Department of Computer Engineering

VISHWAKARMA INSTITUTE OF TECHNOLOGY , PUNE

SyntaxBlend: Crafted Grammar and Redefining Syntax for a Language

- Presented by :- CSD_Group 84
- Guided by :- Prof. Anant Kulage

- Arpit Vidhale[69]
  12111229
- Aditya Warke[78]
  12110830
- Shivani Vikhar[70]
  12110338
- Sanket Thite[62]
  12110732

Table of Content

PROBLEM STATEMENT

In the dynamic realm of software development, creating a custom programming language is a challenging yet essential endeavor. The lack of a dedicated compiler for such custom languages poses a substantial obstacle, hindering their adoption and practical utilization. Addressing the issue there is a need to develop a compiler that bridge this gap and unlock the full potential of specialized languages.

OBJECTIVES

- To attain a holistic understanding of compiler design phase by constructing a language from scratch.
- To investigate and understand the trade-offs involved in language design choices
- To apply design principles practically to craft a programming language.

Name of Research Paper	Authors	Publish year	Key points to look out for:
A Compiler-based Approach for Natural Language to Code Conversion	Sashank Sridhar, Sowmya Sanagavarapu	2020	The paper discusses various existing models for employing NLP techniques in code generation. These models include techniques like semantic rule-based mapping, LSTM models, Java libraries, Abstract Syntax Trees (AST), and End User Programming (EuP) models.
The Lexical and Syntactic Analyzers of the Translator for the El Language	Aleksandr A. Maliavko	2018	This paper discusses the Eclipse Java Compiler (ECJ) and its static analysis capabilities. ECJ is a widely-used compiler that performs semantic analysis to support various features in the Eclipse IDE.
Interpretable sentiment analysis based on sentiment words' syntax information	Qingqing Zhao, Huaping Zhang, Jianyun Shang	2022	This paper introduces an interpretable sentiment analysis approach leveraging sentiment words' syntax information. The method focuses on understanding the sentiment expressed in text by analyzing the syntactic structure of sentiment-bearing words.
Lexical and Syntax Analysis in Compiler Design	Vishal Trivedi	2018	This paper provides a comprehensive exploration of the processes involved in evaluating source code, specifically focusing on Lexical and Syntax Analysis. It addresses the common practice of computers displaying only output and errors during code execution, without revealing the underlying evaluation steps.

PLAN OF ACTION

Define the Language Grammar

Specify syntax and semantics of the programming language.

Lexical Analysis (Tokenization)

Divide the source code into various tokens for further processing

Syntax Analysis (Parsing)

Verify the source code's syntactical correctness and create an AST

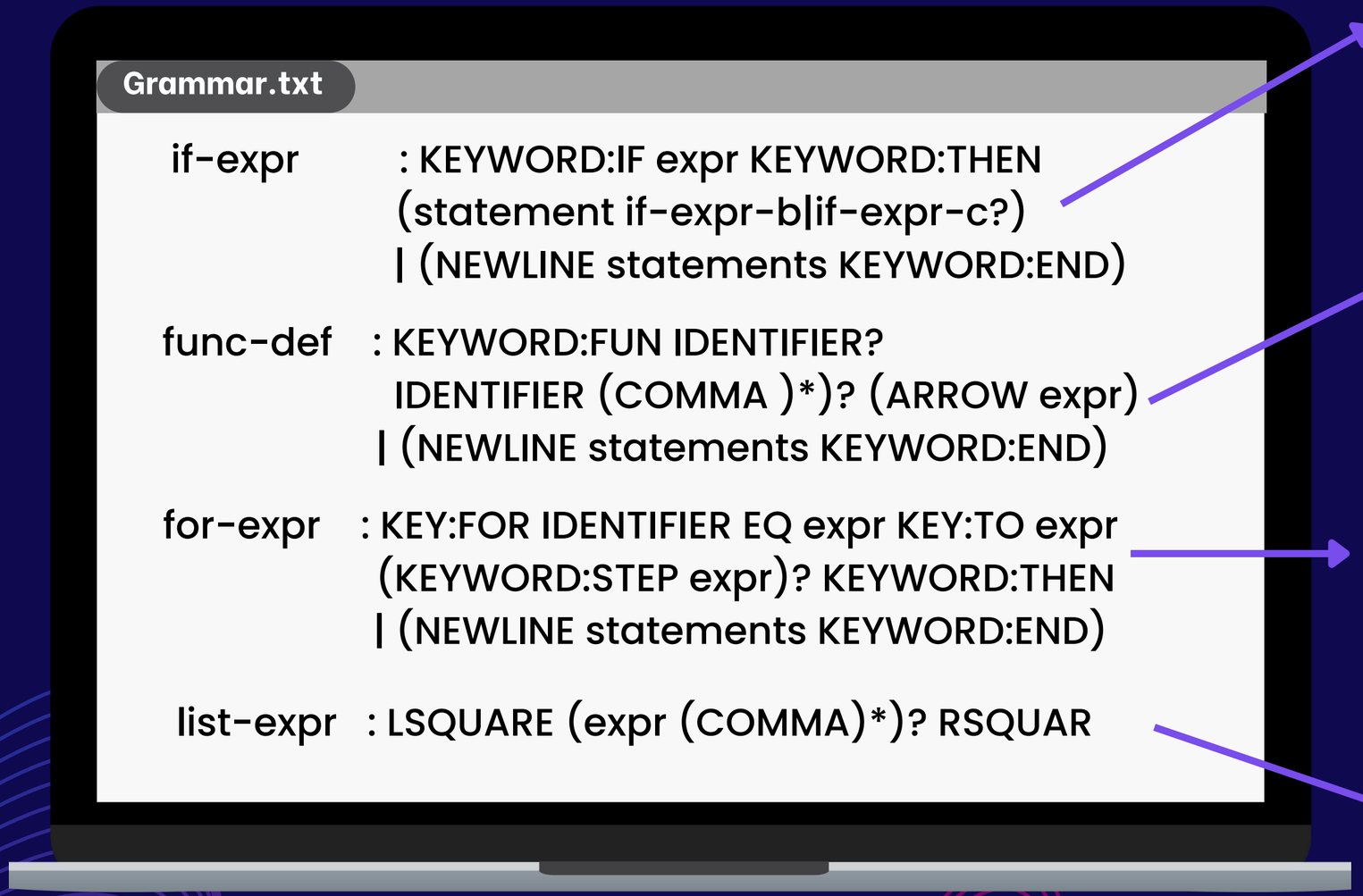
Semantic Analysis

Ensure the code follows language-specific rules and check for semantic errors.

Basic Interperter for defined grammar

Execution of code with defined variables, iteration, and condition cluase

Grammar for saas



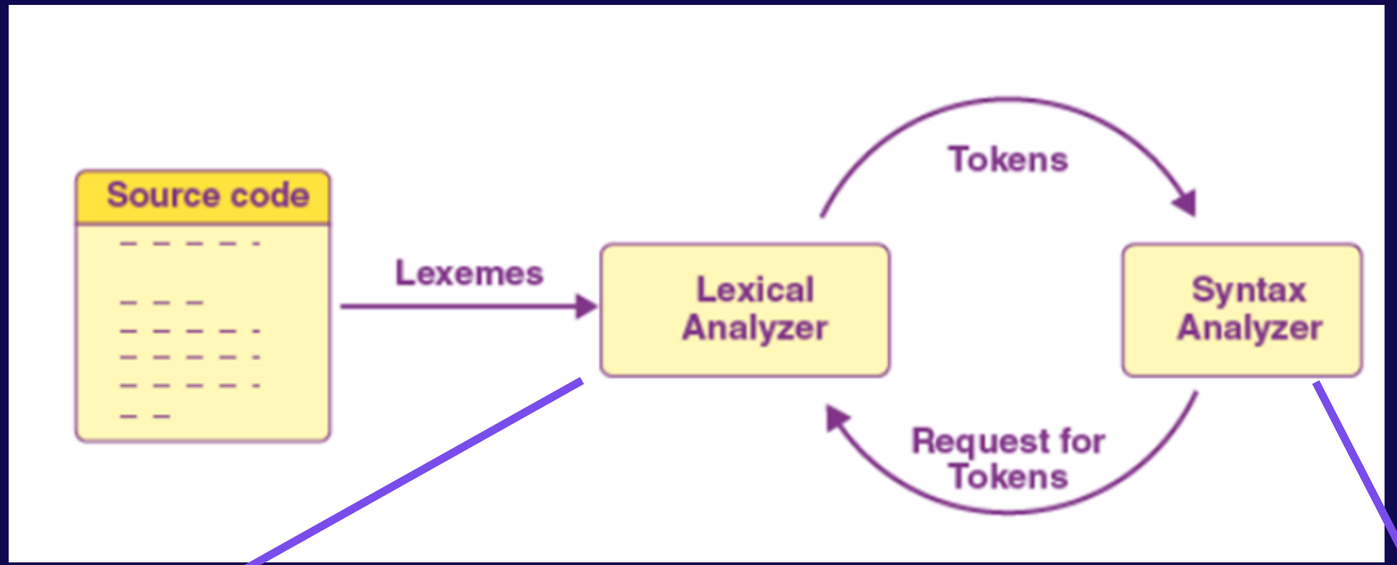
handles conditional statements with IF, ELIF, and ELSE branches.

represent loop constructs with specific conditions.

defines function or method declarations with parameters and an arrow (=>) followed by an expression.

represents list literals enclosed in square brackets, with optional elements.

Lexical Analysis and Parsing

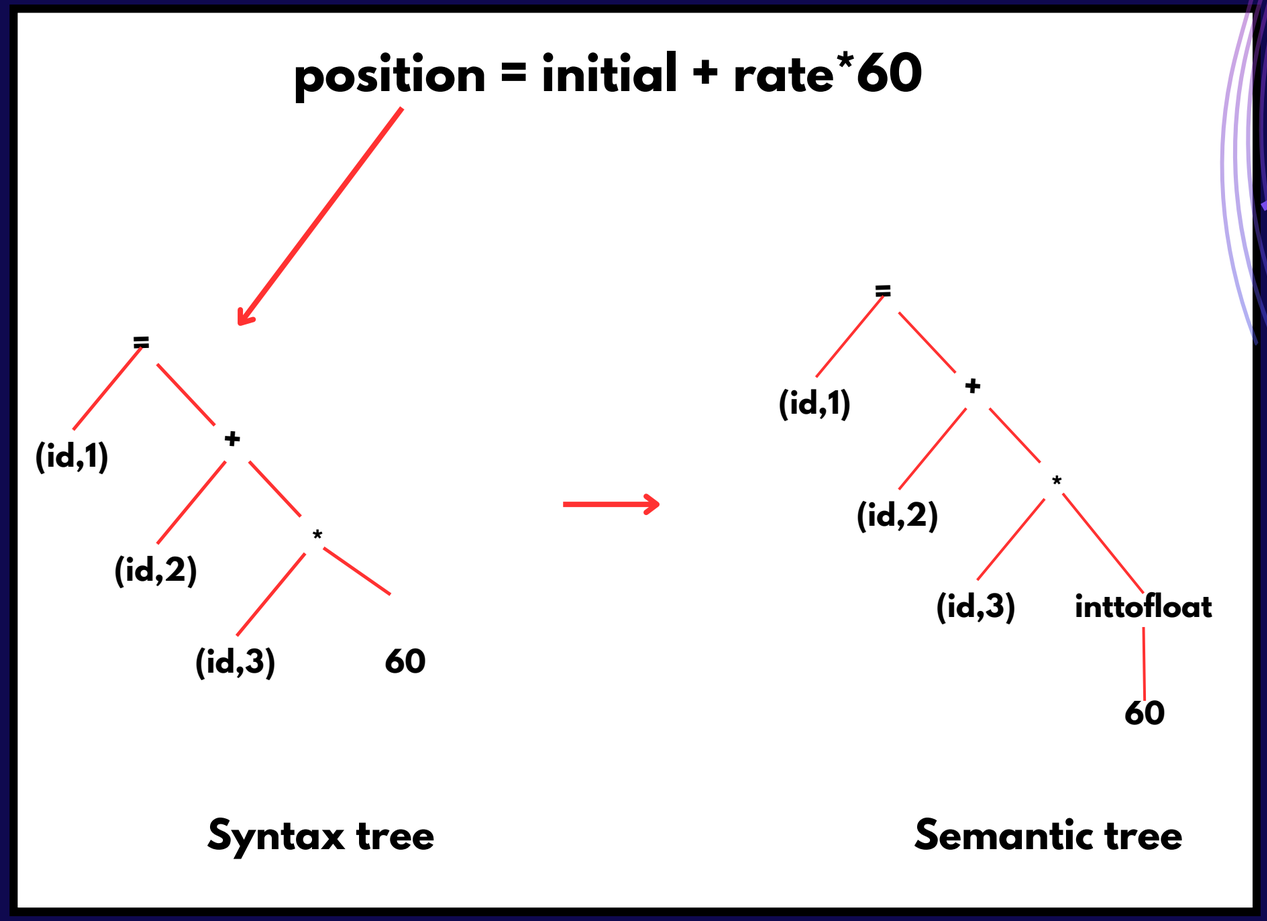


- **Tokenization:** Use regular expressions to define and extract tokens from the source code.
- **Error Handling:** Identify and report invalid characters and unrecognized tokens with informative error messages.

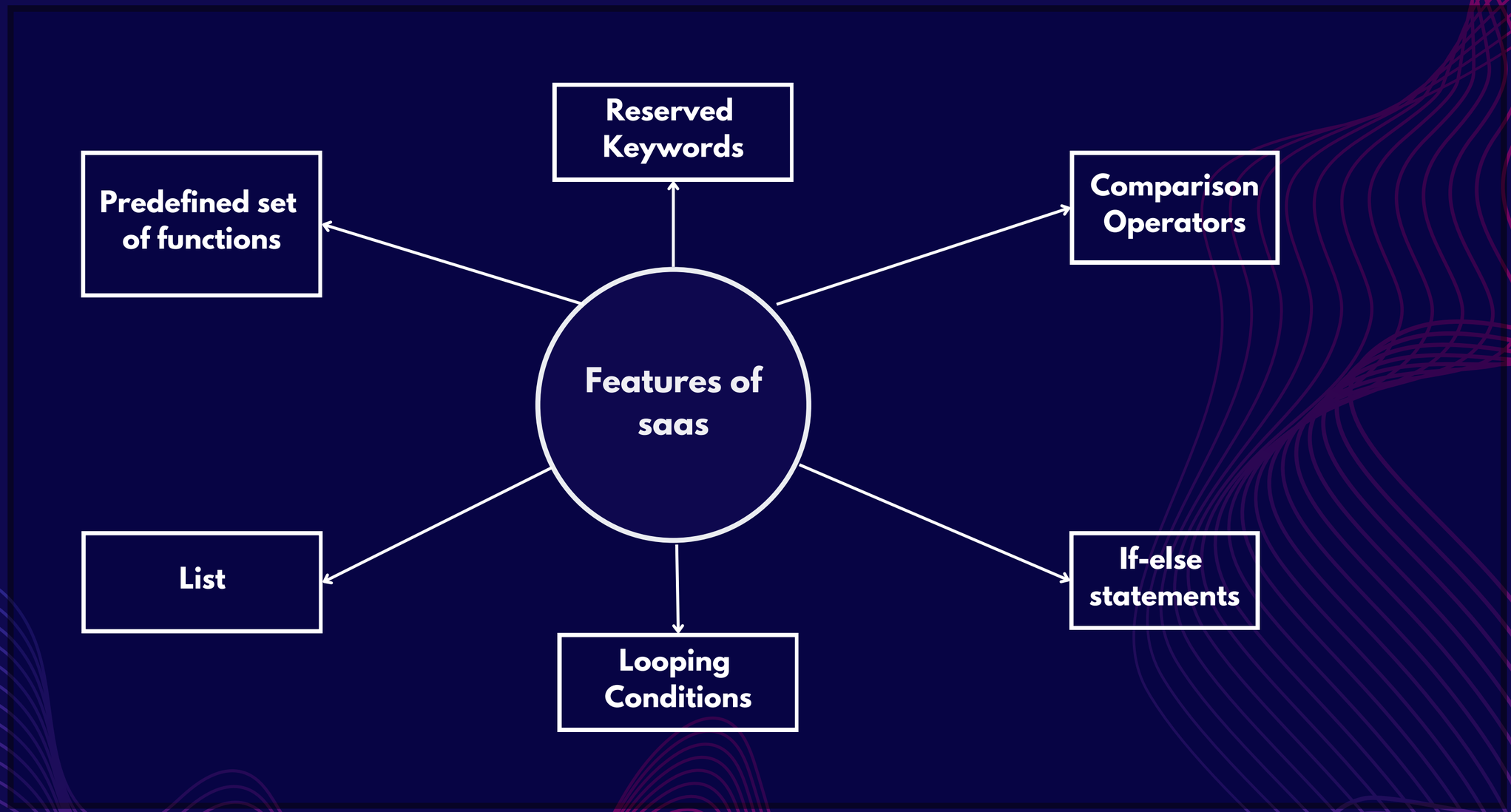
- Grammar Definition: Create a formal grammar using CFGs to define syntax rules.
- Abstract Syntax Tree (AST): Build an AST during parsing to represent the program's structure for easier analysis and execution

Semantic Analysis:

- Verifying that a program follows the language's semantic rules.
- Type Checking ensures expressions and variables have consistent and compatible types for proper operations and safe type conversions



Features of saas:




Thank
You