java.math

类 BigDecimal

java.lang.Object
  └ java.lang.Number
    └ java.math.BigDecimal

所有已实现的接口：

Serializable,Comparable<BigDecimal>

---

public class BigDecimal

extends Number

implements Comparable<BigDecimal>

不可变的、任意精度的有符号十进制数。BigDecimal 由任意精度的整数*非标度值* 和 32 位的整数*标度* (scale) 组成。如果为零或正数，则标度是小数点后的位数。如果为负数，则将该数的非标度值乘以 10 的负 scale 次幂。因此，BigDecimal 表示的数值是 $(unscaledValue \times 10^{-scale})$。

BigDecimal 类提供以下操作：算术、标度操作、舍入、比较、哈希算法和格式转换。toString() 方法提供 BigDecimal 的规范表示形式。

BigDecimal 类使用户能完全控制舍入行为。如果未指定舍入模式，并且无法表示准确结果，则抛出一个异常；否则，通过向该操作提供适当的 MathContext 对象，可以对已选择的精度和舍入模式执行计算。在任何情况下，可以为舍入控制提供八种*舍入模式*。使用此类（例如，ROUND_HALF_UP）中的整数字段来表示舍入模式已过时；应改为使用 RoundingModeenum（例如，RoundingMode.HALF_UP）的枚举值。

当为 MathContext 对象提供 0 的精度设置（例如，MathContext.UNLIMITED）时，算术运算是准确的，它们是不采用任何 MathContext 对象的算术方法。（这是第 5 版之前的版本支持的唯一行为。）为了计算准确结果，不使用附带 0 精度设置的 MathContext 对象的舍入模式设置，因此与该对象无关。在除法中，准确的商可能是一个无限长的十进制扩展；例如，1 除以 3 所得的商。如果商具有无穷的十进制扩展，但是指定了该操作返回准确结果，则抛出 ArithmeticException。否则，像其他操作那样，返回除法运算的准确结果。

当精度设置不为 0 时，BigDecimal算法的规则完全符合 ANSI X3.274-1996 和 ANSI X3.274-1996/AM 1-2000（ 7.4 节）中定义的算法的可选操作模式。与上述标准不同，BigDecimal 包括多种舍入模式，它们对于版本 5 以前的 BigDecimal 版本中的除法是强制性的。这些 ANSI 标准和 BigDecimal 规范之间的任何冲突都按照有利于 BigDecimal 的方式进行解决。

由于同一数值可以有不同的表示形式（具有不同的标度），因此运算和舍入的规则必须同时指定数值结果和结果表示形式中所用的标度。

一般情况下，当准确结果（在除法中，可能有无限多位）比返回的数值具有更多位数时，舍入模式和精度设置确定操作如何返回具有有限位数的结果。 首先，MathContext 的 precision 设置指定要返回

的总位数；这确定了结果的*精度*。位数计数从准确结果的最左边的非零数字开始。舍入模式确定丢弃的尾部位数如何影响返回的结果。

对于所有算术运算符，运算的执行方式是，首先计算准确的中间结果，然后，使用选择的舍入模式将其舍入为精度设置（如有必要）指定的位数。如果不返回准确结果，则将丢弃准确结果的某些数位。当舍入增加了返回结果的大小时，前导数字“9”的进位传播可能会创建新的数位。例如，将值 999.9 舍入为三位数字，则在数值上等于一千，表示为 100×10^{1}。在这种情况下，新的 "1" 是返回结果的前导数位。

除了逻辑的准确结果外，每种算术运算都有一个表示结果的首选标度。下表列出了每个运算的首选标度。

算术运算结果的首选标度

运算	结果的首选标度
加	max(addend.scale(), augend.scale())
减	max(minuend.scale(), subtrahend.scale())
乘	multiplier.scale() + multiplicand.scale()
除	dividend.scale() - divisor.scale()

这些标度是返回准确算术结果的方法使用的标度；准确相除可能必须使用较大的标度除外，因为准确的结果可能有较多的位数。例如， `1/32` 得到 `0.03125`。

舍入之前，逻辑的准确中间结果的标度是该运算的首选标度。如果用 `precision` 位数无法表示准确的数值结果，则舍入会选择要返回的一组数字，并将该结果的标度从中间结果的标度减小到可以表示实际返回的 `precision` 位数的最小标度。如果准确结果可以使用最多 `precision` 个数字表示，则返回具有最接近首选标度的标度的结果表示形式。尤其是，通过移除结尾零并减少标度，可以用少于 `precision` 个数字来表示准确的可表示的商。例如，使用 floor舍入模式将结果舍入为三个数字，

```
19/100 = 0.19 // integer=19, scale=2
```

但是

```
21/110 = 0.190 // integer=190, scale=3
```

注意，对于加、减和乘，标度的缩减量将等于丢弃的准确结果的数字位置数。如果舍入导致进位传播创建一个新的高位，则当未创建新的数位时，会丢弃该结果的附加数字。

其他方法可能与舍入语义稍微不同。例如，使用指定的算法的 `pow`方法得到的结果可能偶尔不同于舍入得到的算术结果，如最后一位有多个单位（*ulp*）。

可以通过两种类型的操作来处理 `BigDecimal` 的标度：标度/舍入操作和小数点移动操作。标度/舍入操作（`setScale` 和 `round`）返回 `BigDecimal`，其值近似地（或精确地）等于操作数的值，但是其标度或精度是指定的值；即：它们会增加或减少对其值具有最小影响的存储数的精度。小数点移动操作（`movePointLeft` 和 `movePointRight`）返回从操作数创建的 `BigDecimal`，创建的方法是按指定方向将小数点移动一个指定距离。

为了简洁明了起见，整个 `BigDecimal` 方法的描述中都使用了伪代码。伪代码表达式 `(i + j)`是“其值为 `BigDecimal i` 加 `BigDecimal j` 的 `BigDecimal`”的简写。伪代码表达式 `(i == j)`是“当且仅当 `BigDecimal i` 表示与 `BigDecimal j` 相同的值时，则为 `true`”的简写。可以类似地解释其他伪代码表达式。方括号用于表示特定的 `BigInteger` 和定义 `BigDecimal` 值的标度对；例如，[19, 2] 表示 `BigDecimal` 在数值上等于 0.19，标度是 2。

注：如果 BigDecimal 对象用作 SortedMap 中的键或 SortedSet 中的元素，则应特别小心，因为 BigDecimal 的*自然排序与 equals 方法不一致*。有关更多信息，请参见 Comparable、SortedMap 或 SortedSet。

当为任何输入参数传递 null 对象引用时，此类的所有方法和构造方法都将抛出 NullPointerException。

另请参见：

BigInteger, MathContext,RoundingMode, SortedMap,SortedSet, 序列化表格

字段摘要

static BigDecimal	ONE 值为 1，标度为 0。
static int	ROUND_CEILING 接近正无穷大的舍入模式。
static int	ROUND_DOWN 接近零的舍入模式。
static int	ROUND_FLOOR 接近负无穷大的舍入模式。
static int	ROUND_HALF_DOWN 向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则为上舍入的舍入模式。
static int	ROUND_HALF_EVEN 向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则向相邻的偶数舍入。
static int	ROUND_HALF_UP 向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则为向上舍入的舍入模式。
static int	ROUND_UNNECESSARY 断言请求的操作具有精确的结果，因此不需要舍入。
static int	ROUND_UP 舍入远离零的舍入模式。
static BigDecimal	TEN 值为 10，标度为 0。
static BigDecimal	ZERO 值为 0，标度为 0。

构造方法摘要

BigDecimal(BigInteger val) 将 BigInteger 转换为 BigDecimal。
BigDecimal(BigInteger unscaledVal, int scale) 将 BigInteger 非标度值和 int 标度转换为 BigDecimal。
BigDecimal(BigInteger unscaledVal, int scale, MathContext mc)

将 BigInteger 非标度值和 int 标度转换为 BigDecimal（根据上下文设置进行舍入）。
BigDecimal(BigInteger val, MathContext mc) 将 BigInteger 转换为 BigDecimal（根据上下文设置进行舍入）。
BigDecimal(char[] in) 将 BigDecimal 的字符数组表示形式转换为 BigDecimal，接受与 BigDecimal(String) 构造方法相同的字符序列。
BigDecimal(char[] in, int offset, int len) 将 BigDecimal 的字符数组表示形式转换为 BigDecimal，接受与 BigDecimal(String) 构造方法相同的字符序列，同时允许指定子数组。
BigDecimal(char[] in, int offset, int len, MathContext mc) 将 BigDecimal 的字符数组表示形式转换为 BigDecimal，接受与 BigDecimal(String) 构造方法相同的字符序列，同时允许指定子数组，并根据上下文设置进行舍入。
BigDecimal(char[] in, MathContext mc) 将 BigDecimal 的字符数组表示形式转换为 BigDecimal，接受与 BigDecimal(String) 构造方法相同的字符序列（根据上下文设置进行舍入）。
BigDecimal(double val) 将 double 转换为 BigDecimal，后者是 double 的二进制浮点值准确的十进制表示形式。
BigDecimal(double val, MathContext mc) 将 double 转换为 BigDecimal（根据上下文设置进行舍入）。
BigDecimal(int val) 将 int 转换为 BigDecimal。
BigDecimal(int val, MathContext mc) 将 int 转换为 BigDecimal（根据上下文设置进行舍入）。
BigDecimal(long val) 将 long 转换为 BigDecimal。
BigDecimal(long val, MathContext mc) 将 long 转换为 BigDecimal（根据上下文设置进行舍入）。
BigDecimal(String val) 将 BigDecimal 的字符串表示形式转换为 BigDecimal。
BigDecimal(String val, MathContext mc) 将 BigDecimal 的字符串表示形式转换为 BigDecimal，接受与 BigDecimal(String) 构造方法相同的字符串（按照上下文设置进行舍入）。

方法摘要

BigDecimal	abs() 返回 BigDecimal，其值为此 BigDecimal 的绝对值，其标度为 this.scale()。
BigDecimal	abs(MathContext mc) 返回其值为此 BigDecimal 绝对值的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	add(BigDecimal augend) 返回一个 BigDecimal，其值为 (this + augend)，其标度为 max(this.scale(), augend.scale())。

BigDecimal	add(BigDecimal augend, MathContext mc) 返回其值为 (this + augend) 的 BigDecimal（根据上下文设置进行舍入）。
byte	byteValueExact() 将此 BigDecimal 转换为 byte，以检查丢失的信息。
int	compareTo(BigDecimal val) 将此 BigDecimal 与指定的 BigDecimal 比较。
BigDecimal	divide(BigDecimal divisor) 返回一个 BigDecimal，其值为 (this / divisor)，其首选标度为 (this.scale() - divisor.scale())；如果无法表示准确的商值（因为它有无穷的十进制扩展），则抛出 ArithmeticException。
BigDecimal	divide(BigDecimal divisor, int roundingMode) 返回一个 BigDecimal，其值为 (this / divisor)，其标度为 this.scale()。
BigDecimal	divide(BigDecimal divisor, int scale, int roundingMode) 返回一个 BigDecimal，其值为 (this / divisor)，其标度为指定标度。
BigDecimal	divide(BigDecimal divisor, int scale, RoundingMode roundingMode) 返回一个 BigDecimal，其值为 (this / divisor)，其标度为指定标度。
BigDecimal	divide(BigDecimal divisor, MathContext mc) 返回其值为 (this / divisor) 的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	divide(BigDecimal divisor, RoundingMode roundingMode) 返回一个 BigDecimal，其值为 (this / divisor)，其标度为 this.scale()。
BigDecimal[]	divideAndRemainder(BigDecimal divisor) 返回由两个元素组成的 BigDecimal数组，该数组包含 divideToIntegralValue的结果，后跟对两个操作数计算所得到的 remainder。
BigDecimal[]	divideAndRemainder(BigDecimal divisor, MathContext mc) 返回由两个元素组成的 BigDecimal数组，该数组包含 divideToIntegralValue的结果，后跟根据上下文设置对两个操作数进行舍入计算所得到的 remainder 的结果。
BigDecimal	divideToIntegralValue(BigDecimal divisor) 返回 BigDecimal，其值为向下舍入所得商值 (this / divisor) 的整数部分。
BigDecimal	divideToIntegralValue(BigDecimal divisor, MathContext mc) 返回 BigDecimal，其值为 (this / divisor) 的整数部分。
double	doubleValue() 将此 BigDecimal 转换为 double。
boolean	equals(Object x) 比较此 BigDecimal 与指定的 Object的相等性。
float	floatValue() 将此 BigDecimal 转换为 float。
int	hashCode() 返回此 BigDecimal 的哈希码。
int	intValue() 将此 BigDecimal 转换为 int。
int	intValueExact() 将此 BigDecimal 转换为 int，以检查丢失的信息。

long	longValue() 将此 BigDecimal 转换为 long。
long	longValueExact() 将此 BigDecimal 转换为 long，以检查丢失的信息。
BigDecimal	max(BigDecimal val) 返回此 BigDecimal 和 val 的最大值。
BigDecimal	min(BigDecimal val) 返回此 BigDecimal 和 val 的最小值。
BigDecimal	movePointLeft(int n) 返回一个 BigDecimal，它等效于将该值的小数点向左移动 n 位。
BigDecimal	movePointRight(int n) 返回一个 BigDecimal，它等效于将该值的小数点向右移动 n 位。
BigDecimal	multiply(BigDecimal multiplicand) 返回一个 BigDecimal，其值为 (this × multiplicand)，其标度为 (this.scale() + multiplicand.scale())。
BigDecimal	multiply(BigDecimal multiplicand, MathContext mc) 返回其值为 (this × multiplicand)的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	negate() 返回 BigDecimal，其值为 (-this)，其标度为 this.scale()。
BigDecimal	negate(MathContext mc) 返回其值为 (-this) 的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	plus() 返回 BigDecimal，其值为 (+this)，其标度为 this.scale()。
BigDecimal	plus(MathContext mc) 返回其值为 (+this) 的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	pow(int n) 返回其值为 ($this^n$) 的 BigDecimal，准确计算该幂，使其具有无限精度。
BigDecimal	pow(int n, MathContext mc) 返回其值为 ($this^n$) 的 BigDecimal。
int	precision() 返回此 BigDecimal 的*精度*。
BigDecimal	remainder(BigDecimal divisor) 返回其值为 (this % divisor) 的 BigDecimal。
BigDecimal	remainder(BigDecimal divisor, MathContext mc) 返回其值为 (this % divisor) 的 BigDecimal（根据上下文设置进行舍入）。
BigDecimal	round(MathContext mc) 返回根据 MathContext 设置进行舍入后的 BigDecimal。
int	scale() 返回此 BigDecimal 的*标度*。
BigDecimal	scaleByPowerOfTen(int n) 返回其数值等于 ($this * 10^n$) 的 BigDecimal。

BigDecimal	setScale(int newScale) 返回一个 BigDecimal，其标度为指定值，其值在数值上等于此 BigDecimal的值。
BigDecimal	setScale(int newScale, int roundingMode) 返回一个 BigDecimal，其标度为指定值，其非标度值通过此 BigDecimal 的非标度值乘以或除以十的适当次幂来确定，以维护其总值。
BigDecimal	setScale(int newScale, RoundingMode roundingMode) 返回 BigDecimal，其标度为指定值，其非标度值通过此 BigDecimal 的非标度值乘以或除以十的适当次幂来确定，以维护其总值。
short	shortValueExact() 将此 BigDecimal 转换为 short，以检查丢失的信息。
int	signum() 返回此 BigDecimal 的正负号函数。
BigDecimal	stripTrailingZeros() 返回数值上等于此小数，但从该表示形式移除所有尾部零的 BigDecimal。
BigDecimal	subtract(BigDecimal subtrahend) 返回一个 BigDecimal，其值为 (this - subtrahend)，其标度为 max(this.scale(), subtrahend.scale())。
BigDecimal	subtract(BigDecimal subtrahend, MathContext mc) 返回其值为 (this - subtrahend) 的 BigDecimal（根据上下文设置进行舍入）。
BigInteger	toBigInteger() 将此 BigDecimal 转换为 BigInteger。
BigInteger	toBigIntegerExact() 将此 BigDecimal 转换为 BigInteger，以检查丢失的信息。
String	toEngineeringString() 返回此 BigDecimal 的字符串表示形式，需要指数时，则使用工程计数法。
String	toPlainString() 返回不带指数字段的此 BigDecimal的字符串表示形式。
String	toString() 返回此 BigDecimal 的字符串表示形式，如果需要指数，则使用科学记数法。
BigDecimal	ulp() 返回此 BigDecimal 的 ulp（最后一位的单位）的大小。
BigInteger	unscaledValue() 返回其值为此 BigDecimal 的*非标度值* 的 BigInteger。
static BigDecimal	valueOf(double val) 使用 Double.toString(double) 方法提供的 double 规范的字符串表示形式将 double 转换为 BigDecimal。
static BigDecimal	valueOf(long val) 将 long 值转换为具有零标度的 BigDecimal。
static BigDecimal	valueOf(long unscaledVal, int scale) 将 long 非标度值和 int 标度转换为 BigDecimal。

从类 java.lang.Number 继承的方法

byteValue, shortValue

从类 java.lang.Object 继承的方法
clone, finalize, getClass, notify, notifyAll, wait, wait, wait