# Formal Languages and Compilers - Project Report

Erick Guerrero, David Ortiz, Anyela Jimenez

May 17, 2025

## Introduction

This report details the problems encountered and solutions implemented during the development of an LL(1) and SLR(1) parser for the Formal Languages and Compilers course project. The project involved designing and implementing algorithms to compute First and Follow sets, and to construct LL(1) and SLR(1) parsing tables, based on the specifications provided in Aho et al., Compilers: Principles, Techniques, and Tools (2nd Edition).

## SLR(1) Parser Challenges and Solutions

We encountered several challenges during the implementation of the SLR(1) parser, primarily concerning the accuracy of parsing actions, the GOTO table, and the handling of the epsilon symbol.

### Incorrect Reduce Sequences and GOTO Table

Initially, the SLR(1) parser produced incorrect reduce sequences and generated an inaccurate GOTO table.

#### Solution

To address this, we revised the 'slr$_parse$' $function to correctly determine reduce actions based on FOLLOW sets. We also corre$

### Epsilon as a Terminal in Tables

The epsilon symbol ('$\epsilon$') was incorrectly appearing as a terminal symbol within the ACTION and GOTO tables.

#### Solution

We implemented an exclusion rule to prevent epsilon from being treated as a terminal in the tables. This involved filtering out epsilon during table population.

## Grammar Input Handling Issues

We faced difficulties related to inputting the grammar through the console.

### Input Errors and Validation

It was challenging to input the grammar correctly via the console. Errors such as incorrect string input or extra tabulations often occurred. The program sometimes accepted these tabulations as empty strings.

### Solution

We implemented input cleaning using the 'strip()' method to remove leading/trailing whitespace. However, due to time constraints, we could not implement comprehensive input validation or interactive editing features.

## Conclusion

We successfully addressed key challenges in implementing the LL(1) and SLR(1) parsers, leading to a more functional tool. Further improvements are needed in SLR(1) conflict resolution and grammar input handling.