

Lecture 0 – Course Overview

COSE215: Theory of Computation

Jihyeok Park



2025 Spring

- **Instructor:** Jihyeok Park (박지혁)
 - **Position:** Assistant Professor in CS, Korea University
 - **Expertise:** Programming Languages, Software Analysis
 - **Office hours:** 14:00–16:00, Tuesdays (appointment by e-mail)
 - **Office:** 609A, Science Library Bldg
 - **Email:** jihyeok_park@korea.ac.kr
- **Class:** COSE215 - 02 (English)
- **Lectures:** 13:30–14:45, Mondays and Wednesdays @ 301, Aegineung (애기능생활관 301호)
- **Homepage:** <https://plrg.korea.ac.kr/courses/cose215/>
- **Teaching Assistant:**
 - Jungyeom Kim (김준겸) – kimjg1119@korea.ac.kr
 - Seongmin Ko (고성민) – 2018320221@korea.ac.kr
 - Hyunjoon Kim (김현준) – rickykhj@korea.ac.kr

- **6 Homework Assignments: 30%**

- Programming assignments in Scala (submission in [LMS](#))
- You can utilize or refer to any other materials (e.g., ChatGPT), but you **MUST** write your **OWN** solution.
- **Cheating is strictly prohibited. Cheating will get you an F.**

- **Midterm exam: 30%**

- April 23 (Wed.) 13:30 – 14:45 (in class, 75 min.)

- **Final exam: 30%**

- June 18 (Wed.) 13:30 – 14:45 (in class, 75 min.)

- **Attendance: 10%**

- Please use [LMS](#) to attend the class with the code provided.
- Today's attendance check is a test run.

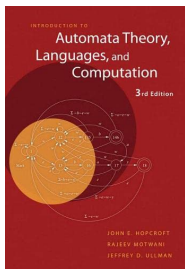
Week	Contents	Week	Contents
1	Basic Concepts	9	Pushdown Automata
2	Deterministic Finite Automata (DFA)	10	Deterministic Pushdown Automata
3	Nondeterministic Finite Automata (NFA)	11	Properties of Context-Free Languages
4	Regular Expressions and Languages	12	Turing Machines (TMs)
5	Properties of Regular Languages	13	Extensions of Turing Machines
6	Context-Free Grammars and Languages	14	Undecidability
7	Parse Trees and Ambiguity	15	P, NP, and NP-Completeness
8	Midterm Exam (Apr. 23 - Wed.)	16	Final Exam (Jun. 18 - Wed.)

- There will be no offline lectures on May 5 (Children's Day).
- Instead, a recorded lecture video will be uploaded to [LMS](#).
- You don't need to check the attendance on May 5.

- **Self-contained lecture notes.**

<https://plrg.korea.ac.kr/courses/cose215/>

- Reference:



John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman. Introduction to automata theory, languages, and computation. Third edition.

- What is the *mathematical model* of computers?

Turing Machine!

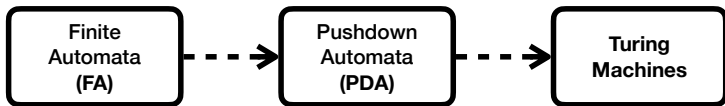
Let's learn **Turing Machine**

- Is it possible to solve *every problem* using computers?

No!

Let's learn **Undecidability** and **Intractability**

A Turing machine is a specific kind of **automaton**.



- **Part 1: Finite Automata (FA)**

- Regular Expressions (REs)
- Regular Languages (RLs)
- Applications: text search, etc.

- **Part 2: Pushdown Automata (PDA)**

- Context-Free Grammars (CFGs)
- Context-Free Languages (CFLs)
- Applications: programming languages, natural language processing, etc.

- **Part 3: Turing Machines (TMs)**

- Lambda Calculus (LC)
- Recursively Enumerable Languages (REs)
- Undecidability and Intractability

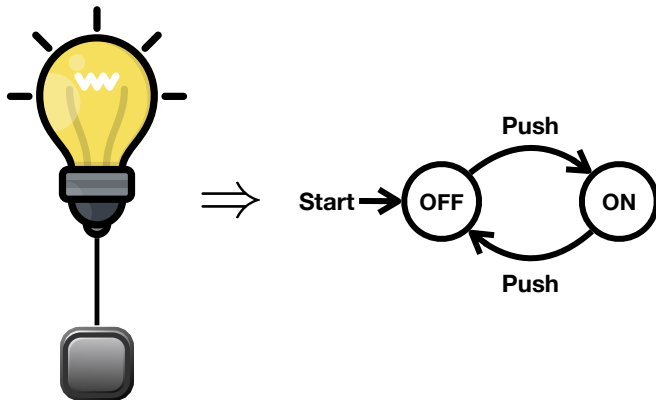
Roadmap: Towards Turing Machine

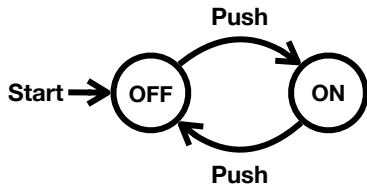
	Automata	Grammars	Languages
(Part 3) Turing Machines	(Lecture 23) ETM \longleftrightarrow (Lecture 21/22) TM \longleftrightarrow (Lecture 24) LC		(Lecture 21) REL \cup DL \supset NP $\stackrel{?}{=} P$ (Lecture 26) (Lecture 25)
(Part 2) Pushdown Automata	(Lecture 14/15) $PDA_{FS} \longleftrightarrow PDA_{ES}$ \cup $DPDA_{FS} \supset DPDA_{ES}$ \cup (Lecture 17) \nsubseteq	(Lecture 16) \longleftrightarrow (Lecture 11/12) CFG Chomsky Normal Form (Lecture 18)	(Lecture 11) CFL \dots (Lecture 13) Parse Trees & Ambiguity Closure Properties (Lecture 19) \dots Pumping Lemma (Lecture 20)
(Part 1) Finite Automata	(Lecture 4) NFA \longleftrightarrow (Lecture 3) DFA \longleftrightarrow (Lecture 5) ϵ -NFA \longleftrightarrow (Lecture 7) RE Equivalence & Minimization (Lecture 10)	(Lecture 6)	(Lecture 3) RL \dots Closure Properties (Lecture 8) \dots Pumping Lemma (Lecture 9)
(Part 0) Basic Concepts	(Lecture 1) Mathematical Preliminaries	(Lecture 2) Scala	

A Turing machine is a specific kind of **automaton**.

Then, what is an **automaton**? A **state transition system** that takes an **input** and changes its **state** based on the input.

For example,





Theorem

The current state is OFF if and only if the button is pushed even times.

- Is it possible to prove it?

Let's learn **mathematical background and notation**.

- Is it possible to implement the automaton?

Let's learn **Scala** as an implementation language.

- Mathematical Preliminaries

Jihyeok Park

jihyeok_park@korea.ac.kr

<https://plrg.korea.ac.kr>