

## Bitwise right shift operators in Java

In C/C++ there is only one right shift operator '>>' which should be used only for positive integers or unsigned integers. Use of right shift operator for negative numbers is not recommended in C/C++, and when used for negative numbers, output is compiler dependent (See [this](#)). Unlike C++, Java supports following two right shift operators.

**1) >> (Signed right shift)** In Java, the operator '>>' is signed right shift operator. All integers are signed in Java, and it is fine to use >> for negative numbers. The operator '>>' uses the sign bit (left most bit) to fill the trailing positions after shift. If the number is negative, then 1 is used as a filler and if the number is positive, then 0 is used as a filler. For example, if binary representation of number is 10....100, then right shifting it by 2 using >> will make it 11.....1. See following Java programs as example '>>'

```
class Test {
    public static void main(String args[]) {
        int x = -4;
        System.out.println(x>>1);
        int y = 4;
        System.out.println(y>>1);
    }
}
```

Output:

```
-2
2
```

**2) >>> (Unsigned right shift)** In Java, the operator '>>>' is unsigned right shift operator. It always fills 0 irrespective of the sign of the number.

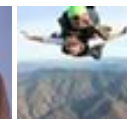
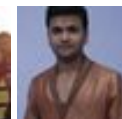
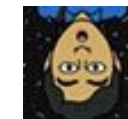
Google™ Custom Search



GeeksforGeeks



53,527 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

```

class Test {
    public static void main(String args[]) {

        // x is stored using 32 bit 2's complement form.
        // Binary representation of -1 is all 1s (111..1)
        int x = -1;

        System.out.println(x>>>29); // The value of 'x>>>29' is 00...0
        System.out.println(x>>>30); // The value of 'x>>>30' is 00...0
        System.out.println(x>>>31); // The value of 'x>>>31' is 00...0
    }
}

```

Output:

```

7
3
1

```

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

**Deploy Early.  
Deploy Often.**

DevOps from Rackspace:  
**Automation**

[FIND OUT HOW ►](#)

**rackspace**  
the open cloud company

**HIGH-PERFORMANCE COMPUTING ON A UNIVERSITY BUDGET**

Define your ideal server

**Download the infographic**

**SERVERSDIRECT**

## Popular Posts

[All permutations of a given string](#)

[Memory Layout of C Programs](#)

[Understanding "extern" keyword in C](#)

[Median of two sorted arrays](#)

[Tree traversal without recursion and without stack!](#)

[Structure Member Alignment, Padding and](#)

[Data Packing](#)

[Intersection point of two Linked Lists](#)

[Lowest Common Ancestor in a BST.](#)

[Check if a binary tree is BST or not](#)

[Sorted Linked List to Balanced BST](#)

## Related Topics:

- Inner class in java
- How to compare two arrays in Java?
- Can we override private methods in Java?
- Can we Overload or Override static methods in java ?
- Static class in Java
- Do we need forward declarations in Java?
- Checked vs Unchecked Exceptions in Java
- Private and final methods in Java



3



0



0

Writing code in comment? Please use [ideone.com](https://ideone.com) and share the link here.

5 Comments

GeeksforGeeks

Sort by Newest ▾



Join the discussion...



**Pardeep Shergill** · 2 months ago

why we don't have a signed left shift operator ?

^ | ▾ · Reply · Share ›



**WhoAml** · 8 months ago

class Tester

```
{
public static void main(String[] args)
{
byte i = -1;
i >>>= 1;
System.out.print(i);
}
```

Market research  
that's fast and  
accurate.

Get \$75 off

 Google consumer surveys

```
}  
}
```

Why the output of above is -1?

1 ^ | v • Reply • Share ›



**Prabhjot** → WhoAmI • 2 months ago

>>> operator is valid only for integers. So when you do "-1" >>> 1. it takes -1 in int form which is 1....11 {32 times} and the result of the op in decimal.

but since i is of type int, previous 24 bits are truncated and you are left decimal.

Hope that helps

^ | v • Reply • Share ›



**Zack** → WhoAmI • 3 months ago

I found that byte actually is 32 bit in Java.

try:

```
byte i = -1;
```

```
i >>>= 25;
```

```
System.out.print(i);
```

the output is 127(1111111), so it means that i is actually 111...111(32 b

^ | v • Reply • Share ›



**puneet k agarwal** → Zack • 19 days ago

Any negative number been stored has signbit+ 2's complement. unsigned operator it will change the signed bit 1 -> 0 that mean -1 will be represented as 1 (signed bit) (1---1 7 times(byte is 1 b

705



Subscribe

## Recent Comments

Abhi You live US or India?

[Google \(Mountain View\) interview](#) · 40 minutes ago

**Aman** Hi, Why arent we checking for conditions...

[Write a C program to Delete a Tree.](#) · 1 hour ago

kzs please provide solution for the problem...

[Backtracking | Set 2 \(Rat in a Maze\)](#) · 1 hour ago

**Sanjay Agarwal** bool

tree::Root\_to\_leaf\_path\_given\_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

[Count trailing zeroes in factorial of a number](#) · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...

[Find subarray with given sum](#) · 2 hours ago

AdChoices ▶


▶ [Java Source Code](#)

▶ [Java Operators](#)

▶ [C++ Java](#)

unsigned shift on this, will give us 0(signed bit) (1-----1 7 times)  
would help :)


^ | v • Reply • Share ›

AdChoices 

▶ [Java Void](#)

▶ [String Java](#)

▶ [32 Bit Java](#)

AdChoices 

▶ [Java Public](#)

▶ [Exception Java](#)

▶ [Java Args](#)

 [Subscribe](#)

 [Add Disqus to your site](#)

@geeksforgeeks, **Some rights reserved**

[Contact Us!](#)

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team