- Formalize these notions:
  \begin{align*}
  \text{Computational problem} & \equiv \text{Language} \\
  \text{Computer} & \equiv \text{Turing machine} \\
  \text{Algorithm to solve a computational problem} & \equiv \text{TM program that decides a language}
  \end{align*}

- Decidable language
  - Problem that has an algorithm
    - TM program halts on every input and gives correct YES/NO answer.

- There are undecidable languages!
  - Halting problem.
  - Hilbert's 10th problem.

- For decidable languages:
  - Running time \equiv \# \text{TM steps (as function of input size)}
- Complexity Theory.

- Classify languages according to their complexity in terms of running time, space, ....

- P, NP, NP-completeness

- .......

- NP-complete problems: Traveling Salesperson, 3SAT, MAX-CUT, SUBSET-SUM*, .......

---

Languages

Def. Alphabet is a finite set of symbols. E.g.

\[ \Sigma = \{0,1\} \]

\[ \Sigma = \{a,b,\ldots,z\} \]

\[ \Gamma = \{a-z, A-Z, \#, ?, -, \$\} \]
Def. String over an alphabet $\Sigma$ is a finite sequence of symbols from $\Sigma$.
- Length of a string = number of symbols.
  - E.g. $\Sigma = \{0, 1\}$.
  - Strings: 0, 1, 00, 11010, ...

Def. Empty string denoted by $\varepsilon$, has length zero.

Def. $\Sigma^*$ is set of all (finite length) strings.
  - E.g. $\Sigma = \{0, 1\}$.
  - $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \ldots \}$.

Def. A language $L$ is a subset of $\Sigma^*$.
$L \subseteq \Sigma^*$.

Examples
$\Sigma = \{0, 1\}$. $L_{\text{even}} =$ set of all even length strings.
$\Sigma = \{0, 1, \ldots, 9\}$ $L_{\text{prime}} =$ set of all strings that, in decimal, are prime integers.
\[ \Sigma = \{a, z, \#, \$, \&\} \] \quad L \text{ = set of all strings that are valid C programs}\\

**Language recognition problem**

**Def** For every language \( L \), the associated recognition problem is ("decision problem")

"Given string \( \alpha \in \Sigma^* \), is \( \alpha \in L \)?"

**Examples**

\( L_{\text{even}} \rightarrow \text{Given } \alpha, \text{ is } |\alpha| \text{ even?} \)

\( L_{\text{prime}} \rightarrow \text{Given } \alpha, \text{ does } \alpha \text{ represent a prime integer?} \)

\( L_{\text{programs}} \rightarrow \text{Given } \alpha, \text{ is } \alpha \text{ a valid C program (compiler!)?} \)

"Fact" "Every" computational problem (decision) can be cast as the recognition problem for an appropriate language."
Encoding ("objects" by strings).

**Example** Encoding Graphs.

\[ \Sigma = \{0, 1, \#\} \]

\[ G \] → Adj. matrix \[ \begin{array}{cccc}
  & 1 & 2 & 3 & 4 \\
 1 & 1 & 0 & 1 & 1 \\
 2 & 1 & 0 & 1 & 1 \\
 3 & 1 & 1 & 0 & 0 \\
 4 & 0 & 1 & 0 & 0 \\
\end{array} \] → 0110#1011#1100#0100

\[ \langle G \rangle \] Encoding as a string.

\[ L_{\text{graphs}} = \{ \langle G \rangle \mid G \text{ is a graph} \} \subseteq \{0, 1, \#\}^* \]

\[ L_{\text{conn}} = \{ \langle G \rangle \mid G \text{ is a connected graph} \} \subseteq L_{\text{graphs}} \]

\[ L_{\text{bipartite}} = \{ \langle G \rangle \mid G \text{ is a bipartite graph} \} \]

\[ L_{\text{Hamiltonian}} = \{ \langle G \rangle \mid G \text{ has a Hamiltonian cycle} \} \]
Hence the associated recognition problems are:

- Is G connected? } in P (= class of prob./lang. w/ polytime algo).
- Is G bipartite? } in P
- Is G Hamiltonian? } NP-complete.

**Encoding**

Similarly one can encode
- numbers, graphs, polynomials,
- (TM) programs! (e.g. a C-program is already a string)

and cast computational problems as language recognition problems!

**Computational model ("device", "computer")**

- "Weak" models.
  - Finite automata
  - Push-down automata
- "Definitive" model.
  - Turing machine. $\equiv$ "real" computer.
Turing Machine

- Alan Turing 1936.
- Model for "general purpose" computer.
- "TM can do everything that a real computer can do."

![Diagram of Turing Machine]

- A move / step:
  - Read symbol scanned by the head.
  - (Over)write, change state, move head Left or Right.

- Computation:
  - Start w/ initial configuration.
  - Make moves.
  - Accept / Reject after entering appropriate state.
Example: Deciding Palindromes.

\[ L = \{ w \mid w \in \{0,1\}^*, \ w = w^R \} \]

- "Remember" 1, Write \[ \boxed{1} \]
- Traverse right till you hit \[ \boxed{1} \], turn left.
- "Match" 1, Write \[ \boxed{1} \]
- Move left till you hit \[ \boxed{1} \], turn right.

- Accept if all symbols "matched".
- Reject if an "unmatch" is detected.

- Running time \( O(n^2) \) if \( n = |w| \).
Formal Definition of $TM$

A (finite) set of instructions of type:

$(\text{current state, symbol read}) \rightarrow$

$(\text{new state, symbol written, Left/Right})$

**Def.** A $TM$ $M$ is a 7-tuple

$$M = (\mathcal{Q}, \Sigma, \Gamma, \delta, q_{\text{start}}, q_{\text{accept}}, q_{\text{reject}})$$

where

- $\mathcal{Q}$ is a set of states.
- $\Sigma$ is input alphabet.
- $\Gamma$ is tape alphabet, $\Sigma \subseteq \Gamma$
- $\mu \in \Gamma \setminus \Sigma$.
- $q_{\text{start}}, q_{\text{accept}}, q_{\text{reject}} \in \mathcal{Q}$ are three special states.

- Transition function.

$$\delta : \mathcal{Q} \times \Gamma \rightarrow \mathcal{Q} \times \Gamma \times \{L,R\}.$$
Example \( L = \{ w \mid |w| \text{ is even} \} \), \( \Sigma = \{0\} \)

- \( \Sigma = \{0\} \) \( \Gamma = \{0, \text{\_} \} \).
- \( Q = \{ q_{\text{start}} = q_{\text{even}}, q_{\text{odd}}, q_{\text{accept}}, q_{\text{reject}} \} \)
- \( S(q_{\text{even}}, 0) = (q_{\text{odd}}, 0, R) \)
- \( S(q_{\text{odd}}, 0) = (q_{\text{even}}, 0, R) \)
- \( S(q_{\text{even}}, \text{\_}) = (q_{\text{accept}}, \text{\_}, R) \)
- \( S(q_{\text{odd}}, \text{\_}) = (q_{\text{reject}}, \text{\_}, R) \).

State diagram
A configuration of a TM consists of (contents of tape, state, head position).

\[
0 \ 1 \ # \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ ...
\]

\[ q_{17} \]

Initial configuration

Input

\[
\begin{array}{c|c|c|c|c|c|c|c}
& x_1 & x_2 & x_3 & \cdots & x_n & \# & \\\n\hline
q_{\text{start}} & & & & & & & \\
\end{array}
\]

Input \( x = x_1 x_2 \cdots x_n \in \Sigma^* \)

Computation

- Start in the initial config.
- Machine keeps making moves until the state reaches \( q_{\text{accept}} \) or \( q_{\text{reject}} \).
- If so, machine stops and accepts or rejects the input respectively.

\( x \notin L \)