**Python Mathematical Functions ( part 2)**

|  |  |
| --- | --- |
| 8 | log10(x)  The base-10 logarithm of x for x> 0. Description The method **log10()** returns base-10 logarithm of x for x > 0. Syntax Following is the syntax for **log10()** method −  import math  math.log10( x )  **Note** − This function is not accessible directly, so we need to import math module and then we need to call this function using math static object. Parameters  * **x** − This is a numeric expression.  Return Value This method returns base-10 logarithm of x for x > 0. Example The following example shows the usage of log10() method.  #!/usr/bin/python  import math # This will import **math** module  print "math.log10(100.12) : ", math.log10(100.12)  print "math.log10(100.72) : ", math.log10(100.72)  print "math.log10(119L) : ", math.log10(119L)  print "math.log10(math.pi) : ", math.log10(math.pi)  When we run above program, it produces following result −  math.log10(100.12) : 2.00052084094  math.log10(100.72) : 2.0031157171  math.log10(119L) : 2.07554696139  math.log10(math.pi) : 0.497149872694 |
| 9 | Max (x1, x2.......)  The largest of its arguments: the value closest to positive infinity. Description The method **max()** returns the largest of its arguments: the value closest to positive infinity. Syntax Following is the syntax for **max()** method −  max( x, y, z, .... ) Parameters  * **x** − This is a numeric expression. * **y** − This is also a numeric expression. * **z** − This is also a numeric expression.  Return Value This method returns largest of its arguments. Example The following example shows the usage of max() method.  #!/usr/bin/python  print "max(80, 100, 1000) : ", max(80, 100, 1000)  print "max(-20, 100, 400) : ", max(-20, 100, 400)  print "max(-80, -20, -10) : ", max(-80, -20, -10)  print "max(0, 100, -400) : ", max(0, 100, -400)  When we run above program, it produces following result −  max(80, 100, 1000) : 1000  max(-20, 100, 400) : 400  max(-80, -20, -10) : -10  max(0, 100, -400) : 100 |
| 10 | Min( x1, x2.......)  The smallest of its arguments: the value closest to negative infinity. Description The method **min()** returns the smallest of its arguments: the value closest to negative infinity. Syntax Following is the syntax for **min()** method −  min( x, y, z, .... ) Parameters  * **x** − This is a numeric expression. * **y** − This is also a numeric expression. * **z** − This is also a numeric expression.  Return Value This method returns smallest of its arguments. Example The following example shows the usage of min() method.  #!/usr/bin/python  print "min(80, 100, 1000) : ", min(80, 100, 1000)  print "min(-20, 100, 400) : ", min(-20, 100, 400)  print "min(-80, -20, -10) : ", min(-80, -20, -10)  print "min(0, 100, -400) : ", min(0, 100, -400)  When we run above program, it produces following result −  min(80, 100, 1000) : 80  min(-20, 100, 400) : -20  min(-80, -20, -10) : -80  min(0, 100, -400) : -400 |
| 11 | modf(x)  The fractional and integer parts of x in a two-item tuple. Both parts have the same sign as x. The integer part is returned as a float. Description The method **modf()** returns the fractional and integer parts of x in a two-item tuple. Both parts have the same sign as x. The integer part is returned as a float. Syntax Following is the syntax for **modf()** method −  import math  math.modf( x )  **Note** − This function is not accessible directly, so we need to import math module and then we need to call this function using math static object. Parameters  * **x** − This is a numeric expression.  Return Value This method returns the fractional and integer parts of x in a two-item tuple. Both parts have the same sign as x. The integer part is returned as a float. Example The following example shows the usage of modf() method.  #!/usr/bin/python  import math # This will import math module  print "math.modf(100.12) : ", math.modf(100.12)  print "math.modf(100.72) : ", math.modf(100.72)  print "math.modf(119L) : ", math.modf(119L)  print "math.modf(math.pi) : ", math.modf(math.pi)  When we run above program, it produces following result −  math.modf(100.12) : (0.12000000000000455, 100.0)  math.modf(100.72) : (0.71999999999999886, 100.0)  math.modf(119L) : (0.0, 119.0)  math.modf(math.pi) : (0.14159265358979312, 3.0) |
| 12 | pow(x,y)  The value of x\*\*y. Return Value This method returns value of xy. Example The following example shows the usage of pow() method.  #!/usr/bin/python  import math # This will import math module  print "math.pow(100, 2) : ", math.pow(100, 2)  print "math.pow(100, -2) : ", math.pow(100, -2)  print "math.pow(2, 4) : ", math.pow(2, 4)  print "math.pow(3, 0) : ", math.pow(3, 0)  When we run above program, it produces following result −  math.pow(100, 2) : 10000.0  math.pow(100, -2) : 0.0001  math.pow(2, 4) : 16.0  math.pow(3, 0) : 1.0 |
| 13 | round(x[n] :  **x** rounded to n digits from the decimal point. Python rounds away from zero as a tie-breaker: round(0.5) is 1.0 and round(-0.5) is -1.0. Description The method **round()** returns x rounded to n digits from the decimal point. Syntax Following is the syntax for **round()** method −  round( x [, n] ) Parameters  * **x** − This is a numeric expression. * **n** − This is also a numeric expression.  Return Value This method returns x rounded to n digits from the decimal point. Example The following example shows the usage of round() method.  #!/usr/bin/python  print "round(80.23456, 2) : ", round(80.23456, 2)  print "round(100.000056, 3) : ", round(100.000056, 3)  print "round(-100.000056, 3) : ", round(-100.000056, 3)  When we run above program, it produces following result −  round(80.23456, 2) : 80.23  round(100.000056, 3) : 100.0  round(-100.000056, 3) : -100.0 |
| 14 | sqrt(x) :  The square root of x for x > 0 Description The method **sqrt()** returns the square root of x for x > 0. Syntax Following is the syntax for **sqrt()** method −  import math  math.sqrt( x )  **Note** − This function is not accessible directly, so we need to import math module and then we need to call this function using math static object. Parameters  * **x** − This is a numeric expression.  Return Value This method returns square root of x for x > 0. Example The following example shows the usage of sqrt() method.  #!/usr/bin/python  import math # This will import **math** module  print "math.sqrt(100) : ", math.sqrt(100)  print "math.sqrt(7) : ", math.sqrt(7)  print "math.sqrt(math.pi) : ", math.sqrt(math.pi)  When we run above program, it produces following result −  math.sqrt(100) : 10.0  math.sqrt(7) : 2.64575131106  math.sqrt(math.pi) : 1.77245385091 |