A stack, according to our axioms, has an unlimited capacity to have items pushed onto it. A limited-stack is a similar data structure but with a limited capacity to have items pushed onto it.

(a) Design axioms for a limited-data-stack.
(b) Design axioms for a limited-program-stack.
(c) Can the limit be 0?

After trying the question, scroll down to the solution.
Design axioms for a limited-data-stack.

§ I suppose \( \text{limit} \) is a given natural, and \( X \) is a given bunch. I introduce the following new syntax: \( \text{stack} \), \( \text{push} \), \( \text{pop} \), \( \text{top} \), \( \text{empty} \), \( \text{size} \). Here are the axioms: Let \( s,t: \text{stack} \) and \( x,y:X \). Then

\[
\begin{align*}
\text{empty: stack} & \quad \text{size empty} = 0 \\
\text{size} & \quad s > 0 \implies \text{pop s: stack} \\
\text{size} & \quad s > 0 \implies \text{size (pop s)} = \text{size s} - 1 \\
\text{size} & \quad s < \text{limit} \implies \text{push s x: stack} \\
\text{size} & \quad s < \text{limit} \implies \text{size (push s x)} = \text{size s} + 1 \\
\text{size} & \quad s < \text{limit} \implies \text{push s x \neq empty} \\
\text{size} & \quad s < \text{limit} \implies (\text{push s x = push t y} \iff s=t \& x=y) \\
\text{size} & \quad s < \text{limit} \implies \text{pop (push s x)} = s \\
\text{size} & \quad s < \text{limit} \implies \text{top (push s x)} = x
\end{align*}
\]

Design axioms for a limited-program-stack.

§ I suppose \( \text{limit} \) is a given natural, and \( X \) is a given bunch. I introduce the following new syntax: \( \text{push} \), \( \text{pop} \), \( \text{top} \), \( \text{mkempty} \), \( \text{size} \). Here are the axioms: Let \( x:X \). Then

\[
\begin{align*}
\text{top} & \quad x \land \text{size' = size s = 1} \iff \text{size < limit} \land \text{push x} \\
\text{size' = size s = 0} & \iff \text{size > 0} \land \text{pop} \\
\text{size' = 0} & \iff \text{mkempty} \\
\text{ok} & \iff \text{size < limit} \land (\text{push x, pop})
\end{align*}
\]

Can the limit be \( 0 \)?

§ Sure. Why not? Then the empty stack is also full, and no operations are possible. But there's no logical problem.