Functions on Strings
Since strings are defined recursively, most functions on strings are as well.

Length:
\[ \text{len}(\varepsilon) = 0; \]
\[ \text{len}(wa) = \text{len}(w) + 1 \text{ for } w \in \Sigma^*, a \in \Sigma \]

Reversal:
\[ \varepsilon^R = \varepsilon; \]
\[ (wa)^R = aw^R \text{ for } w \in \Sigma^*, a \in \Sigma \]

Concatenation
\[ x \cdot \varepsilon = x \text{ for all } x \in \Sigma^*; \]
\[ x \cdot (wa) = (x \cdot w)a \text{ for } w \in \Sigma^*, a \in \Sigma \]

Number of c’s in a string
\[ \#_c(\varepsilon) = 0 \]
\[ \#_c(wc) = \#_c(w) + 1 \text{ for } w \in \Sigma^*; \]
\[ \#_c(wa) = \#_c(w) \text{ for } w \in \Sigma^*, a \in \Sigma \setminus \{c\}. \]

Claim for all \( x, y \in \Sigma^* \) \( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \).

Define Let \( P(y) \) be “for all \( x \in \Sigma^* \) \( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \).”

We prove \( P(y) \) for all \( y \in \Sigma^* \) by structural induction.

Base Case: Let \( x \) be an arbitrary string, \( \text{len}(x \cdot \varepsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\varepsilon) \)

Let \( y \) be an arbitrary string not covered by the base case. By the exclusion rule, \( y = wa \) for a string \( w \) and character \( a \).

Inductive Hypothesis: Suppose \( P(w) \)

Inductive Step: Let \( x \) be an arbitrary string.
\[ \text{len}(xy) = \text{len}(xwa) = \text{len}(xw) + 1 \text{ (by definition of len)} \]
\[ = \text{len}(x) + \text{len}(w) + 1 \text{ (by IH)} \]
\[ = \text{len}(x) + \text{len}(wa) \text{ (by definition of len)} \]

Therefore, \( \text{len}(xy) = \text{len}(x) + \text{len}(y) \), as required.

We conclude that \( P(y) \) holds for all string \( y \) by the principle of induction. Unwrapping the definition of \( y \), we get \( \forall x \forall y \in \Sigma^* \text{len}(xy) = \text{len}(x) + \text{len}(y) \), as required.

\( \Sigma^*: \text{Basis: } \varepsilon \in \Sigma^* \).

Recursive: If \( w \in \Sigma^* \) and \( a \in \Sigma \) then \( wa \in \Sigma^* \)
Structural Induction Template

1. Define $P()$ Show that $P(x)$ holds for all $x \in S$. State your proof is by structural induction.

2. Base Case: Show $P(x)$
   [Do that for every base cases $x$ in $S$.]
   Let $y$ be an arbitrary element of $S$ not covered by the base cases. By the exclusion rule, $y = \langle \text{recursive rules} \rangle$

3. Inductive Hypothesis: Suppose $P(x)$
   [Do that for every $x$ listed as in $S$ in the recursive rules.]

4. Inductive Step: Show $P()$ holds for $y$.
   [You will need a separate case/step for every recursive rule.]

5. Therefore $P(x)$ holds for all $x \in S$ by the principle of induction.

Binary Trees

Basis: A single node is a rooted binary tree.

Recursive Step: If $T_1$ and $T_2$ are rooted binary trees with roots $r_1$ and $r_2$, then a tree rooted at a new node, with children $r_1, r_2$ is a binary tree.

- $size(\ ) = 1$
- $size(\begin{array}{c}T_1 \quad T_2 \end{array}) = size(T_1) + size(T_2) + 1$
- $height(\ ) = 0$
- $height(\begin{array}{c}T_1 \quad T_2 \end{array}) = 1 + \max(height(T_1), height(T_2))$