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

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

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

Base Case:

Inductive Hypothesis:

Inductive Step:

\[ \Sigma^*: \text{Basis}: \varepsilon \in \Sigma^* . \]

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

---

**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.

\[
\begin{align*}
\text{size}(\bullet) &= 1 \\
\text{size}(T_1) &= \text{size}(T_1) \\
\text{size}(T_1 \cdot T_2) &= \text{size}(T_1) + \text{size}(T_2) + 1 \\
\text{height}(\bullet) &= 0 \\
\text{height}(T_1) &= \text{height}(T_1) \\
\text{height}(T_1 \cdot T_2) &= 1 + \max(\text{height}(T_1), \text{height}(T_2))
\end{align*}
\]
Regular Expressions

Basis:
\( \varepsilon \) is a regular expression. The empty string itself matches the pattern (and nothing else does).
\( \emptyset \) is a regular expression. No strings match this pattern.
\( a \) is a regular expression, for any \( a \in \Sigma \) (i.e. any character). The character itself matching this pattern.

Recursive
If \( A, B \) are regular expressions then \( (A \cup B) \) is a regular expression matched by any string that matches \( A \) or that matches \( B \) [or both].
If \( A, B \) are regular expressions then \( AB \) is a regular expression matched by any string \( x \) such that \( x = yz \), \( y \) matches \( A \) and \( z \) matches \( B \).
If \( A \) is a regular expression, then \( A^* \) is a regular expression matched by any string that can be divided into 0 or more strings that match \( A \).

More Examples

\((0^*1^*)^*\)

\(0^*1^*\)

\((0 \cup 1)^* (00 \cup 11)^* (0 \cup 1)^*\)

\((00 \cup 11)^*\)