A resettable variable is defined as follows. There are three new names: \textit{value} (of type \( X \)), \textit{set} (a procedure with one parameter of type \( X \)), and \textit{reset} (a program). Here are the axioms:

\[
\begin{align*}
\text{value}' &= x \iff \text{set } x \\
\text{value}' &= \text{value} \iff \text{set } x. \text{ reset} \\
\text{reset. reset} &= \text{reset}
\end{align*}
\]

Implement this data structure, with proof.

\[
\begin{align*}
\text{Let } \text{value}: X \text{ be a user's variable, and let } \text{old}: X \text{ be an implementer's variable.} \\
\text{set} &= \langle x: X \mapsto \text{old} := \text{value} \cdot \text{value} := x \rangle \\
\text{reset} &= \text{value} := \text{old}\\n\text{Proof:} \\
&& (\text{value}' = x \iff \text{set } x) \\
&= (\text{value}' = x \iff \text{old} := \text{value} \cdot \text{value} := x) \\
&= (\text{value}' = x \iff \text{old}' = \text{value} \land \text{value}' = x) \\
&= \top \\
&= (\text{value}' = \text{value} \iff \text{set } x. \text{ reset}) \\
&= (\text{value}' = \text{value} \iff \text{old} := \text{value} \cdot \text{value} := x \cdot \text{value} := \text{old}) \\
&= (\text{value}' = \text{value} \iff \text{old}' = \text{value}' = \text{value}) \\
&= \top \\
&= (\text{reset. reset} = \text{reset}) \\
&= (\text{value} := \text{old} \cdot \text{value} := \text{old} = \text{value} := \text{old}) \\
&= \top
\end{align*}
\]