# Theoretic

## Contents



## 1 Summary

This package provides opinionated functions to create theorems and similar environments.

```
#theorem[This is a theorem.]
#proof[
  Ends with Equation? No Problem:
  $ norm(x) = sqrt( sum_(k = 1)^d x_k ) . $
]
#theorem(<thm:foo>)[Foo][This is a named theorem.]
#proof[@thm:foo[-]][
  - Ends with a list or enum? Easy.
]
```

*Theorem 1.1* This is a theorem.

*Proof.* Ends with Equation? No Problem:

$$\|x\| = \sqrt{\sum_{k=1}^{d} x_k}.$$ □

*Theorem 1.2 (Foo)* This is a named theorem.

*Proof of Theorem 1.2.*

- Ends with a list or enum? Easy. □

### 1.1 Setup

Put the following at the top of your document:

```
#import "@preview/theoretic:0.2.0" as theoretic: theorem, proof, qed
#show ref: theoretic.show-ref // Otherwise, references won't work.

// set up your needed presets
#let corollary = theorem.with(kind: "corollary", supplement: "Corollary")
#let example = theorem.with(kind: "example", supplement: "Example", number: none)
// ..etc
```

See `#theoretic.theorem()` (Section A.10) for a detailed description of customization options.

## 2 Features

- Except for `#show ref: theoretic.show-ref`, no "setup" is necessary. All configuration is achieved via parameters on the `#theoretic.theorem()` function. Use `theorem.with(..)` for your preset needs. → `#theoretic.theorem()`

- Automatic numbering. If your headings are numbered, it will use top-level heading numbers as the first component, otherwise it will simply number your theorems starting with Theorem 1.

```
#theorem(number: "!!")[
  Number can be overridden per-theorem.
]
#theorem(number: 400)[
  If a `number` is passed (as opposed to a string or
content),
]
#theorem[
  ...subsequent theorems will pick it up.
]
```

*Theorem !!* Number can be overridden per-theorem.

*Theorem 2.400* If a `number` is passed (as opposed to a string or content),

*Theorem 2.401* ...subsequent theorems will pick it up.

- Flexible References via specific supplements. → `#theoretic.show-ref()`

```
@thm:foo vs @thm:foo[-] vs @thm:foo[--] vs @thm:foo[!]
vs @thm:foo[!!] vs @thm:foo[!!!] vs @thm:foo[?] vs
@thm:foo[Statement]
```

Theorem 1.2 (Foo) vs Theorem 1.2 vs 1.2 vs Foo (Theorem 1.2) vs Foo (1.2) vs Foo vs Theorem vs Statement 1.2 (Foo)

- Custom outlines: Outline for headings *and/or* theorems. → `#theoretic.toc()`
  - Filter for specific kinds of theorem to create e.g. a list of definitions.
  - Optionally sorted alphabetically!
  - Theorems can have a different title for outlines (`theorem(toctitle: ..)`) and can even have multiple entries in a sorted outline.
  - Highly customizable! → `#theoretic.toc-entry()`
    - (And this customization can be reused for regular outlines) → `#theoretic.show-entry-as()`
- Exercise solutions: → `#theoretic.solutions()`
  - Every theorem environment can have a solution, which is shown in a separate section.
  - Solutions section automatically hides itself if there are no solutions to show.

```
#theorem(kind: "exercise", supplement: "Exercise",
solution: [
  // no cheating! //
])[
  Go look for the solution of this exercise at the end
of this document.
]
```

*Exercise 2.402* Go look for the solution of this exercise at the end of this document.[1]

- Automatic QED placement! → `#theoretic.proof()` & `#theoretic.qed()`

  In most cases, it should place the QED symbol appropriately automatically:

```
#proof[This is a proof. $x=y$]
#proof[
  This is a proof.
  $ x = y $
]
#proof[
  #set math.equation(numbering: "(1)")
  This is a proof.
  $ x = y $
]
```

*Proof.* This is a proof. $x = y$ □

*Proof.* This is a proof.

$$x = y$$ □

*Proof.* This is a proof.

$$x = y \tag{1}$$ □

```
#proof[
  This is a proof.
  - #lorem(3)
```

[1] Solution in Appendix

```
]
#proof[
  This is a proof.
  - #lorem(3) $ x = y $
]
#proof[
  This is a proof.
  + #lorem(3)
    + #lorem(3)
      + #lorem(3)
        + #lorem(3)
]
```

*Proof.* This is a proof.
- Lorem ipsum dolor. □

*Proof.* This is a proof.
- Lorem ipsum dolor.

$$x = y$$ □

*Proof.* This is a proof.
1. Lorem ipsum dolor.
    1. Lorem ipsum dolor.
        1. Lorem ipsum dolor.
            1. Lorem ipsum dolor. □

Specifically, it works for lists, enums, and unnumbered block equations, which may be nested. If your proof ends wit some other block, you should might want to place a `#qed()` manually. For proper alignment with a block equation, use

```
#set math.equation(numbering: (..) => {qed()}, number-align: bottom)
```

placed directly in front of the equation.

- Any theorem can be restated. → `#theoretic.restate()`

```
theoretic.restate(<thm:foo>)
// the prefix links to the original
```

*Theorem 1.2 (Foo)* This is a named theorem.

# 3 Preset Styles

Use with

```
#import "lib.typ" as theoretic
#import theoretic.styles.<name>: *
```

## 3.1 `basic`

**Theorem 3.1.** Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

**Proposition 3.2.** Lorem ipsum dolor.

**Lemma 3.3.** Lorem ipsum dolor.

**Corollary 3.4.** Lorem ipsum dolor.

**Definition 3.5.** Lorem ipsum dolor.

**Exercise 3.6.** Lorem ipsum dolor.

**Algorithm 3.7.** Lorem ipsum dolor.

**Axiom 3.8.** Lorem ipsum dolor.

**Example.** Lorem ipsum dolor.

**Counter-Example.** Lorem ipsum dolor.

**Remark.** Lorem ipsum dolor.

**Note.** Lorem ipsum dolor.

**Claim.** Lorem ipsum dolor.

*Proof.* Lorem ipsum dolor. □

**Theorem 3.9 (Title).** Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

**Proposition 3.10 (Title).** Lorem ipsum dolor.

**Lemma 3.11 (Title).** Lorem ipsum dolor.

**Corollary 3.12 (Title).** Lorem ipsum dolor.

**Definition 3.13 (Title).** Lorem ipsum dolor.

**Exercise 3.14 (Title).** Lorem ipsum dolor.

**Algorithm 3.15 (Title).** Lorem ipsum dolor.

**Axiom 3.16 (Title).** Lorem ipsum dolor.

**Example (Title).** Lorem ipsum dolor.

**Counter-Example (Title).** Lorem ipsum dolor.

**Remark (Title).** Lorem ipsum dolor.

**Note (Title).** Lorem ipsum dolor.

**Claim (Title).** Lorem ipsum dolor.

*Proof of Title.* Lorem ipsum dolor. □

## 3.2 `fancy`

This style is intended[2] for use with a font that supports `stretch: 85%` and `weight: "semibold"`. It is here shown using the font "Besley*", which you can download on GitHub.

**Theorem 3.17** Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

**Proposition 3.18** Lorem ipsum dolor.

**Lemma 3.19** Lorem ipsum dolor.

**Corollary 3.20** Lorem ipsum dolor.

**Definition 3.21** Lorem ipsum dolor.

**Exercise 3.22** Lorem ipsum dolor.

**Algorithm 3.23** Lorem ipsum dolor.

**Axiom 3.24** Lorem ipsum dolor.

Example Lorem ipsum dolor.

Counter-Example Lorem ipsum dolor.

Remark Lorem ipsum dolor.

Note Lorem ipsum dolor.

Claim Lorem ipsum dolor.

*Proof.* Lorem ipsum dolor. □

**Theorem 3.25** **Title** Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

**Proposition 3.26** **Title** Lorem ipsum dolor.

**Lemma 3.27** **Title** Lorem ipsum dolor.

**Corollary 3.28** **Title** Lorem ipsum dolor.

**Definition 3.29** **Title** Lorem ipsum dolor.

**Exercise 3.30** **Title** Lorem ipsum dolor.

**Algorithm 3.31** **Title** Lorem ipsum dolor.

**Axiom 3.32** **Title** Lorem ipsum dolor.

Example Title Lorem ipsum dolor.

Counter-Example Title Lorem ipsum dolor.

Remark Title Lorem ipsum dolor.

Note Title Lorem ipsum dolor.

Claim Title Lorem ipsum dolor.

*Proof of Title.* Lorem ipsum dolor. □

[2] It still looks okay in other fonts, but it does not reach full potential.

# A Detailed Documentation of all Exported Symbols

## A.1 fmt-body

Function to format the body

Default value of `theorem.fmt-body`.

### Parameters

```
fmt-body(⟨body⟩: content, ⟨solution⟩: content) → content
```

**⟨body⟩** content Positional Theorem content.

## A.2 fmt-prefix

Function to run at beginning of theorem.

Default value of `theorem.fmt-prefix`.

```
#fmt-prefix([Theorem], [1.34], none)...

#fmt-prefix([Theorem], [1.34], [Pythagoras])...
```

*Theorem 1.34* ...

*Theorem 1.34 (Pythagoras)* ...

### Parameters

```
fmt-prefix(⟨supplement⟩: content, ⟨number⟩: content none, ⟨title⟩: content none) → content
```

## A.3 proof

This is just `theorem()` with different defaults.

```
#proof[#lorem(5)]
#proof[@pythagoras[!]][#lorem(6)]
```

*Proof.* Lorem ipsum dolor sit amet. □

*Proof of Pythagoras (Theorem A.3).* Lorem ipsum dolor sit amet, consectetur. □

### Parameters

```
proof(
  fmt-prefix: function,
  fmt-suffix: function none,
  kind: "proof" string,
  supplement: "Proof" content,
  number: none auto none integer content,
  ⟨..args⟩: arguments
) → content
```

**fmt-prefix:** proof-fmt-prefix function

**fmt-suffix:** qed.with(force: false) function or none

## A.4 proof-fmt-prefix

Function to run at beginning of proof.

Default value of `proof.fmt-prefix`.

```
#proof-fmt-prefix([Proof], none, none)...

#proof-fmt-prefix([Proof], none, [@pythagoras])...
```

*Proof.* ...

*Proof of Theorem A.3 (Pythagoras).* ...

**Parameters**

`proof-fmt-prefix(`**⟨supplement⟩:** `content`, **⟨number⟩:** `content` `none`, **⟨title⟩:** `content` `none` `) → content`

## A.5 qed

Place a QED mark and clear the `_thm_needs_qed` flag, so that the theorem environment itself won't place one.

See `proof.fmt-suffix`.

**Parameters**

`qed(`**suffix:** `[#h(1fr)$square$]` `content`, **force:** `boolean` `) → content`

**force:** `true` `boolean` Whether to place suffix no matter the `_thm_needs_qed` flag.

## A.6 restate

Re-state a theorem.

It will reuse the original kind, supplement, number, title, body, and styling. It will *not* re-emit the solution or label, and it will use `toctitle: none` to avoid duplicate toc entries.

```
#let proposition = theorem.with(
  kind: "proposition",
  supplement: "Proposition",
  fmt-prefix: (s,n,t) => smallcaps({ s ; if n ≠ none
[ #n]; if t ≠ none [ (#t)]; h(1em) })
)
#proposition(<funky>)[Funky!][Blah _blah_ blah.]
Restated:
#restate("funky")
Restated with customizations:
#restate(<funky>, fmt-body: (b, s) => text(red, {b; s}))
```

Proposition A.1 (Funky!) Blah *blah* blah.

Restated:

Proposition A.1 (Funky!) Blah *blah* blah.

Restated with customizations:

Proposition A.1 (Funky!) Blah *blah* blah.

**Parameters**

`restate(`**⟨label⟩:** `label` `string`, **⟨..args⟩**`) → content`

**⟨label⟩** `label` or `string` Positional Label of the theorem to restate.

**⟨..args⟩** Positional

Override arguments for the theorem function.

(Setting body, solution, kind, supplement, number, title or toctitle here is not recommended.)

## A.7 show-entry-as

Helper function to adapt actual outlines to look the same as those made with `toc()`. This is useful if you want to have e.g. a list of figures and a list of definitons adn want them to share their style.

Note: Fot typst versions <= 0.12, this function is a bit "hacky" and might not always work. (It deconstructs the `outline.entry` based on heuristics.)

```
#import theoretic: show-entry-as, toc-entry

#outline(target: figure, title: [Typst Default])

#show outline.entry: show-entry-as(toc-entry.with(hanging-indent: 60pt, /*...*/))
#outline(target: figure, title: [Using `theoretic.toc-entry`])

#figure(
  caption: [Example Figure],
  block(height: 2em, width: 100%, fill: gradient.linear(..color.map.viridis))
)
```

**Typst Default**

Figure 1 Example Figure ..... A3

**Using `theoretic.toc-entry`**

Figure 1 Example Figure ..... A3

Figure 1: Example Figure

**Parameters**

`show-entry-as(⟨toc-entry⟩: function)`

**⟨toc-entry⟩** `function` Positional

Customize `toc-entry()` used.

Expects a function taking five positional arguments (level, target, prefix, body, page).

## A.8 show-ref

Show-rule-function to be able to `@` labelled theorems.

Use via `` `#show ref: show-ref` `` at the beginning of your document.

```
#show ref: theoretic.show-ref
#theorem(label: <fact>, supplement: "Fact")[#lorem(2)]
#theorem(label: <pythagoras>,"Pythagoras")[#lorem(2)]
#theorem(label: <zl>, title: "Only Named", number: none)
[#lorem(2)]
#theorem(label: <y>, number: "Y")[#lorem(2)]
#theorem(label: "5", number: none)[#lorem(2)]

As a consequence of @fact and @pythagoras[!!]...
```

*Fact A.2* Lorem ipsum.

*Theorem A.3 (Pythagoras)* Lorem ipsum.

*Theorem (Only Named)* Lorem ipsum.

*Theorem Y* Lorem ipsum.

*Theorem* Lorem ipsum.

As a consequence of Fact A.2 and Pythagoras (A.3)...

The reference can be controlled via the supplement passed:

| | Both | Without Title | Without Number | Neither |
|---|---|---|---|---|
| `@ref` (Full) | Theorem A.3 (Pythagoras) | Fact A.2 / Theorem Y | Theorem (Only Named) | Theorem |
| `@ref[-]` (Compact) | Theorem A.3 | Fact A.2 / Theorem Y | Theorem (Only Named) | Theorem |

| | | | | |
|---|---|---|---|---|
| `@ref[--]` (Number) | A.3 | A.2 / Y | (Only Named) | Theorem |
| `@ref[!]` (Inverted) | Pythagoras (Theorem A.3) | Fact A.2 / Theorem Y | Only Named (Theorem) | Theorem |
| `@ref[!!]` (Compact Inverted) | Pythagoras (A.3) | Fact A.2 / Theorem Y | Only Named | Theorem |
| `@ref[!!!]` (Name) | Pythagoras | Fact A.2 / Theorem Y | Only Named | Theorem |
| `@ref[?]` (Kind) | Theorem | Fact / Theorem | Theorem | Theorem |
| `@ref[Custom]` (Custom Supplement) | Custom A.3 (Pythagoras) | Custom A.2 / Custom Y | Custom (Only Named) | Custom |

Note: the fact that references and links in this document are underlined in gray is achieved with a separate `@show link: it => underline(..)` rule, and not because of this function.

**Parameters**

`show-ref(`**`⟨it⟩:`** `ref` `)`

## A.9 solutions

List all solutions, if any.

See Section B for how it looks. Currently not customizable, working on it.

**Parameters**

`solutions(`**`title:`** `"Solutions"` `content` `)` → `content`

## A.10 theorem

Theorem Environment

```
#set heading(numbering: none)

#theorem[If the headings are not numbered, theorem
numbering starts at 1.]

= Heading
#theorem(title: "Pythagoras")[
  Given a right-angled triangle, the length
  of the hypothenuse squared is equal to the
  sum of the squares of the remainig sides'
  lengths.
]
```

*Theorem 1* If the headings are not numbered, theorem numbering starts at 1.

### Heading

*Theorem 2 (Pythagoras)* Given a right-angled triangle, the length of the hypothenuse squared is equal to the sum of the squares of the remainig sides' lengths.

**Parameters**

```
theorem(
  fmt-prefix: function,
  fmt-body: function,
  fmt-suffix: function none,
  block-args: dict,
  kind: string,
  supplement: content,
  number: auto none integer content,
  title: none content,
  toctitle: auto none content array,
  label: none label string,
  solution: none content,
```

```
    ⟨..unnamed-and-body⟩: arguments
  ) → content
```

**fmt-prefix:** fmt-prefix function

**fmt-body:** fmt-body function

**fmt-suffix:** none function or none

Will be called at the end of the theorem if `_thm_needs_qed` hasn't been cleared. (E.g. by `qed()`)

**block-args:** (:) dict Arguments to pass to the `#block[]` containing the theorem.

**kind:** "theorem" string Used for filtering e.g. when creating table of theorems.

**supplement:** "Theorem" content

What to label the environment.

It is recommended to keep `kind` and `supplement` matching (except for "subtypes", e.g. one might have the kind of "Example" and "Counter-Example" both as `"example"`)

**number:** auto auto or none or integer or content

- If `auto`, will continue numbering from last numbered theorem.
- If `integer`, it will continue the numbering of later theorems from the given number.
- If `content`, it is shown as-is, with no side-effects.

```
#let corollary = theorem.with(
  kind: "corollary",
  supplement: "Corollary")

#corollary[#lorem(2)]

#corollary(number: none)[Skip number]
#corollary[Resume numbering]

#corollary(number: "X")[Custom "number"]
#corollary[Resume numbering]

#corollary(number: 10)[Set number]
#corollary[Continue from set number]
```

*Corollary A.3* Lorem ipsum.

*Corollary* Skip number

*Corollary A.4* Resume numbering

*Corollary X* Custom "number"

*Corollary A.5* Resume numbering

*Corollary A.10* Set number

*Corollary A.11* Continue from set number

**title:** none none or content Title of the Theorem. Usually shown in parentheses after the number.

*This can also be passed as apositional argument.*

**toctitle:** auto auto or none or content or array

Title of the Theorem to be used in outlines.

- `auto` to use the `title`.
- `none` to hide it from the outlines.

If you pass an array, in *sorted* outlines (`toc.sort`) it will be split into multiple entries. All but the first one are marked as secondary.

```
#theorem(
  title: [A to Z],
  toctitle: ([AAAAA], [ZZZZZZ])
```

```
)[
  Compare how this appears in different outlines!
]
```

*Theorem A.12 (A to Z)* Compare how this appears in different outlines!

**label:** none `none` or `label` or `string`

Label (for references)

*This can also be passed as a positional argument. In that case it must be a `label` and not a `string`.*

NB: Simply putting a `<label>` after the `#theorem[]` does not work for referencing.

**solution:** none `none` or `content`

Optional Solution. See also `solutions()`.

```
#theorem(solution: [This will show up wherever
`#theoretic.solutions()` is placed.])[#lorem(5)]
```

*Theorem A.13* Lorem ipsum dolor sit amet.[3]

**⟨..unnamed-and-body⟩** `arguments` Positional

The last positional argument given is used as the theorem body.

Other positional arguments are used for the title and label, depending on their type.

```
// Any of these work:
#theorem(<positional>)[Positional][#lorem(4)]
#theorem(label: <named>, title: [Named])[#lorem(4)]
#theorem([Mixed], label: <mixed>)[#lorem(4)]
```

*Theorem A.14 (Positional)* Lorem ipsum dolor sit.

*Theorem A.15 (Named)* Lorem ipsum dolor sit.

*Theorem A.16 (Mixed)* Lorem ipsum dolor sit.

## A.11 toc

Create an outline that includes named theorems.

Can be styled with show rules for `outline.entry()`. See the source code of this manual for an example.

```
#heading(outlined: false, level: 3)[
  Contents
]
#toc(depth: 1)
```

[3] Solution in Appendix

### Parameters

```
toc(
  depth: integer,
  exclude: array (string),
  level: integer auto,
  toc-entry: function,
  sort: bool
) -> content
```

**`depth: 2`** `integer` Maximum depth of headings to conisder

**`exclude: ("proof", "solution")`** `array (string)` list of `theorem.kinds` to ignore.

```
#heading(outlined: false, level: 3)[
  Table of Examples
]
#toc(
  depth: 0,
  exclude: ("proof", "solution", "theorem")
)
```

**Table of Examples**



**`level: auto`** `integer` or `auto` Fake level to use for theorems. If `auto`, it will use `depth + 1`.

**`toc-entry: toc-entry`** `function`

Customize `toc-entry()` used.

Expects a function taking five positional arguments (level, target, prefix, body, page).

**`sort: false`** `bool` Whether to sort the entries alphabetically.

Only respected if `depth` is 0.

If true, this will also split entries where `toctitle` is an array into separate entries.

```
#theorem("Z")[Blah blah.]
#theorem("A")[Blah blah.]
#heading(outlined: false, level: 3)[
  Sorted Table of Theorems
]
#set text(size: 9pt)
#toc(
  depth: 0,
  sort: true,
  toc-entry: toc-entry.with(hanging-indent: 60pt),
)
```

*Theorem A.17 (Z)* Blah blah.

*Theorem A.18 (A)* Blah blah.

**Sorted Table of Theorems**



## A.12 toc-entry

Create a toc entry.

Pass this to `toc()` using `.with(..)` to customize the `fmt-` parameters used.

This is used because since Typst 0.13, it is no longer possible to call outline.entry outside of an actual ourline element, and one “cannot outline metadata”.

This manual uses

```
set par(justify: false)
let indents = (0pt, 15pt, 37pt)
let hang-indents = (15pt, 22pt, 54pt)
let text-styles = ((weight: 700), (size: 10pt), (size: 9pt, weight: 500), (size: 9pt, fill: luma(20%)), )
theoretic.toc(toc-entry: theoretic.toc-entry.with(
  indent: (level) => { indents.at(level - 1) },
  hanging-indent: (level) => { hang-indents.at(level - 1) },
  fmt-prefix: (prefix, level, _s) => {
    set text(..text-styles.at(level - 1), number-width: "tabular")
    prefix
    h(4pt)
  },
  fmt-body: (body, level, _s) => {
    set text(..text-styles.at(level - 1))
    body
  },
  fmt-fill: (level, _s) => {
    if level == 2 {
      set text(..text-styles.at(2))
      box(width: 1fr, align(right, repeat(gap: 9pt, justify: false, [.])))
    }
  },
  fmt-page: (page, level, _s) => {
    set text(..text-styles.at(level - 1), number-width: "tabular")
    box(width: 18pt, align(right, [#page]))
  },
  above: (level) => {
    if level == 1 {
      auto // paragraph spacing
    } else {
```

```
      7pt
    }
  },
  below: auto,
))
```

### Parameters

```
toc-entry(
  ⟨level⟩: int,
  ⟨target⟩: location,
  ⟨prefix⟩: content none,
  ⟨body⟩: content,
  ⟨page⟩: content,
  secondary: boolean,
  indent: relative length function,
  hanging-indent: relative length function auto,
  above: relative length function,
  below: relative length function,
  fmt-prefix: (prefix, level, secondary) => if prefix != none {
    prefix
    h(0.5em, weak: false)
  } function,
  fmt-body: (body, level, secondary) => if secondary [(#body) ] else [#body ] function,
  fmt-fill: (level, secondary) => box(width: 1fr, repeat[.~]) function,
  fmt-page: (page, level, secondary) => page function
)
```

**`secondary: false`** `boolean`

This is `true` for entries where the toc-title is an array, the entry was split and this is *not* the first one (in order specified).

**`indent: 1em`** `relative length` or `function` How much to indent each entry.

- If `relative length`, it will be multiplied with level - 1.
- If `function`, will be called with the level as argument.

**`hanging-indent: auto`** `relative length` or `function` or `auto`

How much more to indent subsequent lines (in addition th `toc-entry.indent`).

If the prefix is shorter than this, this will lead to a gap between prefix and body; If the prefix is longer, the body will start immediately after the prefix.

- If `function`, will be called with the level as argument.
- If `auto`, will use the width of the prefix

```
#let example-entry = theoretic.toc-entry.with(1, here(),
[Section 1.], lorem(6), [0])
#let example-entry-2 = theoretic.toc-entry.with(2,
here(), [Section 1.1.], lorem(6), [0])

// aligned with end of prefix
#example-entry(hanging-indent: auto)
#example-entry-2(hanging-indent: auto)

#example-entry(hanging-indent: 1em)
#example-entry-2(hanging-indent: 1em)
#example-entry(hanging-indent: 80pt)
#example-entry-2(hanging-indent: 80pt)
```

**above:** `0.7em` `relative length` or `function` If `function`, will be called with the level as argument.

**below:** `0.7em` `relative length` or `function` If `function`, will be called with the level as argument.

### A.13 thm-counter `counter`

Counts theorems.

In most cases, it is not neccesary to reset this manually, it will get updated accordingly if you pass an integer to `theorem.number`.

## B Solutions

*Solution of Exercise 2.402.* Yay! you found it!

*Solution of Theorem A.13.* This will show up wherever `#theoretic.solutions()` is placed.