Here is how I might write an informal proof of $GCD3$. The proof probably looks quite different from the one you wrote. My proof style will seem strange, but you should be able to understand the proof.

THEOREM $GCD3$

$\langle 1 \rangle 1$. It suffices to assume that $m$ and $n$ are positive integers and prove $GCD(m, n) = GCD(m, n - m)$
PROOF: By definition of $GCD3$.

$\langle 1 \rangle 2$. Assuming $d$ divides $m$ and $n$, we can prove that it divides $n - m$.

$\langle 2 \rangle 1$. Choose $q$ and $r$ such that $m = q * d$ and $n = r * d$.
PROOF: $q$ and $r$ exist by the assumption of $\langle 1 \rangle 2$.

$\langle 2 \rangle 2$. $n - m = (r - q) * d$
PROOF: By $\langle 2 \rangle 1$ and simple algebra.

$\langle 2 \rangle 3$. QED
PROOF: $\langle 2 \rangle 2$ implies that $d$ divides $n - m$.

$\langle 1 \rangle 3$. Assuming $d$ divides $m$ and $n - m$, we can prove that it divides $n$.
The proof is similar to that of $\langle 1 \rangle 2$ and is left as an exercise.

$\langle 1 \rangle 4$. QED

$\langle 2 \rangle 1$. $DivisorsOf(m) \cap DivisorsOf(n) = DivisorsOf(m) \cap DivisorsOf(n)$
PROOF: By $\langle 1 \rangle 2$, $\langle 1 \rangle 3$, and the definition of $DivisorsOf$.

$\langle 2 \rangle 2$. QED
PROOF: $\langle 2 \rangle 1$ and the definition of $GCD$ imply $GCD(m, n) = GCD(m, n - m)$, which by $\langle 1 \rangle 1$ is what we have to prove.

You have probably realized that QED stands for what has to be proved. For example, in the step numbered $\langle 2 \rangle 3$, the QED stands for "$d$ divides $n - m$", which is what must be proved in order to prove step $\langle 1 \rangle 2$. Besides using this convention, the proof probably differs from the one you wrote in two important ways.

- The proof is hierarchically structured. I first decomposed the proof of $GCD3$ into the four level-1 steps $\langle 1 \rangle 1$–$\langle 1 \rangle 4$. If the proof of a step isn't very simple, it is decomposed it further. Thus, the proof of $\langle 1 \rangle 2$ is broken into the three level-2 steps $\langle 2 \rangle 1$–$\langle 2 \rangle 3$. Had this been a more difficult theorem, or if I were writing for less sophisticated readers who would find this proof hard to understand, I would have decomposed it into more levels. At the bottom of the hierarchical structure are *leaf* proofs, which are short, simple paragraphs.

  In standard unstructured proofs, adding an explanation of why a statement is true makes the proof longer and therefore harder to read. When writing such a proof, there is a continual tradeoff between adding extra detail needed by some readers versus keeping the proof short and easier to read for other readers. With hierarchical structuring, we can add extra detail with extra levels, which readers can easily skip. This would be especially true if the proof were written as hypertext, so lower levels could easily be hidden and revealed when desired.

- Every step in the proof has a name like $\langle 1 \rangle 3$ or $\langle 2 \rangle 2$. Whenever I use a step that has already been proved or is an assumption, I refer to it by name.

  Naming all facts that are used in a leaf proof makes the proof easier to understand. In an ordinary paragraph proof, it is hard to make clear what facts justify an assertion. Usually, the proof at best hints at why a statement follows from what precedes it. The reader must figure out for herself what facts are needed. In this proof, every required fact is named.

  Naming facts that are used also makes writing the proof easier. Few of us are Mozarts of mathematics, able to write a proof from start to finish without having to go back and rewrite anything. If we have to change a step, it is easy to see what else must be changed by finding where that step has been used.

This informal structured proof should serve to motivate the way formal proofs are written in TLA$^+$. The Toolbox allows us to read proofs as hypertext, hiding and revealing levels at will. In addition to making it easier for a human to read, naming the facts that are used in each leaf proof makes the proof easier for a computer to check.