Review:

- **strings or words** – finite sequences of symbols
- **alphabet** – finite set of symbols
- **languages** – sets of strings
- **operations** (on sets) – union, intersections, etc.
- **relations** – reflexive, symmetric, transitive, etc.

**NOTE:** if all 3 of these relations are satisfied it is an *equivalence relation*.

---

**Pigeon Hole Principle:**

If we have n items to be placed into m slots with n > m then no matter how we do it, there must be at least one slot that has one item.

**Functions:**

- are special cases of relations
- is a mapping from a set (domain) to another set (range), written as:

  ![Diagram](image)

  - ![Legal](image)
  - ![Illegal](image)

  - a bijection: one-to-one function (sometimes called “one-one function”)
    If $f(x_1) = f(x_2)$, then $x_1 = x_2$.

  - inverse function: a bijection can be inversed: $f^{-1}(y) = x$, if $f(x) = y$.

**Names about strings:**

- length: numbers of symbols in the string, denoted by $|x|$
- reverse: let $x = x_1, x_2, \ldots x_n$ $x_i \in \Sigma$ then $x^R = x_n, x_{n-1} \ldots x_2, x_1$
- palindrome: \(x = x^R\) eg. noon, racecar, radar, etc.

- prefix if \(x = y_1y_2\), then \(y_1\) is called the prefix of \(x\)

- suffix if \(x = y_1y_2\) then \(y_2\) is called the suffix of \(x\)

\[\begin{array}{c|c|c}
  y_1 & X & y_2 \\
\end{array}\]

- a very special string: a string that contains no symbols is called an empty string.
  - Denoted by the letter \(e\)
  - Is a string of length 0

---

**Chapter 2**

**2.1 Deterministic finite automata**

Let’s begin by considering two problems.

**Problem 1**: Design a program so that on any input string \(x\), if \(|x|\) is divisible by 4 then the program outputs 1, otherwise it outputs 0.

**Problem 2**: Design a program so that with any given array of numbers, it outputs these numbers in increasing order. (A sorting program such as quicksort, binary search, etc.)

What are the computational resources required for carrying out these tasks?

- Computation Time (time complexity)
  - The first problem can be solved with linear time.

  (Diagram showing how strings are read 4 symbols at a time.)

  - The second problem can be solved with \(\text{n/log(n)}\) time

- Memory
  - The first problem does not need any memory
The second problem needs memory for storing numbers. Hence, some problems do need auxiliary memory. These problems are the simplest kind of computational problems.

Q: How do we characterize these problems?
A: Using deterministic finite automata (DFA).

**Finite Control:**
- is a function that moves to a new state based on the current state and input symbol, which is called the transition function.
- It is modeled with the following diagram:

```
  head
q0   q1
  q2
  q3
```

A DFA, $M$, is a tuple $(Q, \Sigma, \delta, Q_0, F)$
- $Q$ is a finite set of states
- $\Sigma$ is an input alphabet
- $\delta$ is the transition function $\delta: Q \times \Sigma \rightarrow Q$
- $Q_0$ is the initial state
- $F$ is the subset of $Q$ used to indicate the set of favorable states (accepting states).

Problem 1 can be solved using a DFA as follows:

- $Q = \{q_0, q_1, q_2, q_3\}$
- $\Sigma = \{a_1, a_2, \ldots, a_k\}$
- $\delta = \delta_{A, i}$ ($i = 1, 2, \ldots, k$)

```
<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$A_i$ ($i = 1, 2, \ldots, k$)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>$q_0$</em></td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_0$</td>
</tr>
</tbody>
</table>
```

- $F = \{q_0\}$

Note: In this DFA for problem 1, $q_0$ is the favorable state because if nothing is left then the input is divisible by 4.

**Graph representation of DFAs:**
We are interested in directed graphs with labeled edges where each node represents a state, and an edge is labeled by a symbol. Example: if we have $\delta(q, a) = p$, then:

- double circles represent the accepting state
- the arrow represents the initial state.

Using graph representations of DFAs, Problem 1 can be described as follows:

Assuming that, $\Sigma = \{a, b\}$

Let $M = (Q, \Sigma, \delta, Q_0, F)$ be a DFA on input $x$, its computation path is a sequence of states $q_1, q_2 \ldots q_{i_1}$, such that $q_{i_1} = q_0$, where $x = x_1, x_2 \ldots x_{L-1}$ and $\delta(q_{i_j}, x_j) = q_{i_{j+1}}$.

If the computation path $x$ ends at an accepting state, then we say that $x$ is accepted by the DFA.

Let $L(m) = \{x \mid x \text{ is accepted by } M\}$, we call $L(m)$ the languages accepted by $M$ set of all strings on alphabet $a,b$ whose length is divisible by 4.