**AI-Assisted Interactive Flashcard Deck: Finite Automata**

**Flashcard 1: Definition of Finite Automaton**

**Q:** What is a Finite Automaton (FA)?  
**A:** A **Finite Automaton (FA)** is a mathematical model of computation used to recognize patterns and process strings. It consists of a finite set of states, a start state, a set of accept states, and transitions between states based on input symbols.

**Flashcard 2: Types of Finite Automata**

**Q:** What are the two main types of Finite Automata?  
**A:**

1. **Deterministic Finite Automaton (DFA)** – Each input has exactly one transition from a state.
2. **Nondeterministic Finite Automaton (NFA)** – An input can lead to multiple transitions or none.

**Flashcard 3: Components of a Finite Automaton**

**Q:** What are the key components of a Finite Automaton?  
**A:**

* **Q** → Finite set of states
* **Σ** → Finite set of input symbols (alphabet)
* **δ** → Transition function (Q × Σ → Q)
* **q₀** → Start state (initial state)
* **F** → Set of accept (final) states

**Flashcard 4: DFA vs NFA**

**Q:** How does a DFA differ from an NFA?  
**A:**

* **DFA:** Each state has exactly one transition per input symbol.
* **NFA:** A state can have multiple transitions for the same input or no transition at all.

**Flashcard 5: Regular Languages and FA**

**Q:** What type of languages can Finite Automata recognize?  
**A:** Finite Automata recognize **regular languages**, which can be expressed using regular expressions.

**Flashcard 6: Example of DFA**

**Q:** Design a DFA that accepts strings containing "ab" over {a, b}.  
**A:**

1. **Q = {q0, q1, q2}**, Σ = {a, b}, q₀ = q0, F = {q2}
2. **Transitions:**
   * q0 → a → q1
   * q1 → b → q2 (accept state)
   * q2 → a/b → q2 (loop in final state)

**Flashcard 7: NFA Example**

**Q:** Construct an NFA for strings ending in "01" over {0,1}.  
**A:**

1. **Q = {q0, q1, q2}**, Σ = {0,1}, q₀ = q0, F = {q2}
2. **Transitions:**
   * q0 → 0 → q0 | q1
   * q1 → 1 → q2 (accept state)
   * q2 → 0/1 → q2

**Flashcard 8: Conversion from NFA to DFA**

**Q:** How can an NFA be converted into a DFA?  
**A:** By applying the **subset construction method**, where each DFA state represents a set of NFA states.

**Flashcard 9: Minimization of DFA**

**Q:** What is DFA minimization?  
**A:** It is the process of reducing the number of states while maintaining the same language recognition using **equivalence classes** or **partitioning methods**.

**Flashcard 10: Applications of Finite Automata**

**Q:** Where are Finite Automata used?  
**A:**

* **Lexical analysis in compilers**
* **Text pattern matching (e.g., search engines, regex processing)**
* **Network protocol verification**
* **Natural language processing**