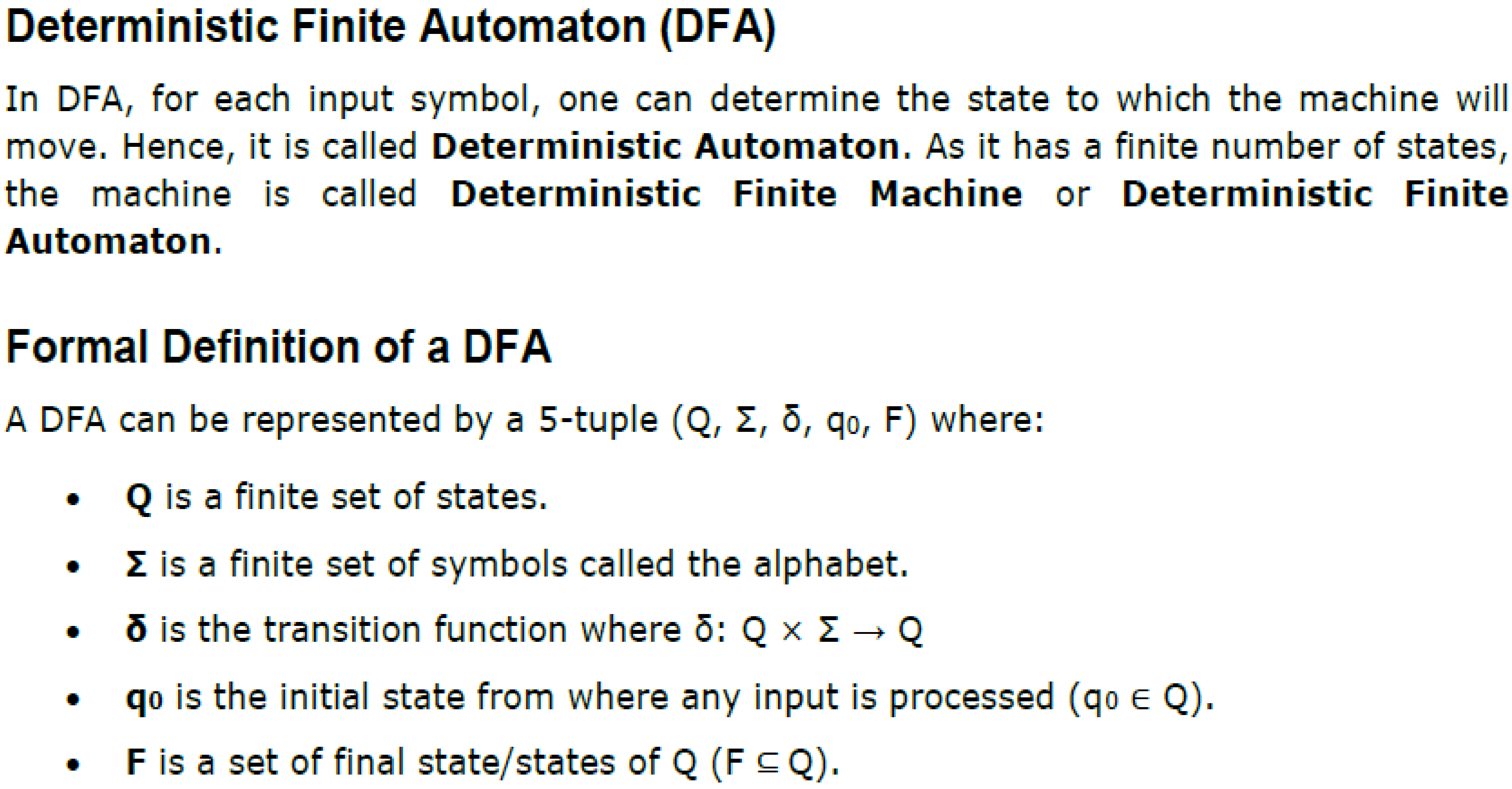
DFA to Code

Compiled By: Dr. Rabia Irfan for CS-354 (BSCS-5)

What is DFA?



Deterministic Finite Automaton (DFA)

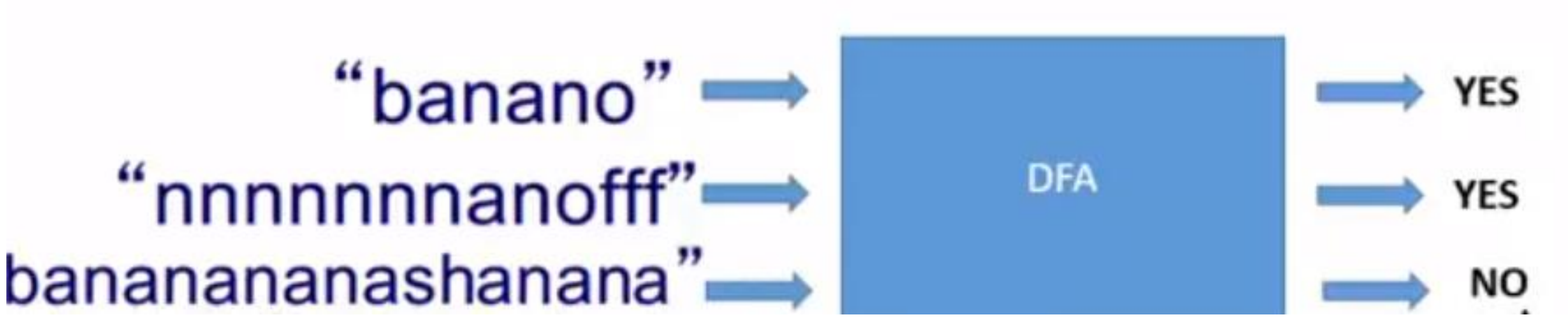
In DFA, for each input symbol, one can determine the state to which the machine will move. Hence, it is called **Deterministic Automaton.** As it has a finite number of states, the machine is called **Deterministic Finite Machine** or **Deterministic Finite Automaton.**

Formal Definition of a DFA

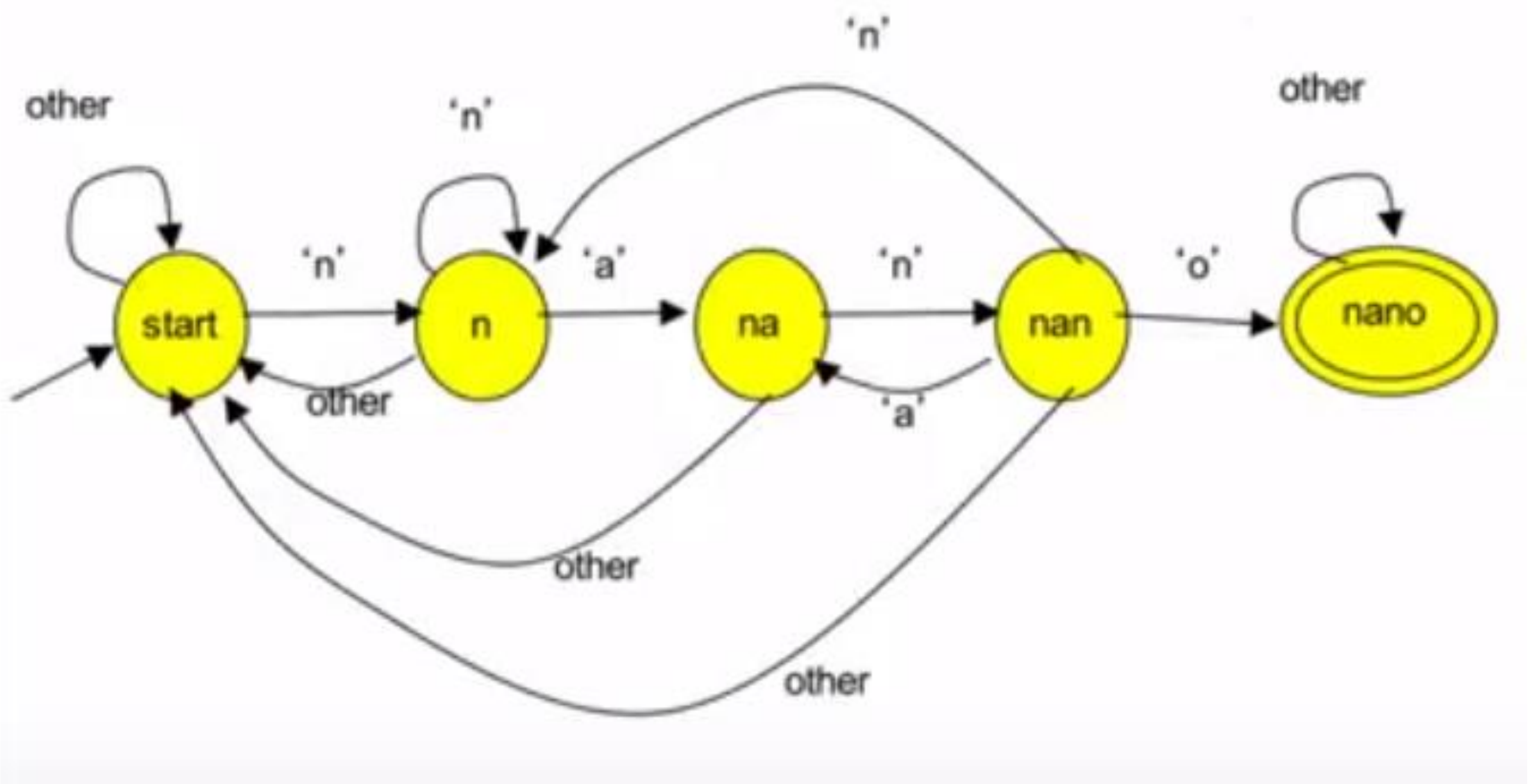
A DFA can be represented by a 5-tuple (Q, Σ, δ, q_0, F) where:

- **Q** is a finite set of states.
- **Σ** is a finite set of symbols called the alphabet.
- **δ** is the transition function where $\delta: Q \times \Sigma \rightarrow Q$
- **q_0** is the initial state from where any input is processed ($q_0 \in Q$).
- **F** is a set of final state/states of Q ($F \subseteq Q$).

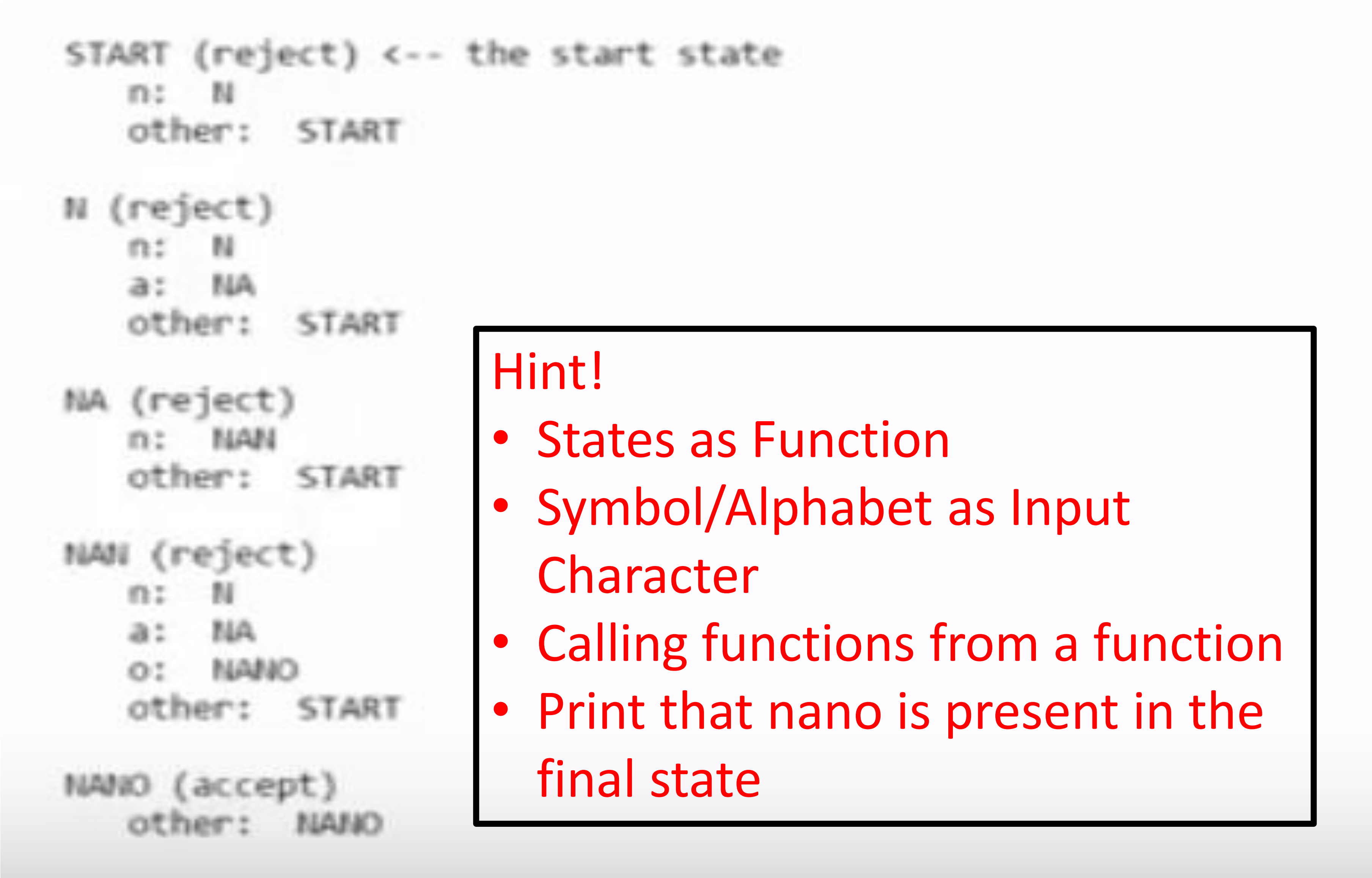
DFA that can recognize string pattern “nano”



DFA-State Diagram



DFA-Program Logic

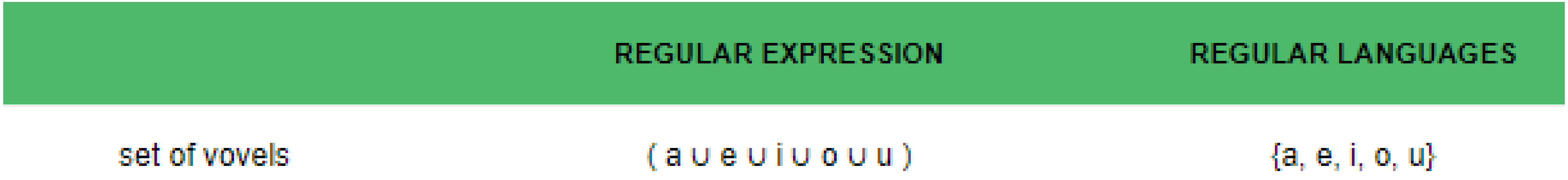


```
START (reject) <-- the start state
    n:   N
    other:   START

N (reject)
    n:   N
    a:   NA
    other:   START

NA (reject)
    n:   NAN
    other:   START

NAN (reject)
    n:   N
    a:   NA
    o:   NANO
    other:   START

NANO (accept)
    other:   NANO
```

Hint!

- States as Function
- Symbol/Alphabet as Input Character
- Calling functions from a function
- Print that nano is present in the final state

DFA-Regular Expression & Languages



	REGULAR EXPRESSION	REGULAR LANGUAGES
set of vovels	$(a \cup e \cup i \cup o \cup u)$	{a, e, i, o, u}

$\{w | w \in \{a, e, i, o, u\}^{*}\}$ ← Regular Language written formally