Implement weak program-stack theory as follows: the implementer's variable is a list that grows and never shrinks. A popped item must be marked as garbage.

§

Except for the requirement that “A popped item must be marked as garbage.”, here is a wasteful but correct solution. Leave push and top alone, and redefine

\[
\text{pop} \quad \equiv \quad s := s + s[0;\ldots;s–1]
\]

Here is an efficient solution, again, ignoring that “A popped item must be marked as garbage.”. The implementer's variables are \( L : \[*X] \) and \( s : \text{nat} \).

\[
\text{push} = (x : X \mapsto \text{if } \#L=s \text{ then } L := L^+[x] \text{ else } L := s \mapsto x \mid L. \ s := s + 1 \ \text{fi})
\]

\[
\text{pop} \quad \equiv \quad s := s – 1
\]

\[
\text{top} = L(s–1)
\]

\[
\text{balance} \quad \equiv \quad s' = s \land \forall i : 0..s. \ L'i = L_i
\]

We needed to define balance but not to implement it. Now we need to prove the axioms. All proofs proceed by substituting the definitions into the axioms and then using list theory.