JAVA.LANG.MATH.POW METHOD

Description

The **java.lang.Math.pow**$doublea, doubleb$ returnsthe value of the first argument raised to the power of the second argument. Special cases:

- If the second argument is positive or negative zero, then the result is 1.0.
- If the second argument is 1.0, then the result is the same as the first argument.
- If the second argument is NaN, then the result is NaN.
- If the first argument is NaN and the second argument is nonzero, then the result is NaN.
- If
  - the absolute value of the first argument is greater than 1 and the second argument is positive infinity, or
  - the absolute value of the first argument is less than 1 and the second argument is negative infinity,

  then the result is positive infinity.
- If
  - the absolute value of the first argument is greater than 1 and the second argument is negative infinity, or
  - the absolute value of the first argument is less than 1 and the second argument is positive infinity,

  then the result is positive zero.
- If the absolute value of the first argument equals 1 and the second argument is infinite, then the result is NaN.
- If
  - the first argument is positive zero and the second argument is greater than zero, or
  - the first argument is positive infinity and the second argument is less than zero,

  then the result is positive zero.
- If
  - the first argument is positive zero and the second argument is less than zero, or
  - the first argument is positive infinity and the second argument is greater than zero,

  then the result is positive infinity.
- If
  - the first argument is negative zero and the second argument is greater than zero but not a finite odd integer, or
  - the first argument is negative infinity and the second argument is less than zero but not a finite odd integer,

  then the result is positive zero.
- If
  - the first argument is negative zero and the second argument is a positive finite odd integer, or
  - the first argument is negative infinity and the second argument is a negative finite odd integer,

  then the result is negative zero.
- If
  - the first argument is negative zero and the second argument is less than zero but not a finite odd integer, or
  - the first argument is negative infinity and the second argument is greater than zero but not a finite odd integer,

  then the result is positive infinity.
- If
  - the first argument is negative zero and the second argument is a negative finite odd integer, or
  - the first argument is negative infinity and the second argument is a positive finite odd integer,

  then the result is negative infinity.
- If the first argument is finite and less than zero
  - if the second argument is a finite even integer, the result is equal to the result of raising the absolute value of the first argument to the power of the second argument
  - if the second argument is a finite odd integer, the result is equal to the negative of the result of raising the absolute value of the first argument to the power of the second argument
  - if the second argument is finite and not an integer, then the result is NaN.
- If both arguments are integers, then the result is exactly equal to the mathematical result of raising the first argument to the power of the second argument if that result can in fact be represented exactly as a `double` value.

$Intheforegoingdescriptions, afloating - pointvalueisconsideredtobeanintegerifandonlyifitisfiniteandafixedpointofthemethodceilor, equivalently, afixedpointofthemethodfloor. Avalueisafixedpointofaone - argumentmethodifandonlyiftheresultofapplyingthemethodtothevalueisequaltothevalue.$

The computed result must be within 1 ulp of the exact result. Results must be semi-monotonic.

Declaration

Following is the declaration for **java.lang.Math.pow** method

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

Parameters

- **a** -- the base.
- **b** -- the exponent.

Return Value

This method returns the value a^b.

Exception

- **NA**

Example

The following example shows the usage of lang.Math.pow method.

```
package com.tutorialspoint;

import java.lang.*;

public class MathDemo {

   public static void main(String[] args) {

   // get two double numbers
   double x = 2.0;
   double y = 5.4;

   // print x raised by y and then y raised by x
```

```
    System.out.println("Math.pow(" + x + "," + y + ")=" + Math.pow(x, y));
    System.out.println("Math.pow(" + y + "," + x + ")=" + Math.pow(y, x));
    }
}
```

Let us compile and run the above program, this will produce the following result:

```
Math.pow(2.0, 5.4)=42.22425314473263
Math.pow(5.4, 2.0)=29.160000000000004
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js