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

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

\(0^*1^*\)

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

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

You can also go the other way
Write a regular expression for “the set of all binary strings of odd length”

Write a regular expression for “the set of all binary strings with at most two ones”

Write a regular expression for “strings that don't contain 00”

Induction: Hats!

Define $P(n)$ to be “in every line of $n$ people with gold and purple hats, with a purple hat at one end and a gold hat at the other, there is a person with a purple hat next to someone with a gold hat”
We show $P(n)$ for all integers $n \geq 2$ by induction on $n$.
Base Case: $n = 2$
Inductive Hypothesis:
Inductive Step:

By the principle of induction, we have $P(n)$ for all $n \geq 2$