Lecture 0 – Course Overview

AAA551: Programming Language Theory

Jihyeok Park

PLRG

2026 Spring

Course Information

- **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:** 507, Jung Woonoh IT & General Education Center
  - **Email:** jihyeok_park@korea.ac.kr

Course Information

- **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:** 507, Jung Woonoh IT & General Education Center
  - **Email:** jihyeok_park@korea.ac.kr

- **Class:** AAA551: Programming Language Theory

Course Information

- **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:** 507, Jung Woonoh IT & General Education Center
  - **Email:** jihyeok_park@korea.ac.kr

- **Class:** AAA551: Programming Language Theory

- **Homepage:** https://plrg.korea.ac.kr/courses/aaa551/

Course Information

- **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:** 507, Jung Woonoh IT & General Education Center
  - **Email:** jihyeok_park@korea.ac.kr

- **Class:** AAA551: Programming Language Theory

- **Homepage:** https://plrg.korea.ac.kr/courses/aaa551/

- **LMS:** https://lms.korea.ac.kr/

Course Information

- **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:** 507, Jung Woonoh IT & General Education Center
  - **Email:** jihyeok_park@korea.ac.kr

- **Class:** AAA551: Programming Language Theory

- **Homepage:** https://plrg.korea.ac.kr/courses/aaa551/

- **LMS:** https://lms.korea.ac.kr/

- **Prerequisites:** Basic knowledge of programming languages and type systems (e.g., from COSE212 or equivalent courses) is expected.

Grading and Schedule

- **2 Assignments: 80%**
  - Programming assignments in Scala (submission in **LMS**)

Grading and Schedule

- **2 Assignments: 80%**
  - Programming assignments in Scala (submission in **LMS**)
- **Attendance: 20%**
  - Please use **LMS** to attend the class with the code provided.
  - No attendance check for the first week, midterm, and final week.

Grading and Schedule

- **2 Assignments: 80%**
  - Programming assignments in Scala (submission in **LMS**)
- **Attendance: 20%**
  - Please use **LMS** to attend the class with the code provided.
  - No attendance check for the first week, midterm, and final week.
- There will be no offline lecture on the following day. Instead, recorded lecture videos will be uploaded to **LMS**.
  - May 5 (Tue.) – National Holiday (어린이날)

Course Materials

- **Self-contained lecture notes.**

https://plrg.korea.ac.kr/courses/aaa551/

Course Materials

- **Self-contained lecture notes.**

  https://plrg.korea.ac.kr/courses/aaa551/

- **Reference**: You can refer to the following textbooks for more details on the topics covered in this course.
  - "Types and Programming Languages" by Benjamin C. Pierce.
  - "Advanced Topics in Types and Programming Languages" by Benjamin C. Pierce.
  - "Introduction to Static Analysis: An Abstract Interpretation Perspective" by Xavier Rival and Kwangkeun Yi.

Motivation: The Gap

- JavaScript
  - `[] + []` produces

Motivation: The Gap

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces

Motivation: The Gap

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces `"[object Object]"`
  - `[] - []` produces

Motivation: The Gap

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces `"[object Object]"`
  - `[] - []` produces 0
  - `[] - {}` produces

Motivation: The Gap

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces `"[object Object]"`
  - `[] - []` produces `0`
  - `[] - {}` produces `NaN`

Motivation: The Gap

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces `"[object Object]"`
  - `[] - []` produces `0`
  - `[] - {}` produces `NaN`

- Python

```
funcs = []

for k in range(3):
    funcs.append(lambda x: x + k)

print(funcs[0](10))
print(funcs[1](10))
print(funcs[2](10))
```

Motivation: The Gap

PLRG

- JavaScript
  - `[] + []` produces `""`
  - `[] + {}` produces `"[object Object]"`
  - `[] - []` produces `0`
  - `[] - {}` produces `NaN`

- Python

```
funcs = []

for k in range(3):
    funcs.append(lambda x: x + k)

print(funcs[0](10))
print(funcs[1](10))
print(funcs[2](10))
```

It prints 12 three times, not 10, 11, and 12.

Motivation: The Gap

- Scala

```
trait Parent:
  val x: Int
  val y = x * 2

object Child extends Parent:
  val x = 10

println(Child.y)
```

Motivation: The Gap

- Scala

```
trait Parent:
  val x: Int
  val y = x * 2

object Child extends Parent:
  val x = 10

println(Child.y)
```

It prints 0, not 20.

Motivation: The Gap

- Scala

```
trait Parent:
  val x: Int
  val y = x * 2

object Child extends Parent:
  val x = 10

println(Child.y)
```

It prints 0, not 20.

- We need to replace fuzzy intuition with **mathematical precision**.

Motivation: The Gap

- Scala

```
trait Parent:
  val x: Int
  val y = x * 2

object Child extends Parent:
  val x = 10

println(Child.y)
```

It prints 0, not 20.

- We need to replace fuzzy intuition with **mathematical precision**.

- To bridge the gap between *“what we think code does”* and *“what it actually does”*, we need **formal semantics**.

Motivation: Defining "Meaning"

What is Semantics?

- Semantics is simply a fancy way of saying "Meaning".
- Goal: To give a precise meaning to programs.

Motivation: Defining “Meaning”

What is Semantics?

- Semantics is simply a fancy way of saying “Meaning”.
- Goal: To give a precise meaning to programs.

Why is this useful?

- For Programmers:
  - To clearly understand what our code is going to do before we run it.
- For Tool Builders (Compilers, Optimizers):
  - How do we know an optimization is safe?
  - Semantics allows us to judge whether these tools are implemented correctly.

Motivation: Programs as Information

A Deeper Question: "What is Information?"

- Standard Information:
  - Familiar mathematical tools like sets and sequences are adequate for static data.

Motivation: Programs as Information

A Deeper Question: "What is Information?"

- Standard Information:
  - Familiar mathematical tools like sets and sequences are adequate for static data.
- Programs as Information:
  - Programs describe computation, but they are also a kind of information.
  - They are a particularly interesting, dynamic kind of information.

Motivation: Programs as Information

A Deeper Question: "What is Information?"

- Standard Information:
  - Familiar mathematical tools like sets and sequences are adequate for static data.
- Programs as Information:
  - Programs describe computation, but they are also a kind of information.
  - They are a particularly interesting, dynamic kind of information.
- The Challenge:
  - Standard math is not enough.
  - We need specialized formal tools to precisely describe this unique type of information.

Course Structure: Two Pillars

We view Programming Languages through two complementary lenses:

- **Semantics (The Meaning)**
  - **Operational:** *How* is a program executed?
  - **Denotational:** *What* is the mathematical object for a program?
  - **Axiomatic:** *Which properties* does a program satisfy?
- **Type Systems (The Guardrails)**
  - **Modern Types:** Univeral, Substructural, Effects
  - **Type Analysis:** Abstract Interpretation
  - **Propositions as Types:** Curry-Howard Correspondence

Next Lecture

- Basic Introduction to Scala

Jihyeok Park
jihyeok_park@korea.ac.kr
https://plrg.korea.ac.kr