REs:

A regular expression or RE R (over some alphabet Σ) is one of:

- ∅ (the null RE);
- ε (the empty RE);
- a (a literal, for any a ∈ Σ);
- R₁ | R₂ (the union of REs R₁ and R₂);
- R₁ · R₂ (the concatenation of REs R₁ and R₂); or
- R₁* (the Kleene star or iteration of RE R₁).

An RE R (over Σ) matches a string w = x₁…xₙ (where each xᵢ ∈ Σ) iff:

- R = ∅ and … well, uhh, then it just doesn’t match, ever: oh well!
- R = ε and n=0 (i.e., w is the empty string);
- R = a and n=1, with x₁ = a;
- R = R₁ | R₂ and either R₁ matches w or R₂ matches w;
- R = R₁ · R₂ and w can be split w = w₁w₂ so that both R₁ matches w₁ and R₂ matches w₂; or
- R = R₁* and either n=0 (i.e., w is the empty string) or w can be split w = w₁w₂ so that both R₁ matches w₁ and R₁* matches w₂.

DFAs:

A deterministic finite automaton or DFA M = ⟨Q, Σ, δ, q₀, F⟩ where

- Q is a finite set of states;
- Σ is an alphabet (finite set of symbols);
- δ ∈ Q × Σ → Q is called the transition function;
- q₀ ∈ Q is the initial state; and
- F ⊆ Q is the set of final states.

A DFA M accepts a string w = x₁…xₙ (where each xᵢ ∈ Σ) iff there are states q₁, …, qₙ ∈ Q such that:

- qᵢ₊₁ = δ(qᵢ, xᵢ₊₁) for every i, 0 ≤ i < n; and
- qₙ ∈ F.

The language of a DFA L(M) = { w ∈ Σ* | M accepts w }.

NFAs:

A non-deterministic finite automaton or NFA N = ⟨Q, Σ, δ, q₀, F⟩ where

- Q is a finite set of states;
- Σ is an alphabet (finite set of symbols);
- δ ∈ Q × Σ⁺ → Q is a transition function (where Σ⁺ = Σ ∪ {ε});
- q₀ ∈ Q is the initial state; and
- F ⊆ Q is the set of final states.

An NFA N accepts a string w = x₁…xₙ (where each xᵢ ∈ Σ) iff w can be written as w = y₁…yₘ where each yᵢ ∈ Σ⁺ and there are states q₁, …, qₘ ∈ Q such that:

- qᵢ₊₁ ∈ δ(qᵢ, xᵢ₊₁) for every i, 0 ≤ i < m; and
- qₚ ∈ F.

The language of an NFA L(N) = { w ∈ Σ* | N accepts w }.