điều kiện là true	(A || B) là true.

!	Toán tử Phủ định logic. Sử dụng để đảo ngược lại trạng thái logic của toán hạng đó. Nếu điều kiện toán hạng là true thì phủ định nó sẽ là false	!(A && B) là true.

Các toán tử gán trong Java

Dưới đây liệt kê các toán tử gán được hỗ trợ bởi Java:

Ví dụ

Toán tử	Miêu tả	Ví dụ
=	Toán tử gán đơn giản. Gán giá trị toán hạng bên phải cho toán hạng trái.	C = A + B sẽ gán giá trị của A + B vào cho C
+=	Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái.	C += A là tương đương với C = C + A
-=	Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái.	C -= A là tương đương với C = C - A
*=	Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái.	C *= A là tương đương với C = C * A
/=	Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng trái.	C /= A là tương đương với C = C / A
%=	Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái.	C %= A là tương đương với C = C % A

<<=	Dịch trái toán hạng trái sang số vị trí là giá trị toán hạng phải.	C <<= 2 là giống như C = C << 2
>>=	Dịch phải toán hạng trái sang số vị trí là giá trị toán hạng phải.	C >>= 2 là giống như C = C >> 2
&=	Phép AND bit	C &= 2 là giống như C = C & 2
^=	Phép OR loại trừ bit	C ^= 2 là giống như C = C ^ 2
|=	Phép OR bit.	C |= 2 là giống như C = C | 2

Toán tử hỗn hợp trong Java

Ngôn ngữ Java cũng hỗ trợ một số toán tử hỗn hợp khác.

Toán tử điều kiện (? :)

Toán tử này gồm ba toán hạng và được sử dụng để ước lượng các biểu thức quan hệ. Mục tiêu của toán tử là quyết định giá trị nào sẽ được gán cho biến. Toán tử này được viết như sau:

```
bien x = (bieu_thuc) ? giatri neu true : giatri neu false
```

Sau đây là ví dụ:

```
public class Test {


   public static void main(String args[]){
      int a , b;
      a = 10;
      b = (a == 1) ? 20: 30; //Day la vi du ve toan tu dieu kien.
      System.out.println( "Gia tri cua b la : " +  b );


      b = (a == 10) ? 20: 30; //Day la vi du ve toan tu dieu kien.
      System.out.println( "Gia tri cua b la : " + b );
   }
```

```
}
```

Nó sẽ cho kết quả sau:

```
Gia tri cua b la : 30
Gia tri cua b la : 20
```

Toán tử instanceof trong Java

Toán tử này chỉ được sử dụng cho các biến tham chiếu đối tượng. Toán tử kiểm tra có hay không đối tượng là một kiểu cụ thể (kiểu class hoặc kiểu interface). Toán tử instanceof được viết như sau:

```
( Object reference variable ) instanceof  (class/interface type)
```

Toán tử trả về true nếu toán hạng trái là biến thể hiện của toán hạng phải. Ví dụ:

```
public class Test {


   public static void main(String args[]){
      String name = "Doan";
      // Duoi day se tra ve true neu name la mot kieu String
      boolean result = name instanceof String;
      System.out.println( result );
   }
}
```

Nó sẽ cho kết quả:

```
true
```

Toán tử này sẽ cũng trả về true nếu đối tượng đang được so sánh là tham số tương thích với kiểu toán hạng phải. Dưới đây là một ví dụ khác:

```
class Animal {}


public class Dog extends Animal {
   public static void main(String args[]){
      Animal a = new Dog();
      boolean result =  a instanceof Dog;
```

```
      System.out.println( result );
   }
}
```

Nó sẽ cho kết quả:

```
true
```

Thứ tự ưu tiên của các toán tử trong Java:

Thứ tự ưu tiên của các toán tử xác định cách biểu thức được tính toán. Ví dụ: toán tử nhân có quyền ưu tiên hơn toán tử cộng.

Ví dụ, x = 7 + 3 * 2; ở đây, x được gán giá trị 13, chứ không phải 20 bởi vì toán tử * có quyền ưu tiên cao hơn toán tử +, vì thế đầu tiên nó thực hiện phép nhân 3 * 2 và sau đó thêm với 7.

Bảng dưới đây liệt kê thứ tự ưu tiên của các toán tử. Các toán tử với quyền ưu tiên cao nhất xuất hiện trên cùng của bảng, và các toán tử có quyền ưu tiên thấp nhất thì ở bên dưới cùng của bảng. Trong một biểu thức, các toán tử có quyền ưu tiên cao nhất được tính toán đầu tiên.

Loại	Toán tử	Thứ tự ưu tiên
Postfix	() [] . (toán tử dot)	Trái sang phải
Unary	++ - - ! ~	Phải sang trái
Tính nhân	* / %	Trái sang phải
Tính cộng	+ -	Trái sang phải
Dịch chuyển	>> >>> <<	Trái sang phải
Quan hệ	> >= < <=	Trái sang phải

Cân bằng	== !=	Trái sang phải
Phép AND bit	&	Trái sang phải
Phép XOR bit	^	Trái sang phải
Phép OR bit	|	Trái sang phải
Phép AND logic	&&	Trái sang phải
Phép OR logic	||	Trái sang phải
Điều kiện	?:	Phải sang trái
Gán	= += -= *= /= %= >>= <<= &= ^= |=	Phải sang trái
Dấu phảy	,	Trái sang phải