

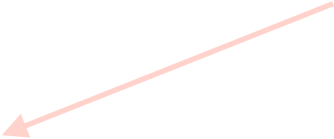
Regular Expression

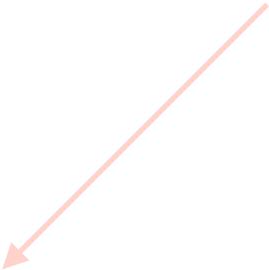
- Regular expression is a formalism that defines a regular language (= a set of symbols defined by regular expression)
- The production rules consists of terminal symbols, non-terminal symbols and operators (repetition { }, concatenation \sqcup , xor $|$, ...).
- A regular expression can be reduced to **a single rule** by replacing every non-terminal symbol with its right-hand side until no non-terminal symbols are left.

integer = digit {digit} .

**digit = "0" | "1" | "2" | "3" | "4" | "5" |
"6" | "7" | "8" | "9" .**

non-terminals





terminal symbol (literals of language)

EBNF -Niklaus Wirth

ebnf = { production } .

production = identifier "=" expression "." .

expression = term { "|" term } .

term = factor { factor } .

factor = identifier | string | "(" expression ")" |
 "[" expression "]" | "{" expression "}" .

string = "" printableCharacter { printableCharacter } "" .

printableCharacters = "a" | ... | "!" .

identifier = "ebnf" | "production" | ... | "identifier" .

Regular Expression

- Regular expression is a formalism that defines a regular language (= a set of symbols defined by regular expression)
- The production rules consists of terminal symbols, non-terminal symbols and operators (repetition { }, concatenation \sqcup , xor $|$, ...).
- A regular expression can be reduced to **a single rule** by replacing every non-terminal symbol with its right-hand side until no non-terminal symbols are left.

