**Chapter 1: FINITE AUTOMATA**

**Topic – 1: Introduction**

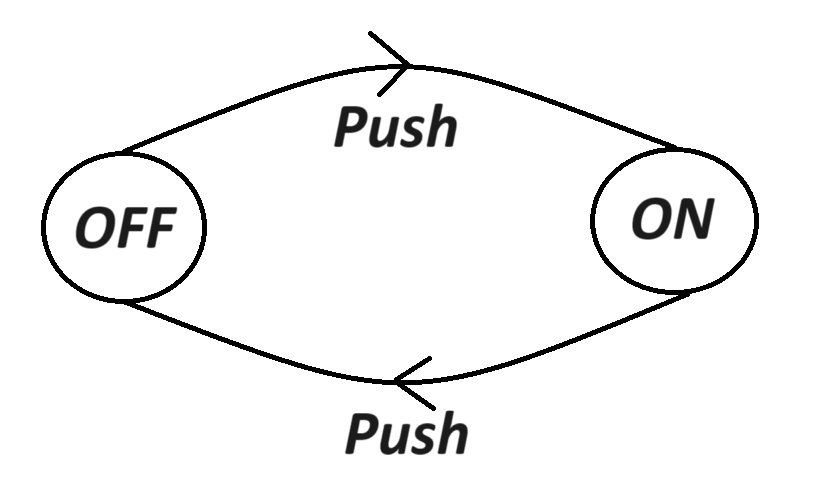
* **Computation:** Following an algorithm to solve a problem.
* Pen and paper can be called a **computational device** because they help in solving.

**Topic – 2: Finite Automata**

* A computational model.
* Has **finite** amount of **memory** i.e. finite number of states.
* **Automata** is plural of **automaton**.

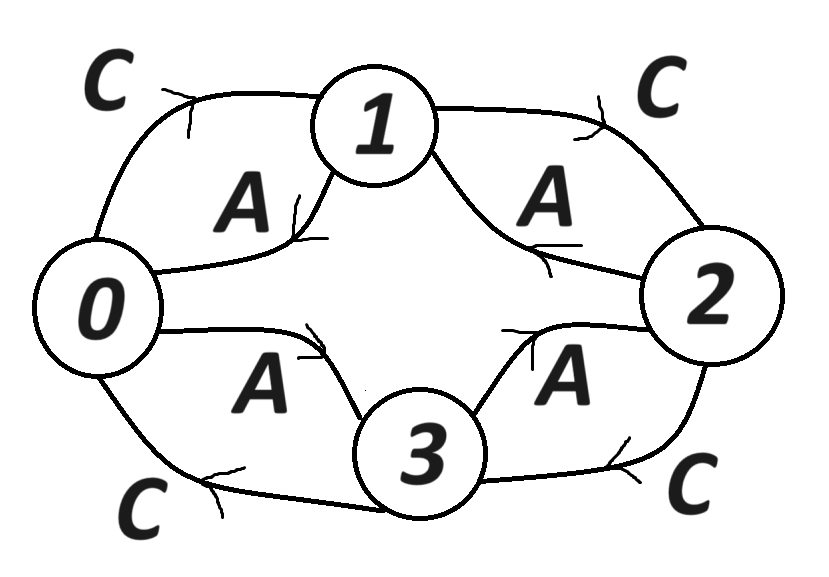
**Example 1**

* An **electric switch** which has finite number of possible states i.e. **ON** and **OFF**.
* **State** of an automata is changed using ***push*** operation.



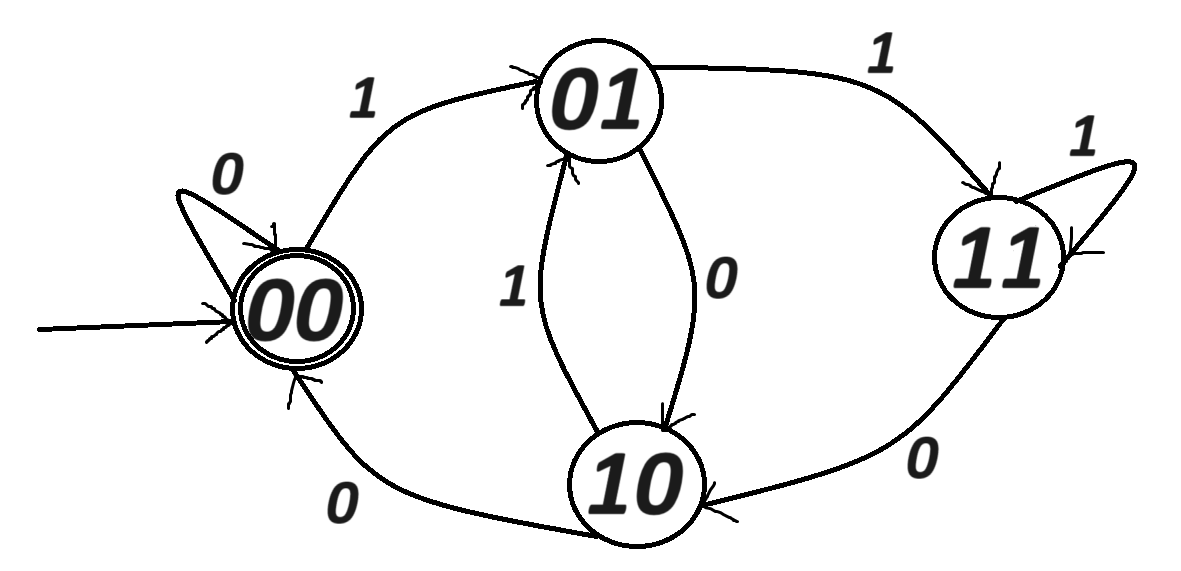
**Example 2**

* Another example can be a **fan regulator**, but it can go **both way** when being rotated.
* Operations on it can be named as **clockwise** & **anti-clockwise**.



**Example 3**

* Let’s say **L** is a set of all binary numbers **divisible by 4**.
* We notice that elements in this set, the binary numbers are tailed with **two zeroes**.
* The **arrow** shows the ***start state*** & the **double ring** shows the ***accept state***.



**Diagram Explanation**

**00** 🡪 **00|1** 🡪 **00|1 = 01**

**00** 🡪 **00|0** 🡪 **00|0 = 00**

**01** 🡪 **01|0** 🡪 **01|0 = 10**

**01** 🡪 **01|1** 🡪 **01|1 = 11**

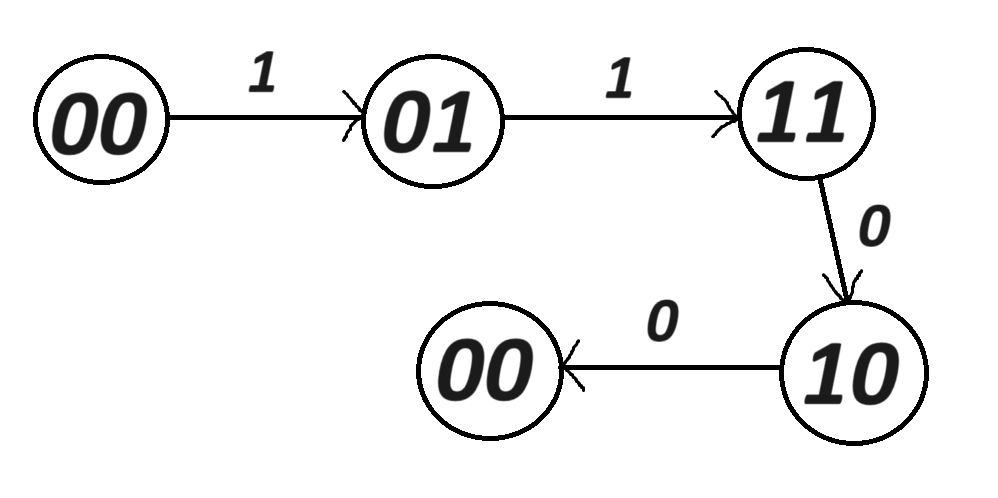
**10** 🡪 **10|0** 🡪 **10|0 = 00**

**10** 🡪 **10|1** 🡪 **10|1 = 01**

**11** 🡪 **11|0** 🡪 **11|0 = 10**

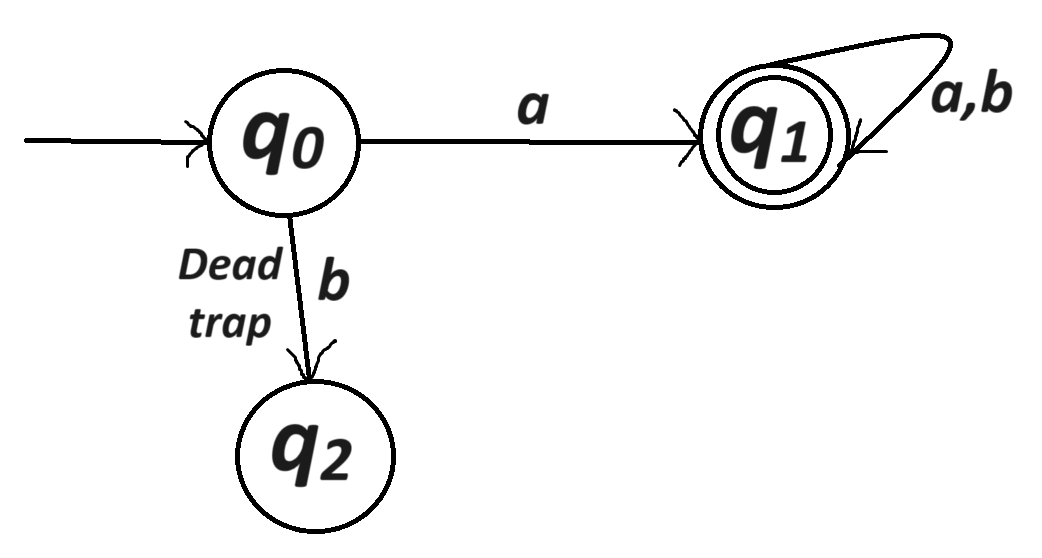
**11** 🡪 **11|1** 🡪 **11|1 = 11**

* This can be used to know if some binary number will stop at **accept state**.
* For example, for **1100**:



**Example 4**

**L = {w: {a,b}\* | string starts with 'a'}**



**Example 5**

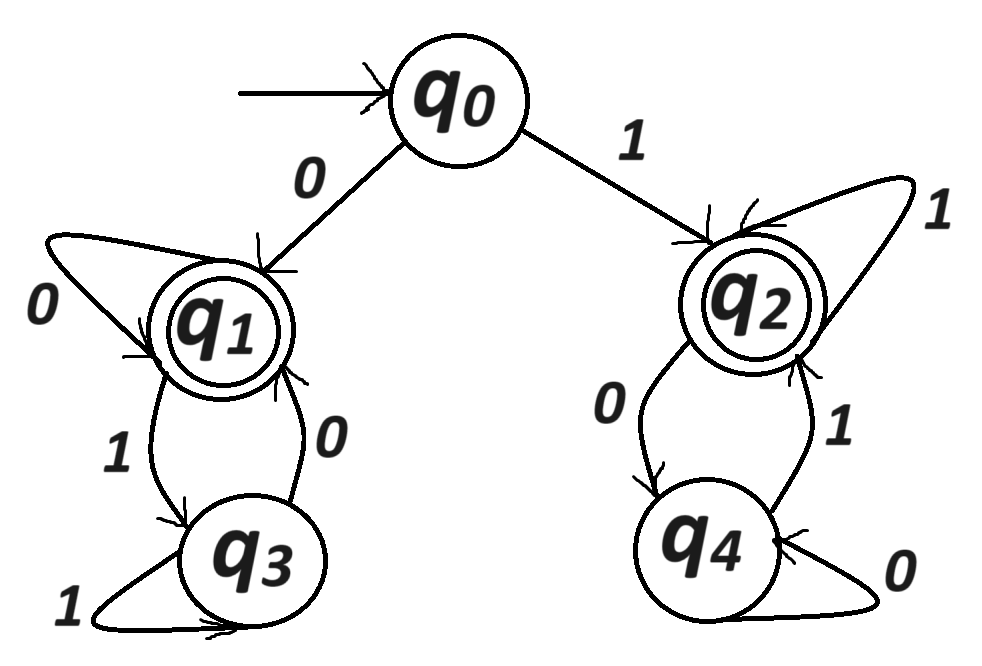
**L = {w: {0,1}\* | string is divisible by 3}**

* We will have 3 states:
  + **q0:** Remainder **0** when divided by 3
  + **q1:** Remainder **1** when divided by 3
  + **q2:** Remainder **2** when divided by 3

**Example 6**

**L = {w: {0,1}\* | string starts & ends with same symbol}**

* We will have 4 possible states:
  + **q1:** String starts with **0** & ends with **0**.
  + **q2:** String starts with **0** & ends with **1**.
  + **q3:** String starts with **1** & ends with **0**.
  + **q4:** String starts with **1** & ends with **1**.
  + **q0:** Empty string



**Topic – 3: Definitions & Notations**

* **Alphabet:** Finite set of symbols, denoted by **Σ**.
* **String:** Sequence of symbols from alphabets, denoted by **w**.
* **Length:** Number of symbols in a string, denoted by **|w|**.
* For example, **|01101| = 5**.
* **Empty string:** String consisting of **0** symbols, denoted by **Ԑ**.

**Σi = {w: w is a string over Σ and |w| = i}**

**Means w is a string with symbols from Σ & having length i.**

**Σ2 = {00,01,10,11}**

**Σ\* = All possible strings that can be made from symbols in Σ**

**Σ\* = Ui >= 0 Σi**

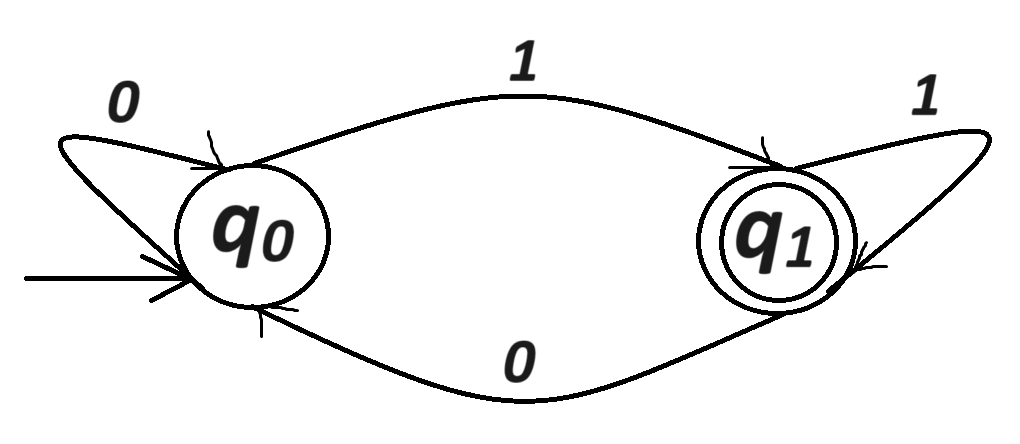
**Σ+ = Ui > 0 Σi**

* **Language:** Collection of strings from **Σ\***.

**Topic – 4: Deterministic Finite Automaton (DFA)**

* Set of 5 tuples **{Q, Σ, δ, q0, F}**.
* **Q** = Finite set of states
* **Σ** = Alphabet
* **δ** = Q x Σ 🡪 Q
* **q0** = Start state
* **F** = Set of accept states

**Description**



* We described **δ** as **Q x Σ 🡪 Q**
* Taking figure above as example, notice how **state q0** and **transition 1** cross over **q1**.

**Notations & Facts**

**M = {Q, Σ, δ, q0, F}**

* So, a **language L** belonging to **Σ\*** is accepted when strings in it have properties of **M**.
* These strings are denoted as **L(M)**.

**Note!**

🡪 For **a language**, there can exist **infinitely many DFA**.

🡪 But for a **DFA**, there can be only **one language**.

🡪 The number of **outgoing** & **incoming arrows** are equal to **number of string digits** in a language (like **0** and **1** in a **binary** language). Refer to **example 6**.

🡪 **DFA** is **"deterministic"** because model deterministically goes to a **unique state**.

🡪 Number of states are as per **possibility** of types of strings.

🡪 When proceeding to next state, we **don’t** consider the past states or past information.

🡪 Word **"finite"** in DFA due to limited number of strings.

**Topic – 5: Computation of DFA**

* **M** accepts **w** when there exists a series **r0, r1, r2, r3,…, r4**.

**Conditions To Compute DFA**

**r0 = q0 (initial condition)**

**ri = δ(ri-1, ai) (transition condition)**

**rn Є F (accept condition)**

* State **i** needs **not** to be distinct.
* **M** accepts **L** if:

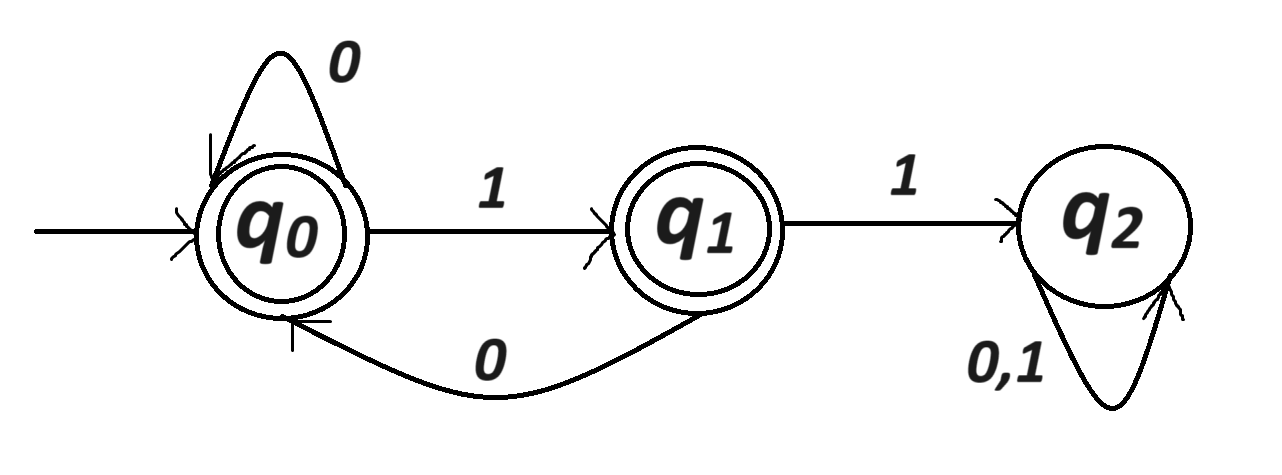
**L = {w Є Σ\* | M accepts w}**

**Topic – 6: Dump State**

* It is a state from which an automaton can **not** reach to an **accept state**.

**Example**

**L = {w | w doesn’t contain 11 as substring}**



* **q2** is dump state.

**Topic – 7: Regular Operations**

* **Unary operation:** Operation performed by a language on itself.
* **Binary operation:** Operation performed among two languages.
* For operations below, let **A**, **B** belong to **Σ\***.

**Union Operation**

**A U B = {x | x Є A or x Є B}**

**Concatenation Operation**

**AB = {xy | x Є A and y Є B}**

* **y** is appended to the end of **x**.
* For example:

**A = {a, b}**

**B = {1, 2, 3, …}**

**AB = {a1, b1, a2, b2,…}**

**Star Operation**

**A\* = {x1x2x3…xk | xi Є A for some i}**

* For example:

**A = {10, 001}**

**A\* = {Є, 10, 001, 1010, 10001, 00110, 001001, …}**

* It goes up to **infinity**.