Search (S)

Overview
Tutorial
Reference
LANGUAGE
Syntax
Styling
Scripting
Context
LIBRARY
Foundations
Model
Text
Math
Accent
Attach
Binomial
Cancel
Cases
Class
Equation
Fraction
Left/Right
Matrix
Primes
Roots

> Reference > Math

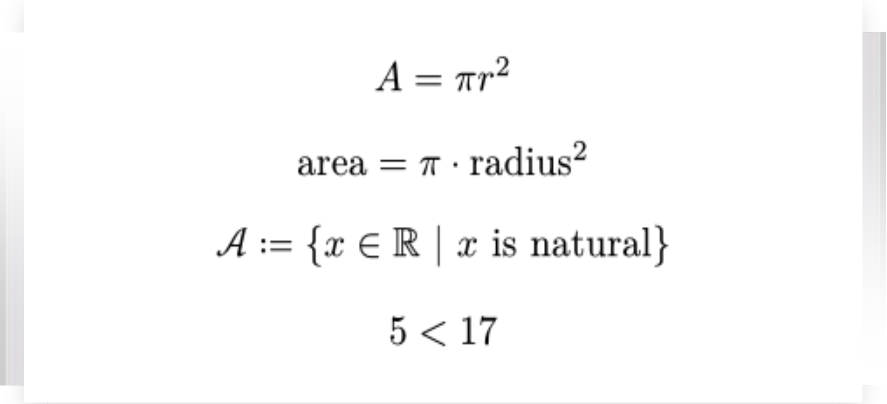
# Math

Typst has special syntax and library functions to typeset mathematical formulas. Math formulas can be displayed inline with text or as separate blocks. They will be typeset into their own block if they start and end with at least one space (e.g. `$ x^2 $`).

## Variables

In math, single letters are always displayed as is. Multiple letters, however, are interpreted as variables and functions. To display multiple letters verbatim, you can place them into quotes and to access single letter variables, you can use the hash syntax.

```
$ A = pi r^2 $
$ "area" = pi dot "radius"^2 $
$ cal(A) :=
    { x in RR | x "is natural" } $
#let x = 5
$ #x < 17 $
```



$$A = \pi r^2$$

$$\text{area} = \pi \cdot \text{radius}^2$$

$$\mathcal{A} := \{x \in \mathbb{R} \mid x \text{ is natural}\}$$

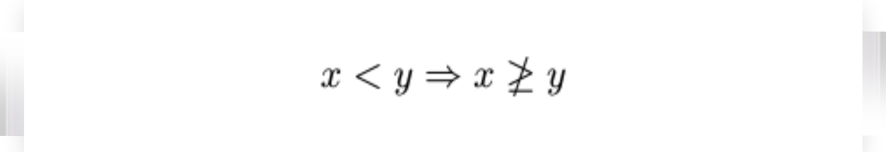
$$5 < 17$$

Sizes
Stretch
Styles
Text Operator
Under/Over
Variants
Vector
Symbols
Layout
Visualize
Introspection
Data Loading
Guides
Changelog
Roadmap
Community

## Symbols

Math mode makes a wide selection of symbols like `pi`, `dot`, or `RR` available. Many mathematical symbols are available in different variants. You can select between different variants by applying modifiers to the symbol. Typst further recognizes a number of shorthand sequences like => that approximate a symbol. When such a shorthand exists, the symbol's documentation lists it.

```
$ x < y => x gt.eq.not y $
```

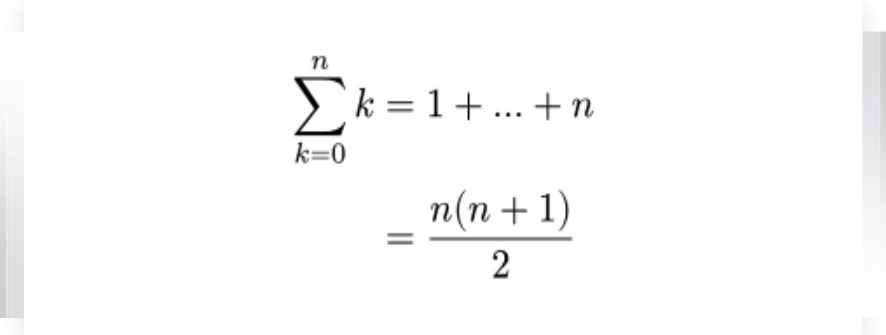


$$x < y \Rightarrow x \ngeq y$$

## Line Breaks

Formulas can also contain line breaks. Each line can contain one or multiple *alignment points* (&) which are then aligned.

```
$ sum_(k=0)^n k
    &= 1 + ... + n \
    &= (n(n+1)) / 2 $
```



$$\begin{aligned}\sum_{k=0}^{n} k &= 1 + \ldots + n \\ &= \frac{n(n+1)}{2}\end{aligned}$$

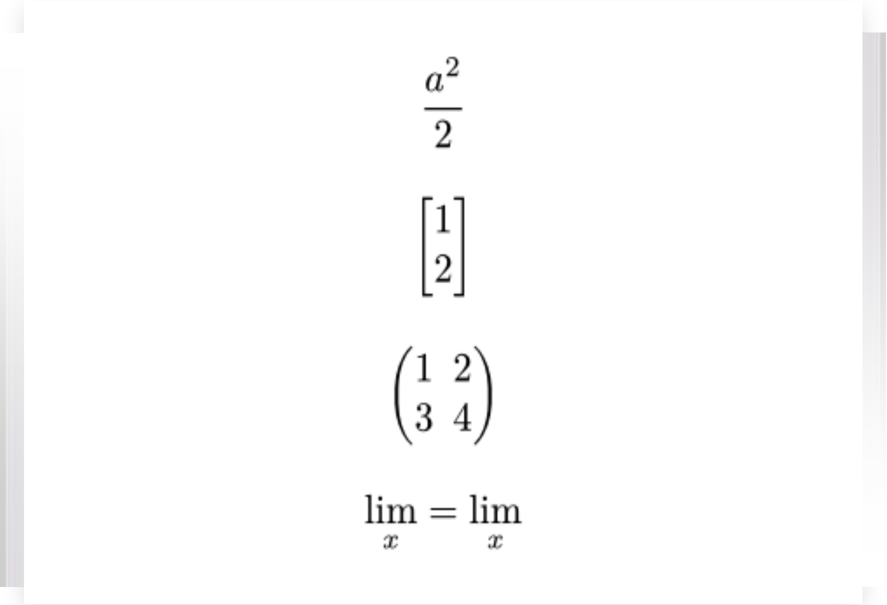
## Function calls

Math mode supports special function calls without the hash prefix. In these "math calls", the argument list works a little differently than in code:

- Within them, Typst is still in "math mode". Thus, you can write math directly into them, but need to use hash syntax to pass code expressions (except for strings, which are available in the math syntax).

- They support positional and named arguments, but don't support trailing content blocks and argument spreading.
- They provide additional syntax for 2-dimensional argument lists. The semicolon (`;`) merges preceding arguments separated by commas into an array argument.

```
$ frac(a^2, 2) $
$ vec(1, 2, delim: "[") $
$ mat(1, 2; 3, 4) $
$ lim_x =
    op("lim", limits: #true)_x $
```



$$\frac{a^2}{2}$$

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$$\lim_x = \lim_x$$

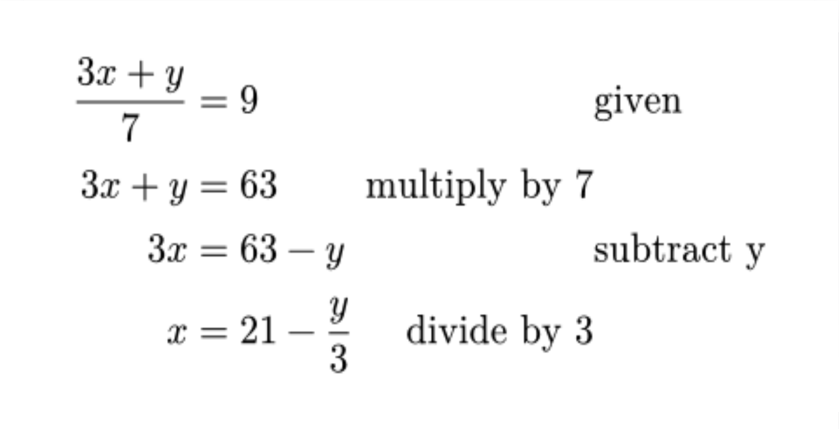
To write a verbatim comma or semicolon in a math call, escape it with a backslash. The colon on the other hand is only recognized in a special way if directly preceded by an identifier, so to display it verbatim in those cases, you can just insert a space before it.

Functions calls preceded by a hash are normal code function calls and not affected by these rules.

## Alignment

When equations include multiple *alignment points* (&), this creates blocks of alternatingly right- and left-aligned columns. In the example below, the expression `(3x + y) / 7` is right-aligned and `= 9` is left-aligned. The word "given" is also left-aligned because `&&` creates two alignment points in a row, alternating the alignment twice. `& &` and `&&` behave exactly the same way. Meanwhile, "multiply by 7" is right-aligned because just one `&` precedes it. Each alignment point simply alternates between right-aligned/left-aligned.

```
$ (3x + y) / 7 &= 9 && "given" \
  3x + y &= 63 & "multiply by 7" \
  3x &= 63 - y && "subtract y" \
  x &= 21 - y/3 & "divide by 3" $
```

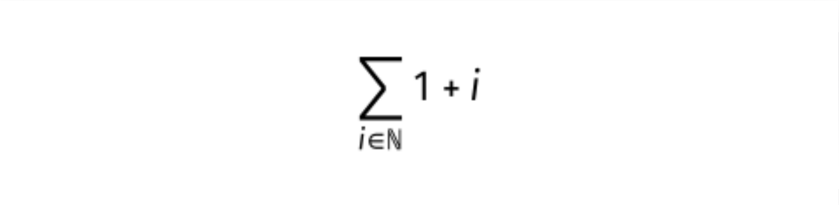


$$\begin{aligned} \frac{3x+y}{7} &= 9 && \text{given} \\ 3x+y &= 63 & \text{multiply by 7} \\ 3x &= 63 - y && \text{subtract y} \\ x &= 21 - \frac{y}{3} & \text{divide by 3} \end{aligned}$$

## Math fonts

You can set the math font by with a show-set rule as demonstrated below. Note that only special OpenType math fonts are suitable for typesetting maths.

```
#show math.equation: set text(font: "Fira Math")
$ sum_(i in NN) 1 + i $
```



$$\sum_{i \in \mathbb{N}} 1 + i$$

## Math module

All math functions are part of the `math` module, which is available by default in equations. Outside of equations, they can be accessed with the `math.` prefix.

## Definitions

- `accent` Attaches an accent to a base.
- `attach` Subscript, superscripts, and limits.
- `binom` A binomial expression.
- `cancel` Displays a diagonal line over a part of an equation.
- `cases` A case distinction.
- `class` Forced use of a certain math class.
- `equation` A mathematical equation.
- `frac` A mathematical fraction.
- lr Delimiter matching.
- `mat` A matrix.
- `op` A text operator in an equation.
- `primes` Grouped primes.
- roots Square and non-square roots.
- sizes Forced size styles for expressions within formulas.
- `stretch` Stretches a glyph.
- styles Alternate letterforms within formulas.
- underover Delimiters above or below parts of an equation.
- variants Alternate typefaces within formulas.
- `vec` A column vector.

Uppercase
Previous page

Accent
Next page

Documentation
Universe
About Us
Contact Us
Privacy
Terms and Conditions
Legal (Impressum)

Blog
GitHub
Discord
Mastodon
Bluesky
LinkedIn
Instagram

Made in Berlin