= 1 at x =
0 unless 00 = 1 .

= = = Continuous exponents = = =

Limits involving algebraic operations can often be evaluated by replacing subexpressions by their limits ; if the resulting expression does not determine the original limit , the expression is known as an indeterminate form . In fact , when f (t) and g (t) are real @-@ valued functions both approaching 0 (as t approaches a real number or ± ?) , with f (t) > 0 , the function f (t) g (t) need not approach 1 ; depending on f and g , the limit of f (t) g (t) can be any nonnegative real number or + ? , or it can diverge . For example , the functions below are of the form f (t) g (t) with f (t) , g (t) ? 0 as t ? 0 + , but the limits are different :
<formula> .
Thus , the two @-@ variable function xy , though continuous on the set { (x , y) : x > 0 } , cannot be extended to a continuous function on any set containing (0 , 0) , no matter how one chooses to define 00 . However , under certain conditions , such as when f and g are both analytic functions and f is positive on the open interval (0 , b) for some positive b , the limit approaching from the right is always 1 .

= = = Complex exponents = = =

In the complex domain , the function zw may be defined for nonzero z by choosing a branch of log z and defining zw as ew log z . This does not define 0w since there is no branch of log z defined at z = 0 , let alone in a neighborhood of 0 .

= = = History of differing points of view = = =

The debate over the definition of <formula> has been going on at least since the early 19th century . At that time , most mathematicians agreed that <formula> , until in 1821 Cauchy listed <formula> along with expressions like <formula> in a table of indeterminate forms . In the 1830s Libri published an unconvincing argument for <formula> , and Möbius sided with him , erroneously claiming that <formula> whenever <formula> . A commentator who signed his name simply as " S " provided the counterexample of <formula> , and this quieted the debate for some time . More historical details can be found in Knuth (1992) .
More recent authors interpret the situation above in different ways :
Some argue that the best value for <formula> depends on context , and hence that defining it once and for all is problematic . According to Benson (1999) , " The choice whether to define <formula> is based on convenience , not on correctness . If we refrain from defining <formula> then certain assertions become unnecessarily awkward . The consensus is to use the definition <formula> , although there are textbooks that refrain from defining <formula> . "
Others argue that <formula> should be defined as 1 . Knuth (1992) contends strongly that <formula> " has to be 1 " , drawing a distinction between the value <formula> , which should equal 1 as advocated by Libri , and the limiting form <formula> (an abbreviation for a limit of <formula> where <formula>) , which is necessarily an indeterminate form as listed by Cauchy : " Both Cauchy and Libri were right , but Libri and his defenders did not understand why truth was on their side . "

= = = Treatment on computers = = =

= = = = IEEE floating point standard = = = =

The IEEE 754 @-@ 2008 floating point standard is used in the design of most floating point libraries . It recommends a number of functions for computing a power :

pow treats 00 as 1 . This is the oldest defined version . If the power is an exact integer the result is the same as for pown , otherwise the result is as for powr (except for some exceptional cases) .
pown treats 00 as 1 . The power must be an exact integer . The value is defined for negative bases ; e.g. , pown (? 3 @,@ 5) is ? 243 .
powr treats 00 as NaN (Not @-@ a @-@ Number ? undefined) . The value is also NaN for cases like powr (? 3 @,@ 2) where the base is less than zero . The value is defined by epower × log (base) .

= = = = Programming languages = = = =

Most programming language with a power function are implemented using the IEEE pow function and therefore evaluate 00 as 1 . The later C and C + + standards describe this as the normative behaviour . The Java standard mandates this behavior . The .NET Framework method System.Math.Pow also treats 00 as 1 .

= = = = Mathematics software = = = =

Sage simplifies b0 to 1 , even if no constraints are placed on b . It takes 00 to be 1 , but does not simplify 0x for other x .
Maple distinguishes between integers 0 , 1 , ... and the corresponding floats 0 @.@ 0 , 1 @.@ 0 , ... (usually denoted 0 . , 1 . , ...) . If x does not evaluates to a number , then x0 and x0.0 are respectively evaluated to 1 (integer) and 1 @.@ 0 (float) ; on the other hand , 0x is evaluated to the integer 0 , while 0.0x is evaluated as 0.x. If both the base and the exponent are zero (or are evaluated to zero) , the result is Float (undefined) if the exponent is the float 0 @.@ 0 ; with an integer as exponent , the evaluation of 00 results in the integer 1 , while that of 0 @.@ 0 results in the float 1 @.@ 0 .
Macsyma also simplifies b0 to 1 even if no constraints are placed on b , but issues an error for 00 . For x > 0 , it simplifies 0x to 0 .
Mathematica and Wolfram Alpha simplify b0 into 1 , even if no constraints are placed on b . While Mathematica does not simplify 0x , Wolfram Alpha returns two results , 0 for x > 0 , and " indeterminate " for real x . Both Mathematica and Wolfram Alpha take 00 to be " (indeterminate) " .
Matlab , Python , Magma , GAP , singular , PARI / GP and the Google and iPhone calculators evaluate 00 as 1 .

= = Limits of powers = =

The section § Zero to the power of zero gives a number of examples of limits that are of the indeterminate form 00 . The limits in these examples exist , but have different values , showing that the two @-@ variable function xy has no limit at the point (0 , 0) . One may consider at what points this function does have a limit .