The number of ways to partition \(a+b\) things into \(a\) things and \(b\) things is 
\((a+b)! / (a!b!)\) where ! is the factorial function. First without time.

\[
x := (a+b)! / (a!b!) \iff
\]

1. If \(a = 0\) then \(x := 1\)
   
2. Else, \(a := a - 1.\) \(x := (a+b)! / (a!b!)\).

(\(a := a + 1.\) \(x := x \times (a+b)/a\) \(\exists \) specialization)

The assignment \(x := (a+b)! / (a!b)!\) means \(x' := (a+b)! / (a!b!)\) \& \(a' := a \land b' := b\). On the right side it is a recursive call. Stating it as an assignment makes the proof easy: just use the substitution law and simplify. The proof is by cases. First case:

\[
a := 0 \land (x := 1) \Rightarrow (x := (a+b)! / (a!b!)) \land a' = a \land b' = b
\]

use 0! = 1

Second case, starting with the right side:

\[
a := 0 \land x' = (a-1+b)! / ((a-1)!b!)x(a+b)/a \land a' = a \land b' = b
\]

simplify

\[
a := 0 \land x' = (a+b)! / (a!b!) \land a' = a \land b' = b
\]

specialization

\[
\Rightarrow x := (a+b)! / (a!b!)\]

Now the time.

\[
t' = t + a \iff
\]

1. If \(a = 0\) then \(x := 1\)
   
2. Else \(a := a - 1.\) \(t := t + 1.\) \(t' = t + a.\) \(x := x \times (a+b)/a\) \(\exists \) specialization

Proof by cases. First case:

\[
a := 0 \land (x := 1) \Rightarrow t' = t + a
\]

definition of assignment

\[
\Rightarrow t' = t + a
\]

Second case, starting with the right side:

\[
a := 0 \land (a := a - 1.\) \(t := t + 1.\) \(t' = t + a.\) a := a + 1. x := x \times (a+b)/a
\]

assignment

\[
\Rightarrow a := a + 1.\) \(x := x \times (a+b)/a \land a' = a \land b' = b \land t' = t
\]

substitution law 3 times

\[
\Rightarrow a := a + 1.\) \(x := x \times (a+b)/(a+1) \land a' = a + 1 \land b' = b \land t' = t
\]

sequential composition

\[
\Rightarrow a := a + 1.\) \(x := x \times (a+b)/(a+1) \land a' = a + 1 \land b' = b \land t' = t
\]

one point 4 times

\[
\Rightarrow t' = t + a
\]

When refining \(x := (a+b)! / (a!b!)\), there was no time variable. Adding the time variable, we cannot write this as an assignment, because that would mean \(t' = t\). We can put the result and the timing together as

\[
x' := (a+b)! / (a!b!) \land a' = a \land b' = b \land t' = t + a
\]

or as

\[
x := (a+b)! / (a!b!).\) \(t := t + a
\]

Here is a solution that is symmetric in \(a\) and \(b\).

\[
x := (a+b)! / (a!b!) \iff
\]

1. If \(a = 0 \lor b = 0\) then \(x := 1\)
   
2. Else \(a := a - 1.\) \(b := b - 1.\) \(x := (a+b)! / (a!b)!.\)

   \(a := a + 1.\) \(b := b + 1.\) \(x := x(a/b) \times (a+b)(a+b-1)(a+b)\)

And its execution time is smaller: \(\min\ a\ b\).
Here is a solution with the same execution time and its recursion does not require a stack.

\[ x' = (a+b)! / (a!b!) \land t' = t + \min a b \leq \]
\[ x:= 1. \quad x' = x \times (a+b)! / (a!b!) \land t' = t + \min a b \leq \]
\[ x' = x \times (a+b)! / (a!b!) \land t' = t + \min a b \leq \]
\[ \text{if } a=0 \lor b=0 \text{ then } \text{ok} \]
\[ \text{else } x:= x/a \times (a+b-1) \times (a+b). \quad a:= a-1. \quad b:= b-1. \quad t:= t+1. \]
\[ x' = x \times (a+b)! / (a!b!) \land t' = t + \min a b \leq \]

Now, here is a for-loop solution. Define

\[ I_k = x = (a+k)! / (a!k)! \]

Then

\[ x' = (a+b)! / (a!b!) \leq x:= 1. \quad I_0 \Rightarrow I' b \]
\[ I_0 \Rightarrow I' b \leq \text{for } k:= 0;..b \text{ do } I_k \Rightarrow I'(k+1) \text{ od} \]
\[ I_k \Rightarrow I'(k+1) \leq x:= x \times (a+k+1)/(k+1) \]

with timing \( t' = t+b \).

Finally, here are two functional solutions. Define

\[ f = \langle a, b : \text{nat} \rightarrow (a+b)! / (a!b)! \rangle \]

Then

\[ f \ a \ b = \text{if } a=0 \text{ then } 1 \text{ else } f(a-1) \times (a+b) / a \leq f \]
with execution time \( a \). For execution time \( \min a b \)

\[ f \ a \ b = \text{if } a=0 \lor b=0 \text{ then } 1 \text{ else } f(a-1) \times (a+b-1) \times (a+b) / a / b \leq f \]