Prove the following definitions implement simple data-stack theory (Subsection 7.0.2).

\[
\begin{align*}
\text{stack} & = \{\text{nil}, [\text{stack}; X] \} \\
\text{push} & = \langle s: \text{stack}; \langle x: X; [s; x] \rangle \rangle \\
\text{pop} & = \langle s: \text{stack}; s \ 0 \rangle \\
\text{top} & = \langle s: \text{stack}; s \ 1 \rangle
\end{align*}
\]

After trying the question, scroll down to the solution.
Consider the implementation to be four axioms, named by their left sides. Now I prove each of the axioms of simple data-stack theory. First, \( \text{stack} \neq \text{null} \) by contradiction.

\[
\begin{align*}
\text{stack} &= \text{null} & \text{conjoin stack axiom} \\
\Rightarrow & \quad \text{stack} = \text{null} \land \text{stack} = \left[ \text{nil} \right], \left[ \text{stack}; X \right] & \text{context, then specialize} \\
\quad \Rightarrow & \quad \text{null} = \left[ \text{nil} \right], \left[ \text{null}; X \right] & \text{both } ; \text{ and } \left[ \right] \text{ distribute over }, \\
\quad \Rightarrow & \quad \text{null} = \left[ \text{nil} \right] & \text{null is identity for }, \\
\quad \Rightarrow & \quad \varepsilon \text{ null} = \varepsilon \left[ \text{nil} \right] & \text{size axioms; note that } \left[ \text{nil} \right] \text{ is an element} \\
\quad \Rightarrow & \quad 0 = 1 & \text{arithmetic axiom} \\
\Rightarrow & \quad \bot
\end{align*}
\]

Let \( s : \text{stack} \) and \( x : X \). Then

\[
\begin{align*}
\text{push} \ s \ x : \text{stack} & \quad \text{use push and stack axioms} \\
\Rightarrow & \quad \left( s : \text{stack} \right) \left( x : X \cdot \left[ s ; x \right] \right) s : X \cdot \left[ s ; x \right] & \text{apply} \\
\quad \Rightarrow & \quad \left[ s ; x \right] : \left[ \text{nil} \right], \left[ \text{stack}; X \right] & \text{generalization} \\
\quad \Rightarrow & \quad \text{top} \left( \text{push} \ s \ x \right) = x & \text{use top and push axioms} \\
\quad \Rightarrow & \quad \left( s : \text{stack} \cdot s \ 0 \right) \left( s : \text{stack} \cdot x : X \cdot \left[ s ; x \right] \right) s = s & \text{apply} \\
\quad \Rightarrow & \quad \left( s : \text{stack} \cdot s \ 0 \right) \left[ s ; x \right] = s & \text{apply} \\
\quad \Rightarrow & \quad \left[ s ; x \right] 0 = s & \text{index} \\
\quad \Rightarrow & \quad \text{top} \left( \text{push} \ s \ x \right) = x & \text{apply} \\
\quad \Rightarrow & \quad \left( s : \text{stack} \cdot s \ 1 \right) \left( s : \text{stack} \cdot x : X \cdot \left[ s ; x \right] \right) s = x & \text{apply} \\
\quad \Rightarrow & \quad \left( s : \text{stack} \cdot s \ 1 \right) \left[ s ; x \right] = x & \text{apply} \\
\quad \Rightarrow & \quad \left[ s ; x \right] 1 = x & \text{index} \\
\quad \Rightarrow & \quad \bot
\end{align*}
\]

The last step, indexing, requires \( x \) to be an item, so this implementation requires \( X \) to be a bunch of items.