

# D lang Semantic Analyzer

**Gleb Popov, Timur Usmanov**  
Compiler Construction  
Innopolis University  
Fall 2025



# Project Context

## Dynamic Lang

object types are not specified and can change while program execution

the language assumes **interpretation**

## C++ Language

the implementation language is C++

it provides extensive memory management and optimization features

## Personal parser

hand-written parser

if you want a thing done well, do it yourself :)

# Recall: Lexer

```
var x := 5  
print x
```

```
tkVar tkIdent("x") tkAssign tkIntLiteral(5) tkNewLine  
tkPrint tkIdent tkNewLine
```

```
var t := {x:=1}  
t := t + {y:=2}
```

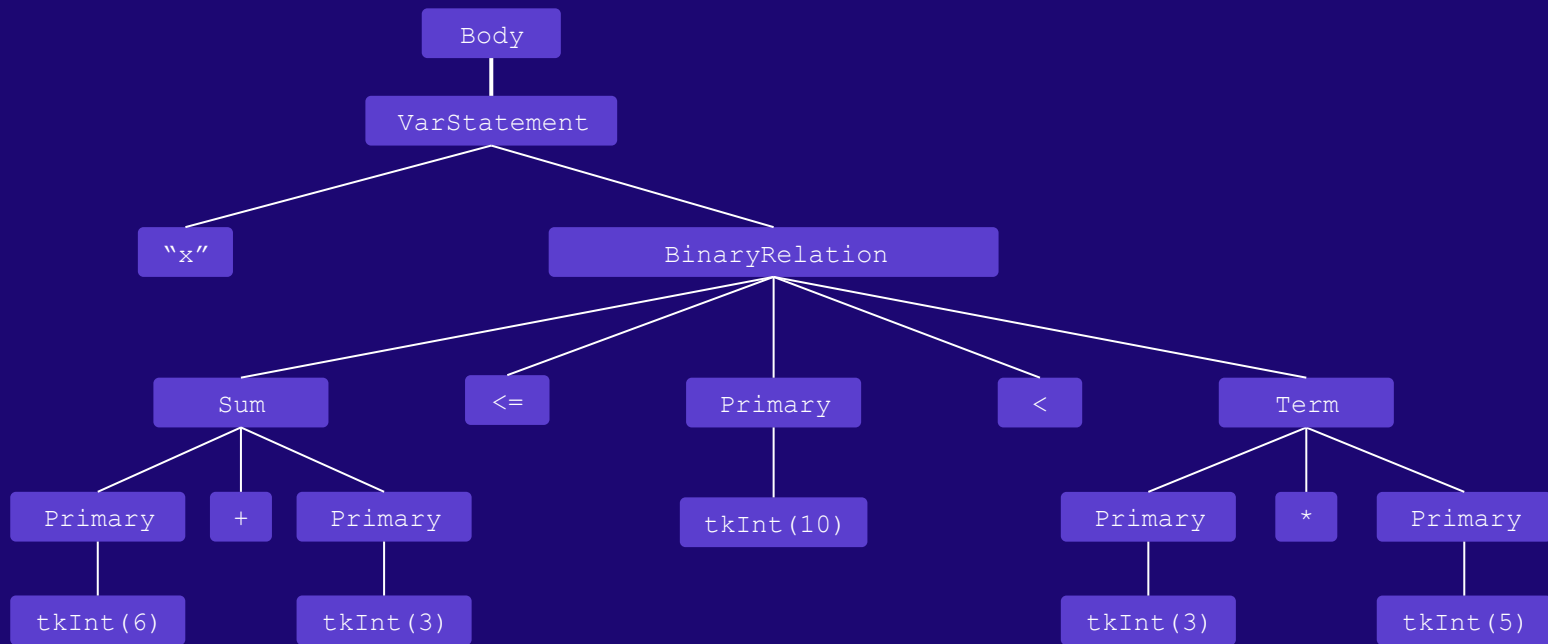
```
tkVar tkIdent(t) tkAssign tkOpenCurlyBrace tkIdent("x")  
tkAssign tkIntLiteral(1) tkClosedCurlyBrace tkNewLine  
tkIdent("t") tkAssign tkIdent("t") tkPlus tkOpenCurlyBrace  
tkIdent("y") tkAssign tkIntLiteral(2) tkClosedCurlyBrace
```

```
var x := 3  
if x < 10 then  
    print "small"  
else  
    print "big"  
end
```

```
tkVar tkIdent("x") tkAssign tkIntLiteral(3) tkNewLine tkIf  
tkIdent("x") tkLess tkIntLiteral(10) tkThen tkNewLine  
tkPrint tkStringLiteral("small") tkNewLine tkElse tkNewLine  
tkPrint tkStringLiteral("big") tkNewLine tkEnd tkNewLine
```

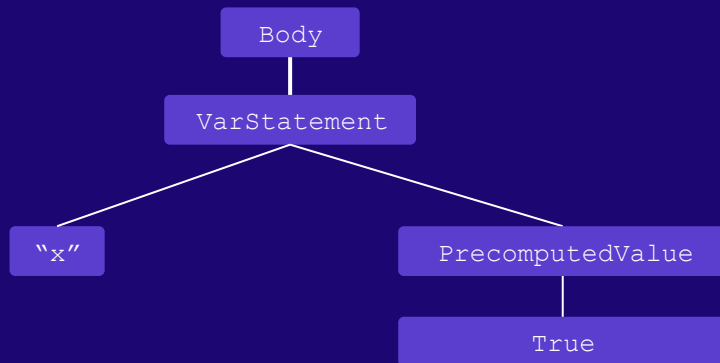
# Recall: Syntax Analyzer

```
var x := 6 + 3 <= 10 < 3 * 5
```



# Semantic Analysis: Output

```
var x := 6 + 3 <= 10 < 3 * 5
```



# Semantic Analysis: Non-Modifying Checks

- Closure parsing (capture variables outside of function scope)
- Variable existence checking
- Validation of element name uniqueness in tuples
- Unused variables warnings
- Zero division warnings
- `exit` only in cycles, `return` only in functions
- function return type and purity inference
- ...

All 36 amazing semantic analysis features can be found [here](#).

# Semantic Analysis: Modifying Checks

- Expression value precomputation
- Expression type inference
- Removing redundant branches in *if*-statements (if the condition value is known)
- Removing dead code (after `return` / `exit`)
- Numeric sum reordering & optimization
- Check if *while*-condition is false for the first iteration
- String concatenation optimization
- ...

All 36 amazing semantic analysis features can be found [here](#).