```typ
#set page(width: 10cm, height: auto)
#set heading(numbering: "1.")

= Fibonacci sequence
The Fibonacci sequence is defined through the
recurrence relation $F_n = F_(n-1) + F_(n-2)$.
It can also be expressed in _closed form:_

$ F_n = round(1 / sqrt(5) phi.alt^n), quad
  phi.alt = (1 + sqrt(5)) / 2 $

#let count = 8
#let nums = range(1, count + 1)
#let fib(n) = (
  if n <= 2 { 1 }
  else { fib(n - 1) + fib(n - 2) }
)

The first #count numbers of the sequence are:

#align(center, table(
  columns: count,
  ..nums.map(n => $F_#n$),
  ..nums.map(n => str(fib(n))),
))
```

```typ
#import "../dystac.typ" : dystac, input, dynamic

#set heading(numbering: "1.")

#show math.equation : it => html.frame(it)

#dystac() // javascript script for dynamic compilation of typst

= Fibonacci sequence
The Fibonacci sequence is defined through the
recurrence relation $F_n = F_(n-1) + F_(n-2)$.
It can also be expressed in _closed form:_

$ F_n = round(1 / sqrt(5) phi.alt^n), quad
  phi.alt = (1 + sqrt(5)) / 2 $

#let count = 8

The first #input(name: "count", default: count) numbers of the sequence are:

#dynamic(variables : (count: count),
⟶\`\`\`typ
// i supposed to be positive
// j supposed to be positive
// return a boolean equal to i <= j
// without using the symbol
#let lt(i, j) = {
  if i == 0 { return true }
  if j == 0 { return false }
  lt(i - 1, j - 1)
}
#let nums = range(1, count + 1)
#let fib(n) = {
  if lt(n, 2) {
    1
  } else { fib(n - 1) + fib(n - 2) }
}
#align(center, table(
  columns: count,
  ..nums.map(n => $F_#n$),
  ..nums.map(n => str(fib(n))),
))
\`\`\`)
```