Books

1. *Introduction to Automata Theory, Languages, and Computation*
   Hopcroft, Ullman

2. *Elements of the Theory of Computation*
   Lewis, Papadimitriou

3. *Theory of Computation*
   Sipser

Weightage:
Assignments / quizzes – 20%
Mid-sem exam – 30%
End-sem exam – 50%
Objectives

- Build formal models of problems + algorithms / programs (strings, languages, automata)
- Reason about what can / cannot be solved using a computer ((un)computability / (un)decidability)
- Reason about what can be solved easily (complexity)
Motivation

- Regular expressions — very widely used
e.g., shell, editors / word processors, text processing in general
- Finite state machines — frequently used to model problems
- Compiler construction (lexical analysis, parsing)
- Structured / semi-structured information (HTML, XML)
Terminology

**Alphabet** \((\Sigma)\) : *finite*, non-empty set

**Symbol** : each element of \(\Sigma\)

**String / word** : (finite) sequence of symbols from \(\Sigma\)

**Language** : set of strings (possibly *infinite*) over a given \(\Sigma\)
**Problem** $\equiv$ language

Examples:
- Does a given string contain the string *auto*?
Formalisms

**Problem** $\equiv$ language

Examples:

- Does a given string contain the string *auto*?

\[
\Sigma = \{A, \ldots, Z, a, \ldots, z\}
\]

\[
L = \{w \mid w \text{ contains the string } auto\}
\]
Formalisms

\textbf{Problem} \equiv \text{language}

Examples:

- Does a given string contain the string \textit{auto}?

  \[ \Sigma = \{A, \ldots, Z, a, \ldots, z\} \]
  
  \[ L = \{w \mid w \text{ contains the string } \textit{auto}\} \]

- Is a given number divisible by 2?

  \[ \Sigma = \{0, 1, \ldots, 9\} \]
  
  \[ L = \{n \mid n \text{ is divisible by 2}\} \]
Problem $\equiv$ function $f$ from strings to strings

Example: Sort a given list of $n$ integers (appropriately encoded)
**Problem** ≡ function $f$ from strings to strings

Example: Sort a given list of $n$ integers (*appropriately encoded*)

$$L = \{w#v \mid v = f(w)\}$$
Program / algorithm \(\equiv\) (increasingly complex) automata

- Finite automata / regular expressions
- Pushdown automata / context-free grammars
- Turing machines / general grammars / \(\mu\)-recursive functions