compact1, compact2, compact3

java.math

Class BigInteger

java.lang.Object
    java.lang.Number
        java.math.BigInteger

All Implemented Interfaces:

Serializable, Comparable<BigInteger>

```
public class BigInteger
extends Number
implements Comparable<BigInteger>
```

Immutable arbitrary-precision integers. All operations behave as if BigIntegers were represented in two's-complement notation (like Java's primitive integer types). BigInteger provides analogues to all of Java's primitive integer operators, and all relevant methods from java.lang.Math. Additionally, BigInteger provides operations for modular arithmetic, GCD calculation, primality testing, prime generation, bit manipulation, and a few other miscellaneous operations.

Semantics of arithmetic operations exactly mimic those of Java's integer arithmetic operators, as defined in *The Java Language Specification*. For example, division by zero throws an `ArithmeticException`, and division of a negative by a positive yields a negative (or zero) remainder. All of the details in the Spec concerning overflow are ignored, as BigIntegers are made as large as necessary to accommodate the results of an operation.

Semantics of shift operations extend those of Java's shift operators to allow for negative shift distances. A right-shift with a negative shift distance results in a left shift, and vice-versa. The unsigned right shift operator (>>>) is omitted, as this operation makes little sense in combination with the "infinite word size" abstraction provided by this class.

Semantics of bitwise logical operations exactly mimic those of Java's bitwise integer operators. The binary operators (`and`, `or`, `xor`) implicitly perform sign extension on the shorter of the two operands prior to performing the operation.

Comparison operations perform signed integer comparisons, analogous to those performed by Java's relational and equality operators.

Modular arithmetic operations are provided to compute residues, perform exponentiation, and compute multiplicative inverses. These methods always return a non-negative result, between `0` and `(modulus - 1)`, inclusive.

Bit operations operate on a single bit of the two's-complement representation of their operand. If necessary, the operand is sign- extended so that it contains the designated bit. None of the single-bit operations can produce a BigInteger with a different sign from the BigInteger being operated on, as they affect only a single bit, and the "infinite word size" abstraction provided by this class ensures that there are infinitely many "virtual sign bits"

preceding each BigInteger.

For the sake of brevity and clarity, pseudo-code is used throughout the descriptions of BigInteger methods. The pseudo-code expression `(i + j)` is shorthand for "a BigInteger whose value is that of the BigInteger `i` plus that of the BigInteger `j`." The pseudo-code expression `(i == j)` is shorthand for "`true` if and only if the BigInteger `i` represents the same value as the BigInteger `j`." Other pseudo-code expressions are interpreted similarly.

All methods and constructors in this class throw `NullPointerException` when passed a null object reference for any input parameter. BigInteger must support values in the range $-2^{\texttt{Integer.MAX\_VALUE}}$ (exclusive) to $+2^{\texttt{Integer.MAX\_VALUE}}$ (exclusive) and may support values outside of that range. The range of probable prime values is limited and may be less than the full supported positive range of `BigInteger`. The range must be at least 1 to $2^{500000000}$.

Implementation Note:

`BigInteger constructors and operations throw ArithmeticException when the result is out of the supported range of` $-2^{\texttt{Integer.MAX\_VALUE}}$ `(exclusive) to` $+2^{\texttt{Integer.MAX\_VALUE}}$ `(exclusive).`

Since:

`JDK1.1`

See Also:

`BigDecimal`, `Serialized Form`

Field Summary

Fields

Modifier and Type	Field and Description
`static BigInteger`	**`ONE`** The BigInteger constant one.
`static BigInteger`	**`TEN`** The BigInteger constant ten.
`static BigInteger`	**`ZERO`** The BigInteger constant zero.

Constructor Summary

Constructors

Constructor and Description
`BigInteger``(byte[] val)` Translates a byte array containing the two's-complement binary representation of a BigInteger into a BigInteger.
`BigInteger``(int signum, byte[] magnitude)` Translates the sign-magnitude representation of a BigInteger into a BigInteger.

BigInteger(int bitLength, int certainty, **Random** rnd)
Constructs a randomly generated positive BigInteger that is probably prime, with the specified bitLength.

BigInteger(int numBits, **Random** rnd)
Constructs a randomly generated BigInteger, uniformly distributed over the range 0 to ($2^{numBits}$ - 1), inclusive.

BigInteger(**String** val)
Translates the decimal String representation of a BigInteger into a BigInteger.

BigInteger(**String** val, int radix)
Translates the String representation of a BigInteger in the specified radix into a BigInteger.

Method Summary

All Methods Static Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
BigInteger	**abs**() Returns a BigInteger whose value is the absolute value of this BigInteger.
BigInteger	**add**(**BigInteger** val) Returns a BigInteger whose value is (this + val).
BigInteger	**and**(**BigInteger** val) Returns a BigInteger whose value is (this & val).
BigInteger	**andNot**(**BigInteger** val) Returns a BigInteger whose value is (this & ~val).
int	**bitCount**() Returns the number of bits in the two's complement representation of this BigInteger that differ from its sign bit.
int	**bitLength**() Returns the number of bits in the minimal two's-complement representation of this BigInteger, *excluding* a sign bit.
byte	**byteValueExact**() Converts this BigInteger to a byte, checking for lost information.
BigInteger	**clearBit**(int n) Returns a BigInteger whose value is equivalent to this BigInteger with the designated bit cleared.
int	**compareTo**(**BigInteger** val) Compares this BigInteger with the specified BigInteger.
BigInteger	**divide**(**BigInteger** val)

	Returns a BigInteger whose value is `(this / val)`.
`BigInteger[]`	**`divideAndRemainder(BigInteger val)`** Returns an array of two BigIntegers containing `(this / val)` followed by `(this % val)`.
`double`	**`doubleValue()`** Converts this BigInteger to a `double`.
`boolean`	**`equals(Object x)`** Compares this BigInteger with the specified Object for equality.
`BigInteger`	**`flipBit(int n)`** Returns a BigInteger whose value is equivalent to this BigInteger with the designated bit flipped.
`float`	**`floatValue()`** Converts this BigInteger to a `float`.
`BigInteger`	**`gcd(BigInteger val)`** Returns a BigInteger whose value is the greatest common divisor of `abs(this)` and `abs(val)`.
`int`	**`getLowestSetBit()`** Returns the index of the rightmost (lowest-order) one bit in this BigInteger (the number of zero bits to the right of the rightmost one bit).
`int`	**`hashCode()`** Returns the hash code for this BigInteger.
`int`	**`intValue()`** Converts this BigInteger to an `int`.
`int`	**`intValueExact()`** Converts this `BigInteger` to an `int`, checking for lost information.
`boolean`	**`isProbablePrime(int certainty)`** Returns `true` if this BigInteger is probably prime, `false` if it's definitely composite.
`long`	**`longValue()`** Converts this BigInteger to a `long`.
`long`	**`longValueExact()`** Converts this `BigInteger` to a `long`, checking for lost information.
`BigInteger`	**`max(BigInteger val)`** Returns the maximum of this BigInteger and `val`.
`BigInteger`	**`min(BigInteger val)`** Returns the minimum of this BigInteger and `val`.
`BigInteger`	**`mod(BigInteger m)`** Returns a BigInteger whose value is `(this mod m)`.

BigInteger	**modInverse**(**BigInteger** m) Returns a BigInteger whose value is $(this^{-1} \bmod m)$.
BigInteger	**modPow**(**BigInteger** exponent, **BigInteger** m) Returns a BigInteger whose value is $(this^{exponent} \bmod m)$.
BigInteger	**multiply**(**BigInteger** val) Returns a BigInteger whose value is (this * val).
BigInteger	**negate**() Returns a BigInteger whose value is (-this).
BigInteger	**nextProbablePrime**() Returns the first integer greater than this BigInteger that is probably prime.
BigInteger	**not**() Returns a BigInteger whose value is (~this).
BigInteger	**or**(**BigInteger** val) Returns a BigInteger whose value is (this | val).
BigInteger	**pow**(int exponent) Returns a BigInteger whose value is $(this^{exponent})$.
static **BigInteger**	**probablePrime**(int bitLength, **Random** rnd) Returns a positive BigInteger that is probably prime, with the specified bitLength.
BigInteger	**remainder**(**BigInteger** val) Returns a BigInteger whose value is (this % val).
BigInteger	**setBit**(int n) Returns a BigInteger whose value is equivalent to this BigInteger with the designated bit set.
BigInteger	**shiftLeft**(int n) Returns a BigInteger whose value is (this << n).
BigInteger	**shiftRight**(int n) Returns a BigInteger whose value is (this >> n).
short	**shortValueExact**() Converts this BigInteger to a short, checking for lost information.
int	**signum**() Returns the signum function of this BigInteger.
BigInteger	**subtract**(**BigInteger** val) Returns a BigInteger whose value is (this - val).
boolean	**testBit**(int n) Returns true if and only if the designated bit is set.
byte[]	**toByteArray**() Returns a byte array containing the two's-complement

	Returns a byte array containing the two's-complement representation of this BigInteger.
String	**toString**() Returns the decimal String representation of this BigInteger.
String	**toString**(int radix) Returns the String representation of this BigInteger in the given radix.
static **BigInteger**	**valueOf**(long val) Returns a BigInteger whose value is equal to that of the specified long.
BigInteger	**xor**(**BigInteger** val) Returns a BigInteger whose value is (this ^ val).

Methods inherited from class java.lang.Number

byteValue, shortValue

Methods inherited from class java.lang.Object

clone, finalize, getClass, notify, notifyAll, wait, wait, wait

Field Detail

ZERO

```
public static final BigInteger ZERO
```

The BigInteger constant zero.

Since:
1.2

ONE

```
public static final BigInteger ONE
```

The BigInteger constant one.

Since:
1.2

TEN

```
public static final BigInteger TEN
```

The BigInteger constant ten.