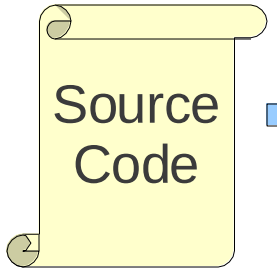
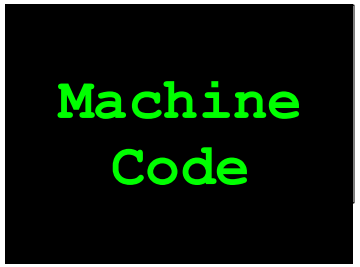
Compiler
– 3-1. Role of the Syntax Analysis –

JIEUNG KIM

YONSEI UNIVERSITY

Where are we?





Outlines

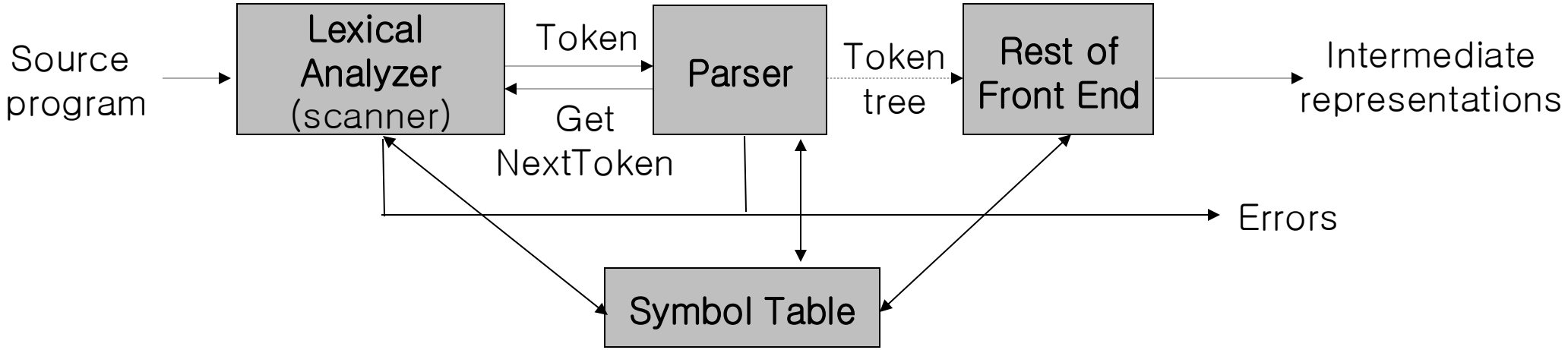
- Role of the syntax analysis (parser)
- Context free grammar
- Push down automata
- Top-down parsing
- Bottom-up parsing
- Simple LR
- More powerful LR parsers and other issues in parsers
- Syntactic error handler
- Parser generator

Role of the syntax analysis (parser)

Role of the parser

- What is syntax analysis (or Parsing)?
  - After lexical analysis (scanning), we have a series of tokens.
  - In syntax analysis (or parsing), we want to interpret what those tokens mean.

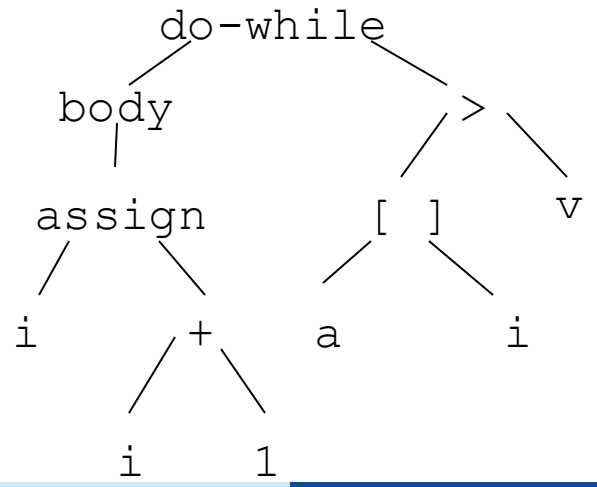
Role of the parser



Original code

```
Do
    i = i+1;
while  (a[i] < v);
```

Syntax Tree



three-address instructions

```
1: i = i + 1
2: t1 = a[ i ]
3: if t1 < v goto 1
```

Role of the parser

- Role of the parser
  - To check syntax (= string recognizer)
    - And to report syntax errors accurately
  - To invoke semantic actions
    - For static semantics checking
      - Example: type checking of expressions, functions, etc.
    - For syntax-directed translation of the source code to an intermediate representation

Role of the parser

- Syntax error handling
  - What if there is no error handling actions?
    - Planning error handling right from start is important
  - Levels of error
    - Lexical errors: misspellings of id, keyword, operators
    - Syntax errors: misplaced semicolons or braces, case without switch
    - Semantic errors: type mismatch between operators and operands
    - Logical errors: incorrect reasoning (difficult to detect)
      - Example: = instead of ==, unfinishable recursions
  - Parser should detect errors efficiently and ASAP

Role of the parser

- Syntax error handling
  - Lexical

```
if x < 1 thenn y = 5.
```

“Typos”

  - Syntactic

```
if ((x < 1) & (y > 5))) …
{ … { … _ … }
```

  - Semantic

```
if (x + 5) then
```

Type errors
Undefined IDs, etc

  - Logical errors

```
if (I < 9) then ...
```

Should be <= not <
Bugs
Compiler cannot detect logical errors

Syntax error handling

- Syntax error handling
  - Goals of error-handling in Parser
    - Report the presence of errors clearly and accurately
    - Recover from each error quickly to detect subsequent errors
    - Add minimal overhead to the processing correct programs
  - Must report exact place in the source program

Syntax error handling

- Error recovery strategy
  - How should the parser recover from errors
    - Simplest: quit after error message?
    - List all possible errors: annoying for avalanche of “spurious” errors
    - Balance is important
      - Panic mode, phrase-level, error-productions, and global-correction

Study of parser

- Study of parser
  - Role 1: Checking the syntax
    - Base on grammar
  - The process of discovering a derivation for some sentence
    - Need a mathematical model of syntax — a grammar G
    - Need an algorithm for testing membership in L(G)
  - Derivations
    - A sequence of derivation steps
    - At each step, we choose a non-terminal to replace
    - Different choices can lead to different derivations

Study of parser

- Study of parser
  - Underlying mathematical models
    - Context free grammar
    - Push down automata
  - How to use those models
    - Top-down parsing
    - Bottom-up parsing
    - Simple LR
    - More powerful LR parsers and other issues in parsers
  - Tools
    - Parser generator (including Yacc)

Questions?