Class java.lang.Math

```
java.lang.Object
   |
   +----java.lang.Math
```

---

public final class **Math**
extends Object

The standard Math library. For the methods in this Class, error handling for out−of−range or immeasurable results are platform dependent. This class cannot be subclassed or instantiated because all methods and variables are static.

Version:
1.18, 10/05/95

Variable Index

o **E**
The float representation of the value E.
o **PI**
The float representation of the value Pi.

Method Index

o **IEEEremainder**(double, double)
Returns the remainder of f1 divided by f2 as defined by IEEE 754.
o **abs**(int)
Returns the absolute integer value of a.
o **abs**(long)
Returns the absolute long value of a.
o **abs**(float)
Returns the absolute float value of a.
o **abs**(double)
Returns the absolute double value of a.
o **acos**(double)
Returns the arc cosine of a, in the range of 0.0 through Pi.

- **asin**(double)
  Returns the arc sine of a, in the range of −Pi/2 through Pi/2.
- **atan**(double)
  Returns the arc tangent of a, in the range of −Pi/2 through Pi/2.
- **atan2**(double, double)
  Converts rectangular coordinates (a, b) to polar (r, theta).
- **ceil**(double)
  Returns the "ceiling" or smallest whole number greater than or equal to a.
- **cos**(double)
  Returns the trigonometric cosine of an angle.
- **exp**(double)
  Returns the exponential number e(2.718...) raised to the power of a.
- **floor**(double)
  Returns the "floor" or largest whole number less than or equal to a.
- **log**(double)
  Returns the natural logarithm (base e) of a.
- **max**(int, int)
  Takes two int values, a and b, and returns the greater number of the two.
- **max**(long, long)
  Takes two long values, a and b, and returns the greater number of the two.
- **max**(float, float)
  Takes two float values, a and b, and returns the greater number of the two.
- **max**(double, double)
  Takes two double values, a and b, and returns the greater number of the two.
- **min**(int, int)
  Takes two integer values, a and b, and returns the smallest number of the two.
- **min**(long, long)
  Takes two long values, a and b, and returns the smallest number of the two.
- **min**(float, float)
  Takes two float values, a and b, and returns the smallest number of the two.
- **min**(double, double)
  Takes two double values, a and b, and returns the smallest number of the two.
- **pow**(double, double)
  Returns the number a raised to the power of b.
- **random**()
  Generates a random number between 0.0 and 1.0.

  Random number generators are often referred to as pseudorandom number generators because the numbers produced tend to repeat themselves after a period of time.
- **rint**(double)
  Converts a double value into an integral value in double format.
- **round**(float)
  Rounds off a float value by first adding 0.5 to it and then returning the largest integer that is less than or equal to this new value.
- **round**(double)
  Rounds off a double value by first adding 0.5 to it and then returning the largest integer that is less than or equal to this new value.
- **sin**(double)
  Returns the trigonometric sine of an angle.

o **sqrt**(double)
Returns the square root of a.
o **tan**(double)
Returns the trigonometric tangent of an angle.

Variables

o **E**

```
public final static double E
```

The float representation of the value E. E is equivalent to 2.718281828459045235 4f in Java.

o **PI**

```
public final static double PI
```

The float representation of the value Pi. Pi is equivalent to 3.14159265358979323846f in Java.

Methods

o **sin**

```
public static double sin(double a)
```

Returns the trigonometric sine of an angle.
Parameters:
a – an assigned angle that is measured in radians

o **cos**

```
public static double cos(double a)
```

Returns the trigonometric cosine of an angle.
Parameters:
a – an assigned angle that is measured in radians

o **tan**

```
public static double tan(double a)
```

Returns the trigonometric tangent of an angle.
Parameters:
a – an assigned angle that is measured in radians

o **asin**

```
public static double asin(double a)
```

Returns the arc sine of a, in the range of –Pi/2 through Pi/2.

Parameters:

a – (–1.0) = a = 1.0

o **acos**

```
public static double acos(double a)
```

Returns the arc cosine of a, in the range of 0.0 through Pi.

Parameters:

a – (–1.0) = a = 1.0

o **atan**

```
public static double atan(double a)
```

Returns the arc tangent of a, in the range of –Pi/2 through Pi/2.

Parameters:

a – an assigned value

Returns:

the arc tangent of a.

o **exp**

```
public static double exp(double a)
```

Returns the exponential number e(2.718...) raised to the power of a.

Parameters:

a – an assigned value

o **log**

```
public static double log(double a) throws ArithmeticException
```

Returns the natural logarithm (base e) of a.

Parameters:

a – a is a number greater than 0.0

Throws: ArithmeticException

If a is less than 0.0 .

o **sqrt**

```
public static double sqrt(double a) throws ArithmeticException
```

Returns the square root of a.

Parameters:

a – a is a number greater than or equal to 0.0

Throws: ArithmeticException
If a is a value less than 0.0 .

o **IEEEremainder**

```
public static double IEEEremainder(double f1,
                                   double f2)
```

Returns the remainder of f1 divided by f2 as defined by IEEE 754.
Parameters:
f1 – the dividend
f2 – the divisor

o **ceil**

```
public static double ceil(double a)
```

Returns the "ceiling" or smallest whole number greater than or equal to a.
Parameters:
a – an assigned value

o **floor**

```
public static double floor(double a)
```

Returns the "floor" or largest whole number less than or equal to a.
Parameters:
a – an assigned value

o **rint**

```
public static double rint(double a)
```

Converts a double value into an integral value in double format.
Parameters:
a – an assigned double value

o **atan2**

```
public static double atan2(double a,
                           double b)
```

Converts rectangular coordinates (a, b) to polar (r, theta). This method computes the phase theta by computing an arc tangent of b/a in the range of –Pi to Pi.
Parameters:
a – an assigned value
b – an assigned value
Returns:
the polar coordinates (r, theta).

o **pow**

```
public static double pow(double a,
                         double b) throws ArithmeticException
```

Returns the number a raised to the power of b. If (a == 0.0), then b must be greater than 0.0; otherwise you will throw an exception. An exception will also occur if (a = 0.0) and b is not equal to a whole number.

Parameters:

a – an assigned value with the exceptions: (a == 0.0) –> (b > 0.0) && (a (b == a whole number)

b – an assigned value with the exceptions: (a == 0.0) –> (b > 0.0) && (a (b == a whole number)

Throws: ArithmeticException

If (a == 0.0) and (b = 0.0) .

Throws: ArithmeticException

If (a = 0.0) and b is not equal to a whole number.

o **round**

```
public static int round(float a)
```

Rounds off a float value by first adding 0.5 to it and then returning the largest integer that is less than or equal to this new value.

Parameters:

a – the value to be rounded off

o **round**

```
public static long round(double a)
```

Rounds off a double value by first adding 0.5 to it and then returning the largest integer that is less than or equal to this new value.

Parameters:

a – the value to be rounded off

o **random**

```
public static synchronized double random()
```

Generates a random number between 0.0 and 1.0.

Random number generators are often referred to as pseudorandom number generators because the numbers produced tend to repeat themselves after a period of time.

Returns:

a pseudorandom double between 0.0 and 1.0.

o **abs**

```
public static int abs(int a)
```

Returns the absolute integer value of a.
Parameters:
a – an assigned integer value

o **abs**

```
public static long abs(long a)
```

Returns the absolute long value of a.
Parameters:
a – an assigned long value.

o **abs**

```
public static float abs(float a)
```

Returns the absolute float value of a.
Parameters:
a – an assigned float value

o **abs**

```
public static double abs(double a)
```

Returns the absolute double value of a.
Parameters:
a – an assigned double value

o **max**

```
public static int max(int a,
                      int b)
```

Takes two int values, a and b, and returns the greater number of the two.
Parameters:
a – an integer value to be compared
b – an integer value to be compared

o **max**

```
public static long max(long a,
                       long b)
```

Takes two long values, a and b, and returns the greater number of the two.
Parameters:
a – a long value to be compared
b – a long value to be compared

o **max**

```
public static float max(float a,
                        float b)
```

Takes two float values, a and b, and returns the greater number of the two.

Parameters:

a – a float value to be compared

b – a float value to be compared

o **max**

```
public static double max(double a,
                         double b)
```

Takes two double values, a and b, and returns the greater number of the two.

Parameters:

a – a double value to be compared

b – a double value to be compared

o **min**

```
public static int min(int a,
                      int b)
```

Takes two integer values, a and b, and returns the smallest number of the two.

Parameters:

a – an integer value to be compared

b – an integer value to be compared

o **min**

```
public static long min(long a,
                       long b)
```

Takes two long values, a and b, and returns the smallest number of the two.

Parameters:

a – a long value to be compared

b – a long value to be compared

o **min**

```
public static float min(float a,
                        float b)
```

Takes two float values, a and b, and returns the smallest number of the two.

Parameters:

a – a float value to be compared

b – a float value to be compared

o **min**

```
public static double min(double a,
                         double b)
```

Takes two double values, a and b, and returns the smallest number of the two.

Parameters:

a – a double value to be compared

b – a double value to be compared