عملگرهای بیتی به شما اجازه میدهند که شکل باینری انواع دادهها را دستکاری کنید. برای درک بهتر این درس توصیه میشود که شما سیستم باینری و نحوه تبدیل اعداد دهدهی به باینری را یاد بگیرید. در سیستم باینری (دودویی) که کامپیوتر از آن استفاده میکند وضعیت هر چیز یا خاموش است یا روشن. برای نشان دادن حالت روشن از عدد 1 و برای نشان دادن حالت خاموش از عدد 0 استفاده میشود. بنابراین اعداد باینری فقط میتوانند صفر یا یک باشند. اعداد باینری را اعداد در مبنای 2 و اعداد اعشاری را اعداد در مبنای 10 می گویند. یک بیت نشان دهنده یک رقم باینری است و هر بایت نشان دهنده 8 بیت است. به عنوان مثال برای یک داده از نوع int به 32 بیت یا 4 بایت فضا برای ذخیره آن نیاز داریم، این بدین معناست که اعداد از 32 رقم 0 و 1 برای ذخیره استفاده میکنند. برای مثال عدد 100 وقتی به عنوان یک متغیر از نوع int ذخیره میشود در کامپیوتر به صورت زیر خوانده میشود : 0000000000000000000000000000001100100 عدد 100 در مبنای ده معادل عدد 100100 در مبنای 2 است. در اینجا 7 رقم سمت راست نشان دهنده عدد 100 در مبنای 2 است و مابقی صفرهای سمت راست برای پر کردن بیتهایی است که عدد از نوع int نیاز دارد. به این نکته توجه کنید که اعداد باینری از سمت راست به چپ خوانده میشوند. عملگرهای بیتی جاوا در جدول زیر نشان داده شدهاند : مثال عملگر بیتی AND 2 x = y & z; Binary

عملگرهای بیتی

X

0

0

دست میآید:

عملگر بیتی(OR(J)

X

0

0

توجه کنید:

است؟

عملگر بیتی (۸)XOR

0

0

جدول درستی این عملگر در زیر آمده است:

XOR را بر روی دو مقدار مشاده میکنید:

عملگر بیتی (~) NOT

X

در زیر معادل باینری اعداد بالا (5 و 7) نشان داده شده است.

0

0

با نگاه کردن به جدول درستی عملگر بیتی XOR ، میتوان فهمید که چرا نتیجه عدد 2 میشود.

این عملگر یک عملگر یگانی است و فقط به یک عملوند نیاز دارد. در زیر جدول درستی این عملگر آمده است:

عملگر بیتی NOT مقادیر بیتها را معکوس میکند. در زیر چگونگی استفاده از این عملگر آمده است :

به نمایش باینری مثال بالا که در زیر نشان داده شده است توجه نمایید.

تغییر مکان به سمت چپ

تغییر مکان به سمت راست

عملگر بیتی تغییر مکان (shift)

عملگر تغییر مکان به سمت چپ

صفر به سمت راست اضافه می شود.

عملگر تغییر مکان به سمت راست

به عنوان مثال:

مورد بررسی قرار دهیم:

عملگر

NOTX

این نوع عملگرها به شما اجازه میدهند که بیتها را به سمت چپ یا راست جا به جا کنید. دو نوع عملگر بیتی تغییر مکان وجود دارد که هر کدام دو عملوند

دسته

Binary

Binary

قبول میکنند. عملوند سمت چپ این عملگرها حالت باینری یک مقدار و عملوند سمت راست تعداد جابه جاییبیت ها را نشان میدهد.

این عملگر، بیتهای عملوند سمت چپ را به تعداد n مکان مشخص شده، توسط عملوند سمت راست، به سمت چپ منتقل میکند. به عنوان مثال :

مشاهده میکنید که همه بیتها به اندازه دو واحد به سمت چپ منتقل شدهاند. در این انتقال دو صفر از صفرهای سمت چپ کم میشود و در عوض دو

با استفاده از عملگرتغییر مکان به سمت راست بیتهای مقدار 100 را به اندازه 4 واحد به سمت چپ جا به جا میکنیم. اجازه بدهید تأثیر این جا به جایی را

هر بیت به اندازه 4 واحد به سمت راست منتقل میشود، بنابراین 4 بیت اول سمت راست حذف شده و چهار صفر به سمت چپ اضافه میشود.

در مثال بالا، ما بیتهای مقدار 10 را دو مکان به سمت چپ منتقل کردهایم، حال بیایید تأثیر این انتقال را بررسی کنیم :

این عملگر شبیه به عمگر تغییر مکان به سمت چپ است با این تفاوت که بیتها را به سمت راست جا به جا میکند.

0

	بیتی OR	Binary	x = y z;	
	بیتی XOR	Binary	$x = y \wedge z$;	
	بیتی NOT	Unary	x = ~y;	
	AND Assignment بيتى	Binary	x &= y;	
	OR Assignment بيتى	Binary	x = y;	
	بیتی XOR Assignment	Binary	x ^= y;	
(2)	AND			

عملگر بیتی (AND(& عملگر بیتی AND کاری شبیه عملگر منطقی AND انجام میدهد با این تفاوت که این عملگر بر روی بیتها کار میکند. اگر مقادیر دو طرف آن I باشد مقدار I را بر میگرداند و اگر یکی یا هر دو طرف آن صفر باشد مقدار صفر را بر میگرداند. جدول درستی عمگر بیتی AND در زیر آمده است: X AND Y Y 1 0 0

0

0

همانطور که در مثال بالا مشاهده میکنید نتیجه عملکرد عملگر AND بر روی دو مقدار 5 و 3 عدد یک میشود. اجازه بدهید ببینیم که چطور این نتیجه را به

ابتدا دو عدد 5 و 3 به معادل باینریشان تبدیل میشوند. از آنجاییکه هر عدد صحیح(int) 32 بیت است از صفر برای پر کردن بیتهای خالی استفاده

X OR Y

0

نتیجه عملگر بیتی OR در صورتی صفر است که عملوندهای دو طرف آن صفر باشند. اگر فقط یکی از دو عملوند یک باشد نتیجه یک خواهد شد. به مثال زیر

وقتی که از عملگر بیتی OR برای دو مقدار در مثال بالا (7 و 9) استفاده میکنیم نتیجه 15 میشود. حال بررسی میکنیم که چرا این نتیجه به دست آمده

X XOR Y

0

1

1

0

در صورتیکه عملوندهای دو طرف این عملگر هر دو صفر یا هر دو یک باشند نتیجه صفر در غیر اینصورت نتیجه یک میشود. در مثال زیر تأثیر عملگر بیتی

با استفاده از جدول درستی عملگر بیتی OR میتوان نتیجه استفاده از این عملگر را تشخیص داد. عدد ۱۱۱۱ باینری معادل عدد 15 صحیح است.

اگر مقادیر دو طرف عملگر بیتی OR هر دو صفر باشند نتیجه صفر در غیر اینصورت ۱ خواهد شد. جدول درستی این عملگر در زیر آمده است :

int result = 5 & 3;

int result = 7 | 9;

int result = 5 ^ 7;

int result = ~7;

System.out.println(result);

7: 000000000000000000000000000000111

-8: 111111111111111111111111111111000

مثال

x = y << 2

x = y >> 2

int result = 10 << 2;

int result = 100 >> 4;

System.out.println(result);

100: 0000000000000000000000000001100100

System.out.println(result);

40: 00000000000000000000000000000101000

System.out.println(result);

7: 0000000000000000000000000000000111

15

System.out.println(result);

7: 0000000000000000000000000000000111

15: 0000000000000000000000000000001111

5:

System.out.println(result);

0000000000000000000000000000000011

1

0

میکنیم. با استفاده از جدول درستی عملگر بیتی AND میتوان فهمید که چرا نتیجه عدد یک میشود.

0

0