Let \( x \) and \( y \) be integer variables. Find the exact precondition for \( x' = y \) to be refined by \( y := 1 \).

\[
\forall x', y'. x' = y \iff (y := 1)
\]

\[
\forall x', y'. x' = y \iff x' = x \land y' = 1
\]

\( x = y \)

Let \( x \) and \( y \) be integer variables. Prove \( x + y = 5 \) is an invariant for
\[
x := x + 1. \quad y := y - 1
\]

\[
(x + y = 5 \Rightarrow x' + y' = 5) \iff (x := x + 1. \quad y := y - 1)
\]

\( x = y \)

The notation \( \textbf{do } P \textbf{ while } b \textbf{ od} \) has been used as a loop construct that is executed as follows. First, \( P \) is executed; then \( b \) is evaluated, and if its value is \( \top \), then execution is repeated, and if its value is \( \bot \), then execution is finished. Let \( m \) and \( n \) be integer variables. Prove
\[
m := m + n - 10. \quad n := 10 \iff \textbf{do } m := m - 1. \quad n := n + 1 \textbf{ while } n \neq 10 \textbf{ od}
\]

Apparently, we are not talking about time in this question; we don't have variable \( t \). So we can't talk about termination or nontermination, because those are timing issues.

I prove
\[
m := m + n - 10. \quad n := 10 \iff \textbf{do } m := m - 1. \quad n := n + 1 \textbf{ if } n \neq 10 \textbf{ then } m := m + n - 10. \quad n := 10 \textbf{ else } \textbf{ ok fi}
\]

starting with the right side.
\[
m := m - 1. \quad n := n + 1. \quad \textbf{if } n \neq 10 \textbf{ then } m := m + n - 10. \quad n := 10 \textbf{ else } \textbf{ ok fi}
\]

replace \( n := 10 \) and \( \textbf{ ok fi} \)
\[
m := m - 1. \quad n := n + 1. \quad \textbf{if } n \neq 10 \textbf{ then } m := m + n - 10. \quad m' = m \land n' = 10 \textbf{ else } m' = m \land n' = n \textbf{ fi}
\]

substitution law in \( \textbf{ then part } \)
\[
m := m - 1. \quad n := n + 1. \quad \textbf{if } n \neq 10 \textbf{ then } m' = m + n - 10 \land n' = 10 \textbf{ else } m' = m \land n' = n \textbf{ fi}
\]

substitution law twice
\[
\textbf{if } n \neq 10 \textbf{ then } m' = m + n - 10 \land n' = 10 \textbf{ else } m' = m + n - 10 \land n' = 10 \textbf{ fi}
\]

\( \textbf{ case idempotent } \)
\[
m' = m + n - 10 \land n' = 10 \textbf{ definition of assignment and sequential composition}
\]
\[
m := m + n - 10. \quad n := 10
\]

If we were talking about time, we couldn't leave the specification as
\[
m := m + n - 10. \quad n := 10
\]

because, by the recursive measure, that takes no time, and the implementation
\[
\textbf{ do } m := m - 1. \quad n := n + 1 \textbf{ while } n \neq 10 \textbf{ od}
\]

is a loop with time \( (n < 10 \Rightarrow t' = t + 10 - n) \land (n \geq 10 \Rightarrow t' = \infty) \).
Let $L$ be a list-of-integers variable, $L: [*int]$ . Here is a \texttt{for}\text{-}loop that changes all the negative items of $L$ to 0 , and otherwise leaves $L$ unchanged.

\texttt{for n:= 0;..#L do if L i < 0 then L := i \rightarrow 0 \mid L else ok fi od}

State formally what must be proven in order to prove that this program is correct. You do not need to prove it; you just need to say what must be proven.

TYPO - Sorry about that. Announced in the test: the \texttt{for}\text{-}loop should be

\texttt{for i:= 0;..#L do if L i < 0 then L := i \rightarrow 0 \mid L else ok fi od}

§ Define $F_i$ as

$$F_i \equiv \#L = \#L \land (\forall j: 0,..i \cdot L' j = L j) \land (\forall j: i,..\#L \cdot L' j = (L j)^{0})$$

Then $F_0$ specifies the problem. We must prove

$$F_i \iff i: 0,..\#L \land (\text{if } L i < 0 \text{ then } L := i \rightarrow 0 \mid L \text{ else ok fi. } F(i+1))$$

and

$$F(\#L) \iff \text{ok}$$