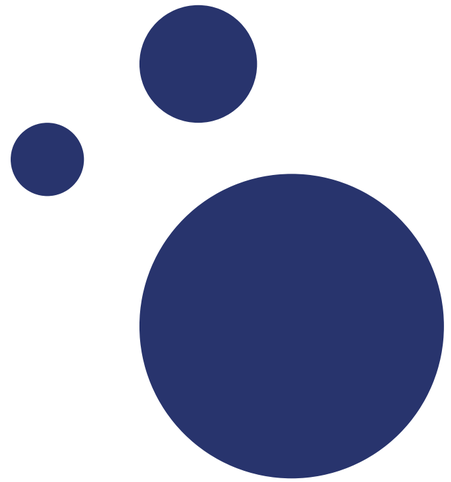
VISUALCOMP: EDUCATIONAL COMPILER WITH VISUALIZATION IN JAVA

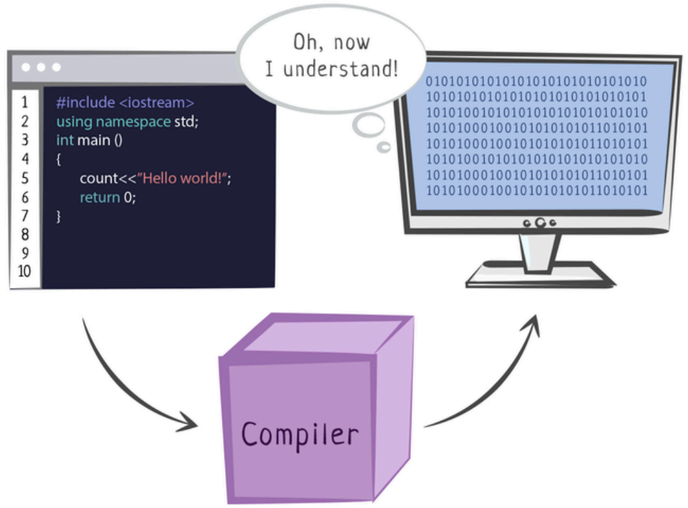
Juan Bustos - 20221020114
Maria Restrepo - 20221020044
COMPUTER SCIENCE III

OUTLINE



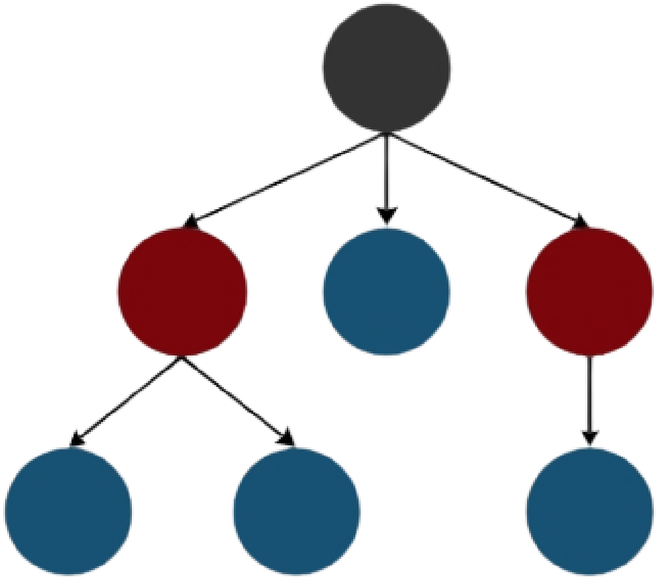
INTRODUCTION

- Compiler design is abstract and hard to visualize.
- Industrial tools hide inner steps and overwhelm beginners.
- Students often struggle to understand compilation phases.

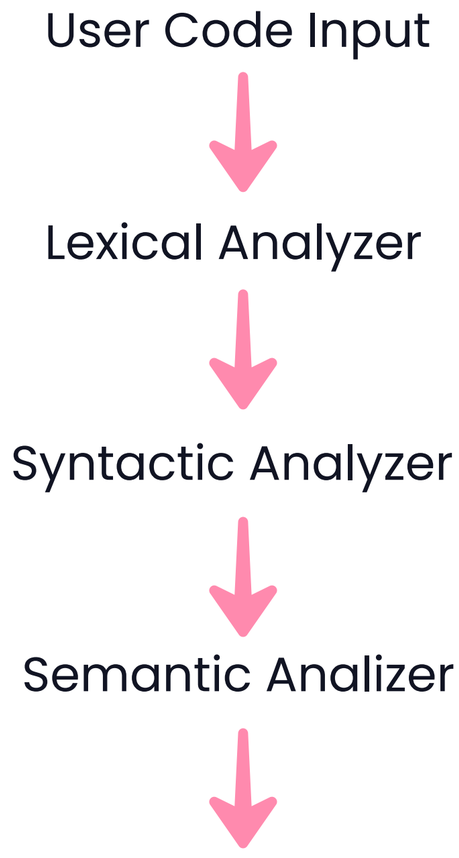


PROJECT GOAL

- Develop an educational compiler in Java.
- Supports input in a simplified programming language.
- Performs lexical, syntactical, and semantic analysis.
- The derivation tree is displayed after compilation.

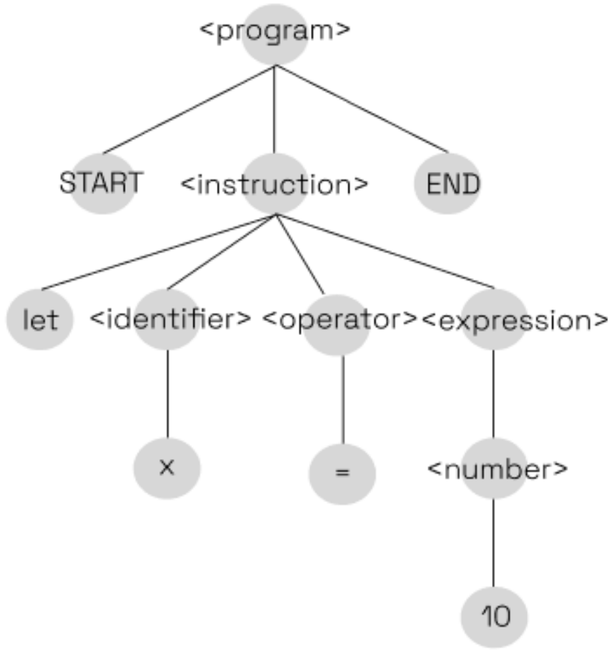


SYSTEM ARCHITECTURE



USER INTERFACE

Derivation Tree Simulator



START
let x = 10
END

Compile

LANGUAGE DESIGN

- Keywords: START, END, let, if, while, print
- Tokens: KEYWORD, IDENTIFIER, NUMBER, OPERATOR, PARENTHESIS, BRACES, etc.
- Grammar: defined using context-free productions.

Σ = { a–z, A–Z, 0–9, + - * / = < > ( ) { }

```
START

let x = 10

if ( x < 5 ) {

    print x

}

END
```

GRAMMAR RULES

<S> → START <Instructions> END
<Instructions> → <Instruction> <Instructions> | ε

<Instruction> → let <Identifier> = <Expression>
| print <Identifier>
| if (<Condition>) <Block>
| while (<Condition>) <Block>

<Block> → { <Instructions> }

<Expression> → <Identifier> <Operator> <Expression>
| <Number>

<Condition> → <Expression>
<Identifier> → [a-zA-Z][a-zA-Z0-9]*
<Number> → [0-9]+
<Operator> → + | - | * | / | = | < | >

ERROR HANDLING

Lexical Errors

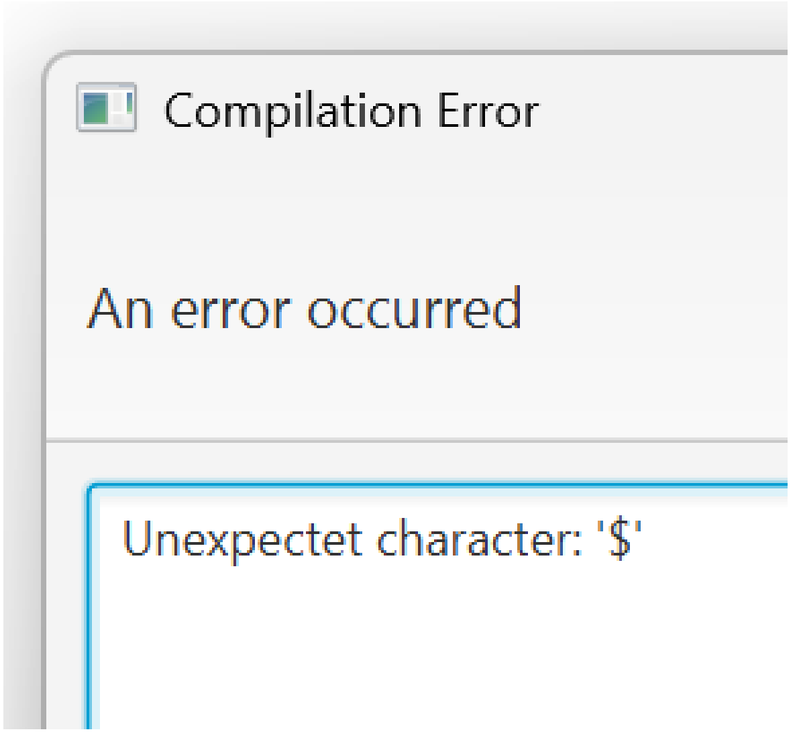
- Unexpected symbols → let x = 10$

Syntactic Errors

- Structure mismatches → while (x < 5 {

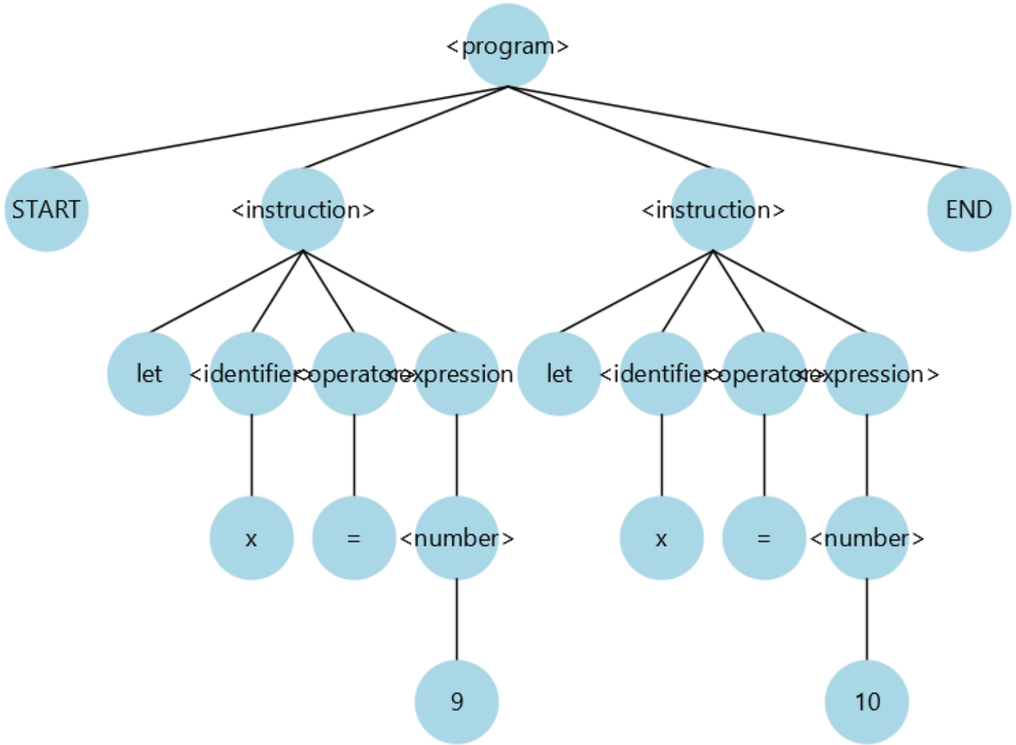
Semantic Errors

- Incorrect block usage → while x < 5 { print x }



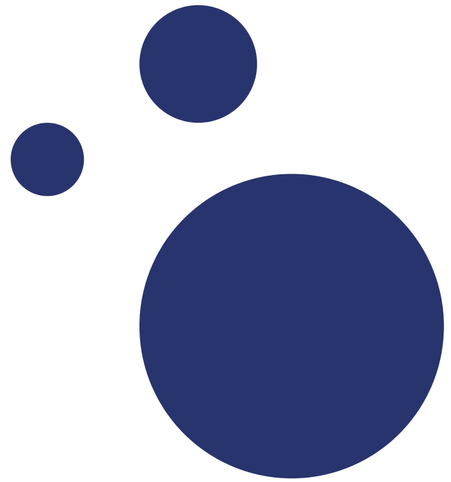
ACHIEVED RESULTS

- Successfully analyzes code: lexically, syntactically, semantically
- Tree visualization reflects real grammar derivation
- Interface is clean, animated, and interactive



CONCLUSIONS

- VisualComp makes learning compilers easier.
- It helps connect theory with practice.
- Visualization enables understanding of each compilation phase.
- Provides clear feedback on errors, supporting debugging skills.



REFERENCES

- Aho et al. Compilers: Principles, Techniques, and Tools, 2nd ed.

- T. Parr, The Definitive ANTLR 4 Reference. Dallas, TX, USA: The Pragmatic Bookshelf.

- Appel. Modern Compiler Implementation in Java