Quiz

Get 1/4

1-2. Enumerate the 2 influences on language design.

3. What is the basis for imperative languages?

4. Who is your lab instructor? What can you say about him/her?

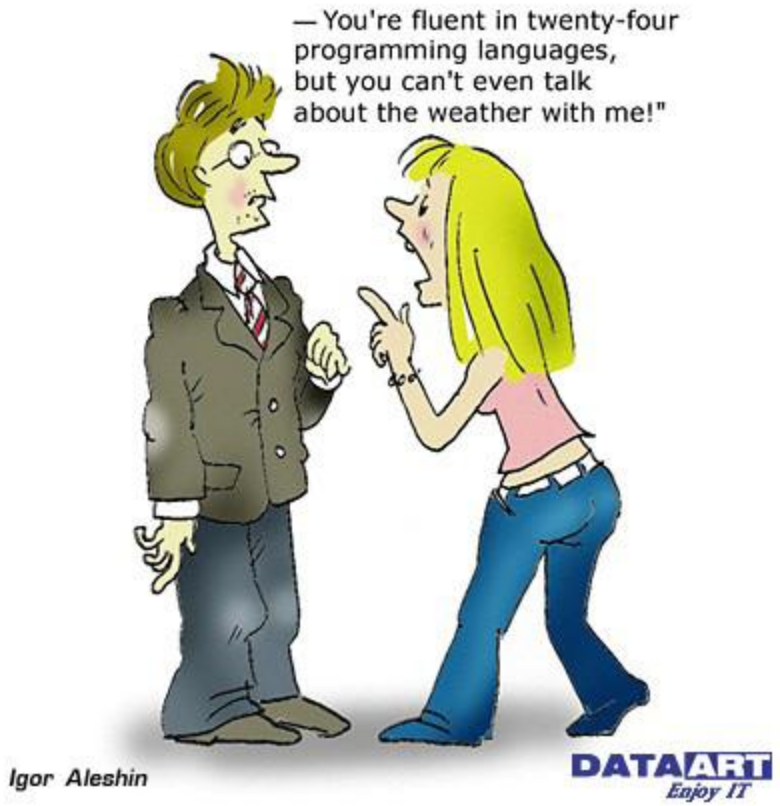
Chapter 3: Language Translation Issues

CMSC 124, 1st Semester, AY 2009-10

Chapter 3: Language Translation Issues

Topics Covered

- Stages in Translation
- Language Specification
  - ❖ Syntax
  - ❖ Semantics
- Syntax
  - ❖ Grammars and BNF
  - ❖ Syntax Diagrams
  - ❖ Finite State Automata
  - ❖ Pushdown Automata
- Semantics



Chapter 3: Language Translation Issues

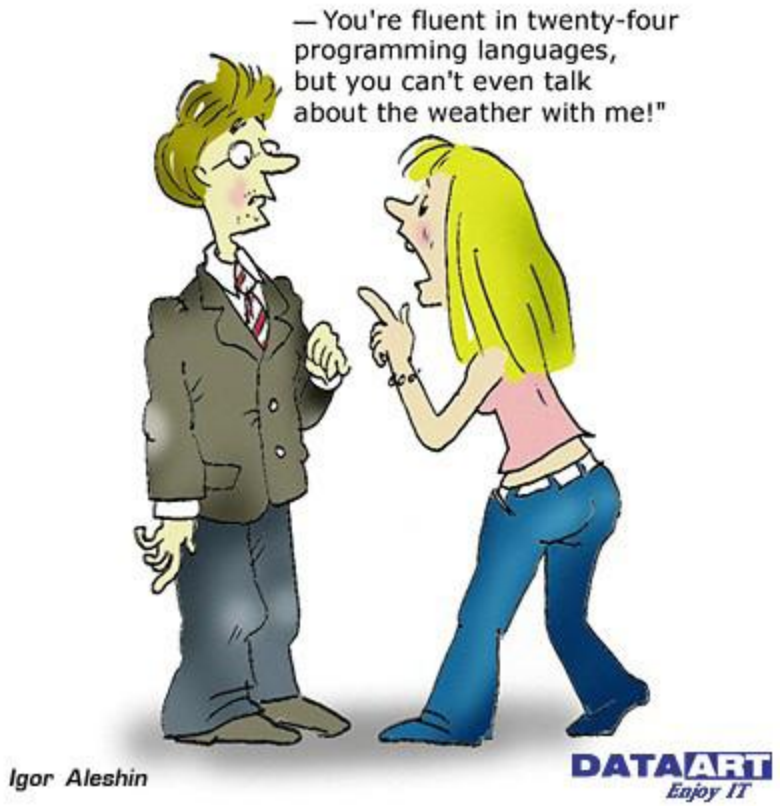
Writing a Translator/Compiler: Span

- Programming Languages **(CMSC 124)**
- Machine Architecture **(CMSC 132)**
- Language Theory **(CMSC 141)**
- Algorithms **(CMSC 123)**
- Software Engineering **(CMSC 128)**
- Machine-Level Programming **(CMSC 131)**

Chapter 3: Language Translation Issues

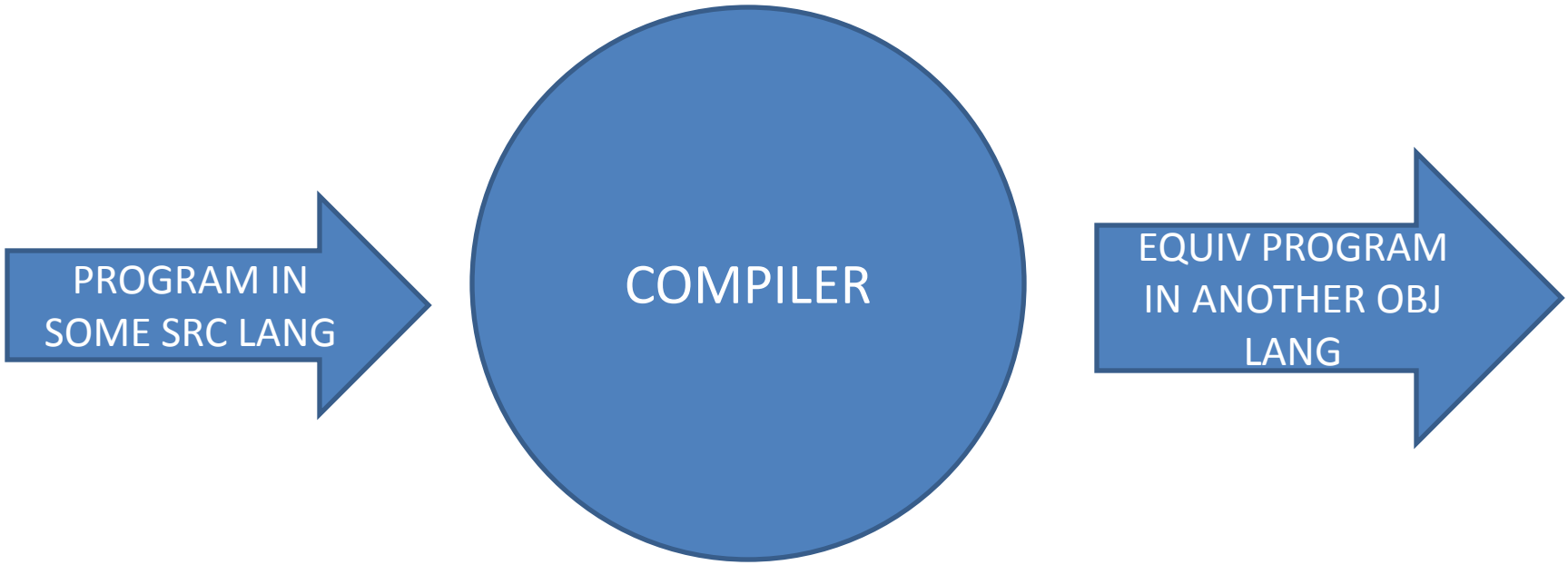
Topics Covered

- Stages in Translation
- Language Specification
  - ❖ Syntax
  - ❖ Semantics
- Syntax
  - ❖ Grammars and BNF
  - ❖ Syntax Diagrams
  - ❖ Finite State Automata
  - ❖ Pushdown Automata
- Semantics



Chapter 3: Language Translation Issues

Recall: Compilers



Analysis-Synthesis Model of Compilation

Two Parts of Compilation

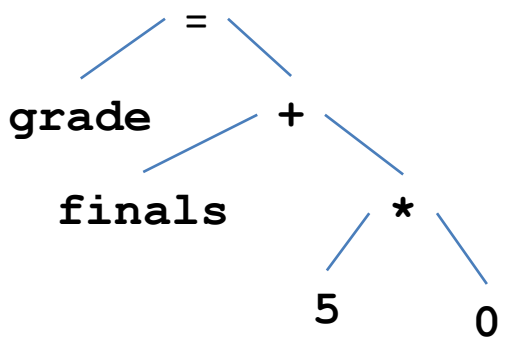
- **Analysis**

  Breaks up the source program into constituent pieces and creates an intermediate represent'n of the source program.

- **Synthesis**

  Constructs the desired target program from the intermediate representation

```
grade = finals + 5 * 0
```



Chapter 3: Language Translation Issues

Analysis of the Source Program

- **Linear Analysis**
  Stream of characters is read from left to right and group into tokens.
  - **Tokens –** Sequences of chars. having a collective meaning.
- **Hierarchical Analysis**
  Tokens are grouped hierarchically into nested collections with collective meaning.
- **Semantic Analysis**
  Performs certain checks to ensure that components of a program fit together meaningfully.

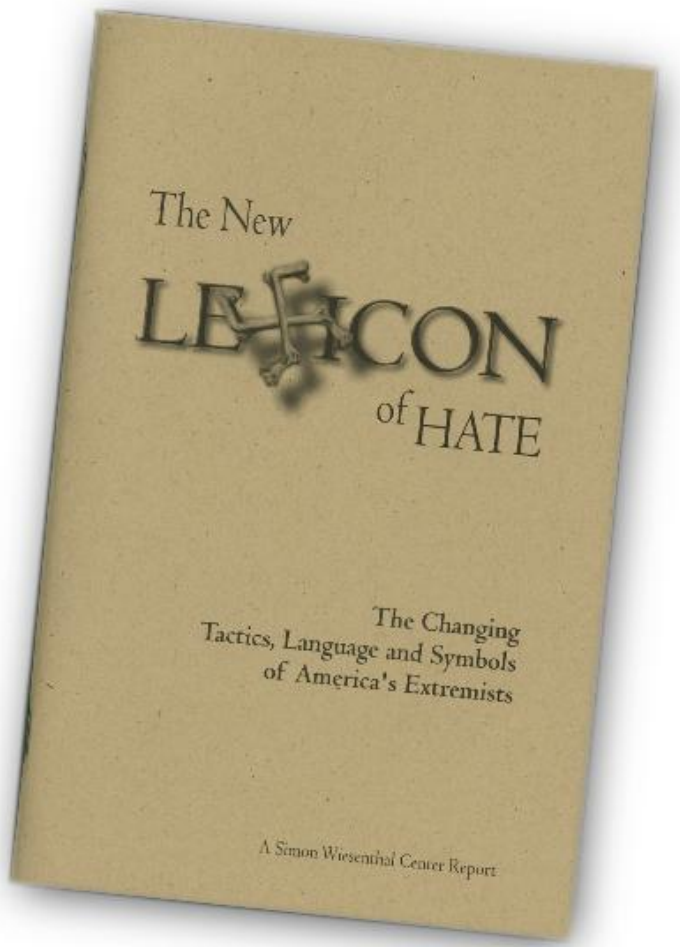
Chapter 3: Language Translation Issues

Compilation Phase #1: Lexical Analysis

- AKA linear analysis or scanning.
- Consider the following:

```
grade = finals + 5 * 0
```

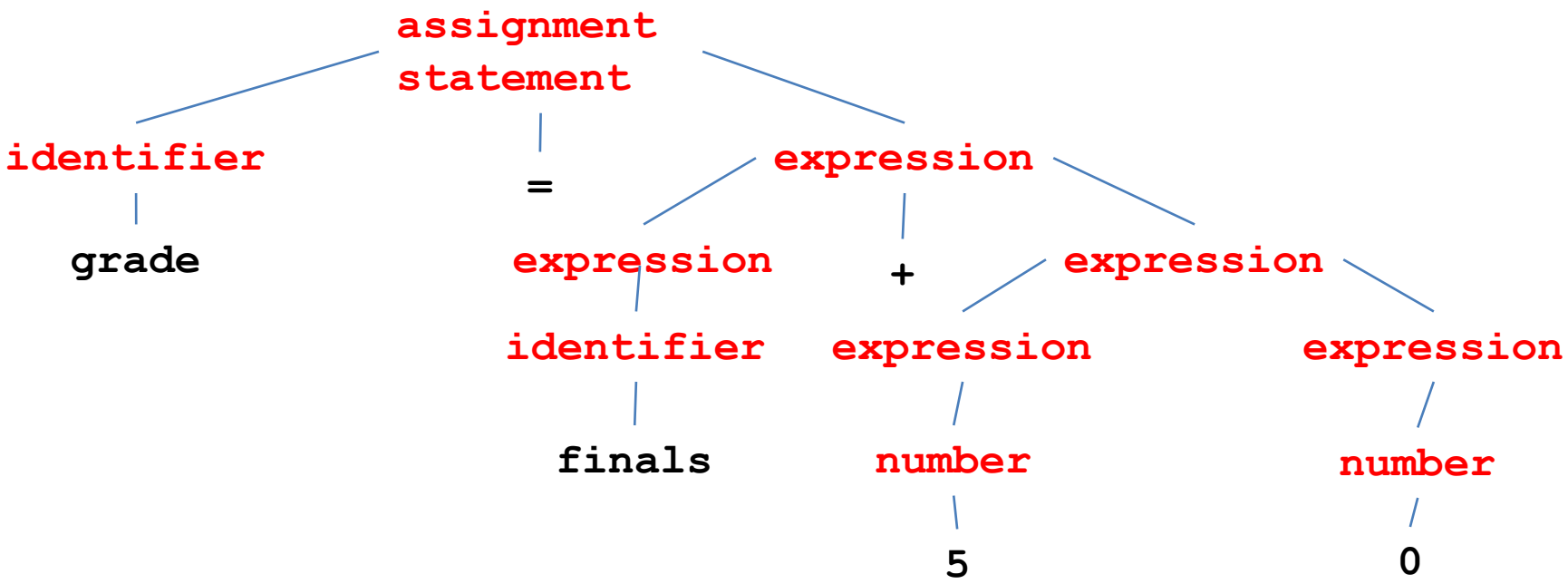
- Tokens:
  - identifier **grade**
  - assignment symbol **=**
  - identifier **finals**
  - plus sign **+**
  - number **5**
  - multiplication sign ***
  - number **0**
- Blanks are normally eliminated.



Chapter 3: Language Translation Issues

Compilation Phase #2: Syntax Analysis

- AKA hierarchical analysis or parsing.
- Grouping tokens into grammatical phrases.
- The parse tree for **`grade = finals + 5 * 0`**



Chapter 3: Language Translation Issues

Compilation Phase #2: Syntax Analysis

- **Expression Rules:**
  - ✓ Any **`identifier`** is an expression.
  - ✓ Any **`number`** is an expression.
  - ✓ If **`expression1`** and **`expression2`** are expressions, then so are:
    - ➢ **`expression1 + expression2`**
    - ➢ **`expression1 * expression2`**
    - ➢ **`( expression1 )`**

Chapter 3: Language Translation Issues

Compilation Phase #2: Syntax Analysis

- **Statement Rules:**
  - ✓ If **`identifier`** is an identifier **`expression`** and is an expression then:

    `identifier = expression`

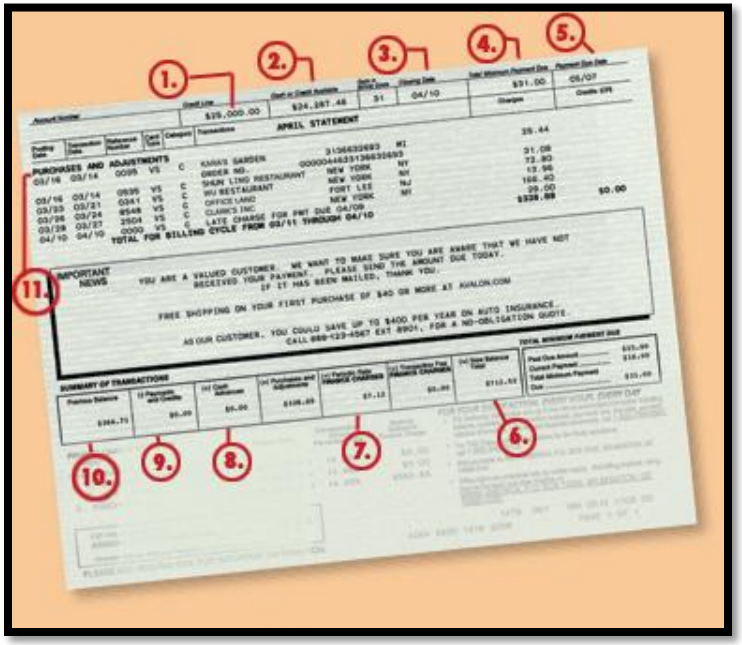
    is a statement.

  - ✓ If **`expression`** is an expression and **`statement`** is a statement then:

    `while (expression) do statement`

    `if (expression) then statement`

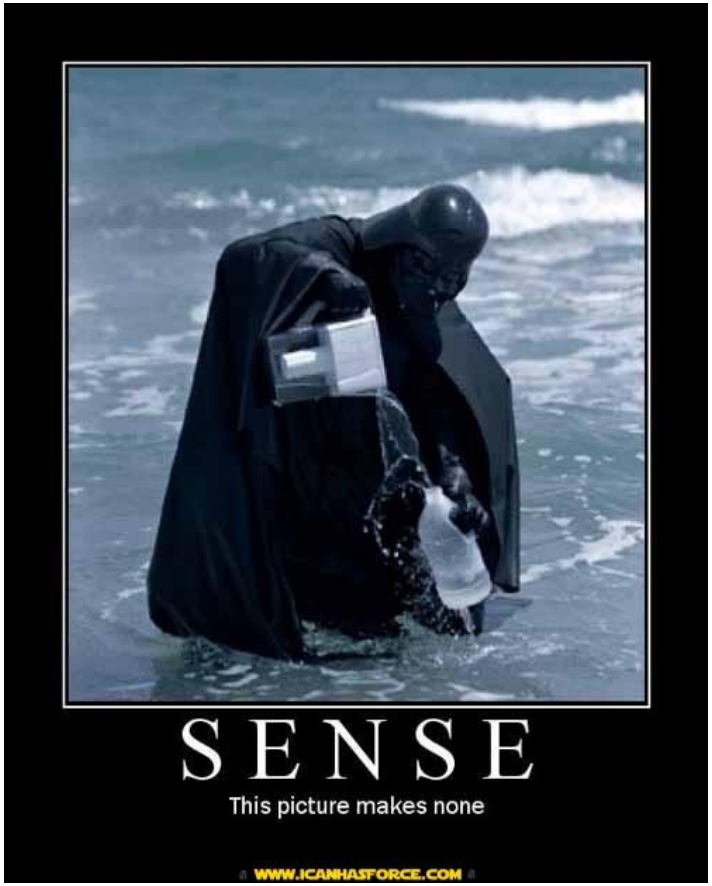
    are statements.



Chapter 3: Language Translation Issues

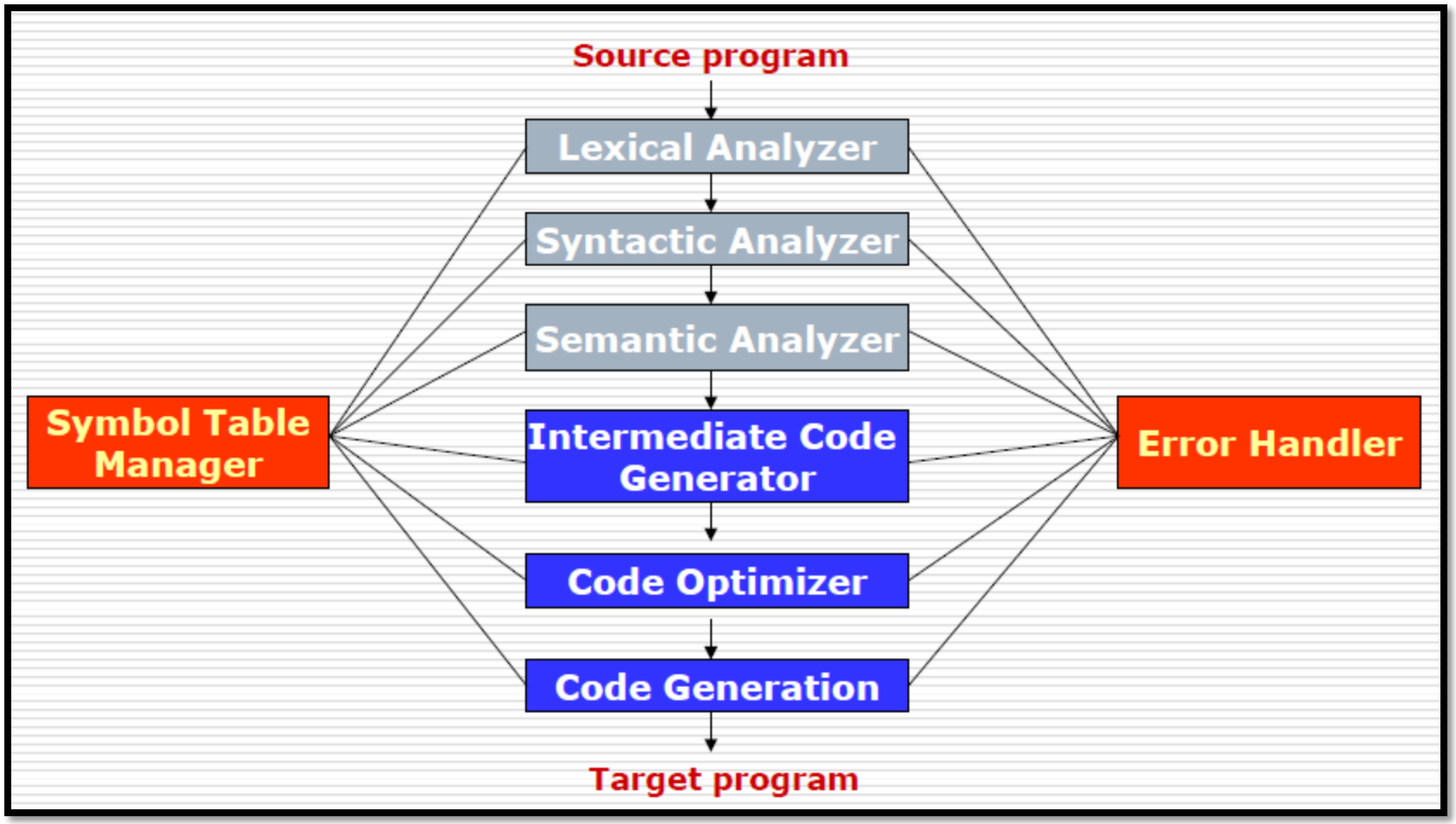
Compilation Phase #3: Semantic Analysis

- Checks for semantic errors.
- Performs type checking.
- Identifies the operators and operands of expressions and statements.
- Checks for valid operands.



Chapter 3: Language Translation Issues

Compilation Phases



Chapter 3: Language Translation Issues

Two Other Activities in the Phases of Compilation

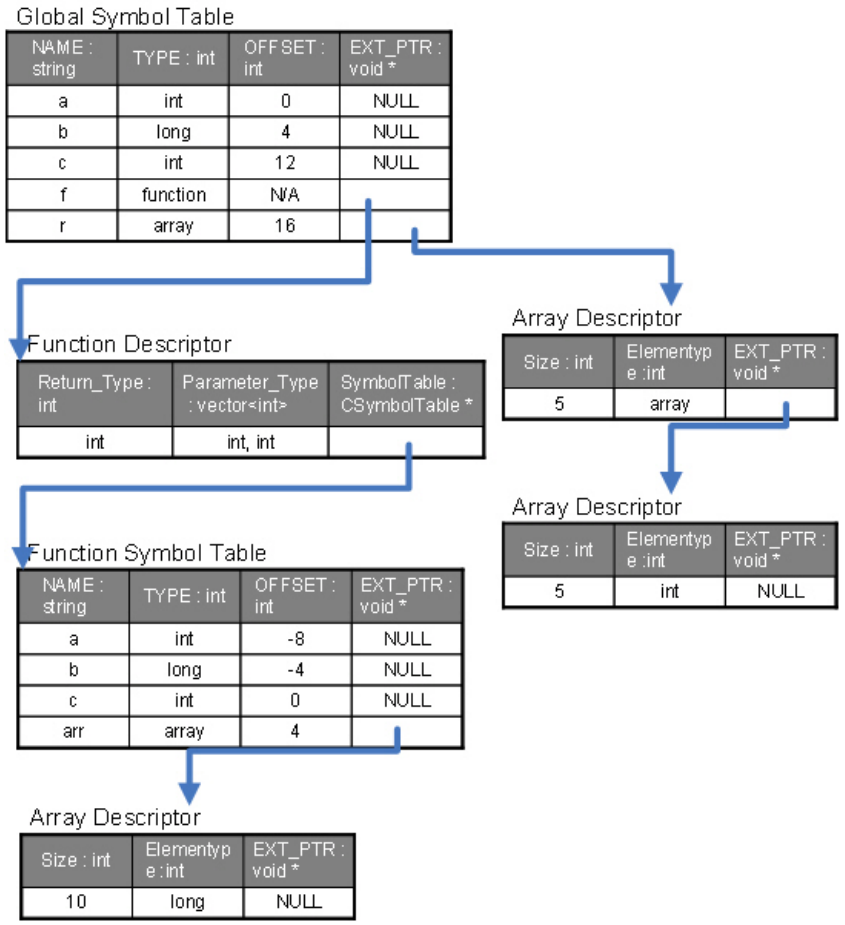
- Symbol-Table Maintenance
- Error Detection and Reporting

Chapter 3: Language Translation Issues

Symbol-Table Maintenance

Symbol-Table

A data structure used by a language translator where each identifier in a program's source code is stored; Associated with information relating to its declaration or appearance in the source, such as its type, scope level and sometimes its location.



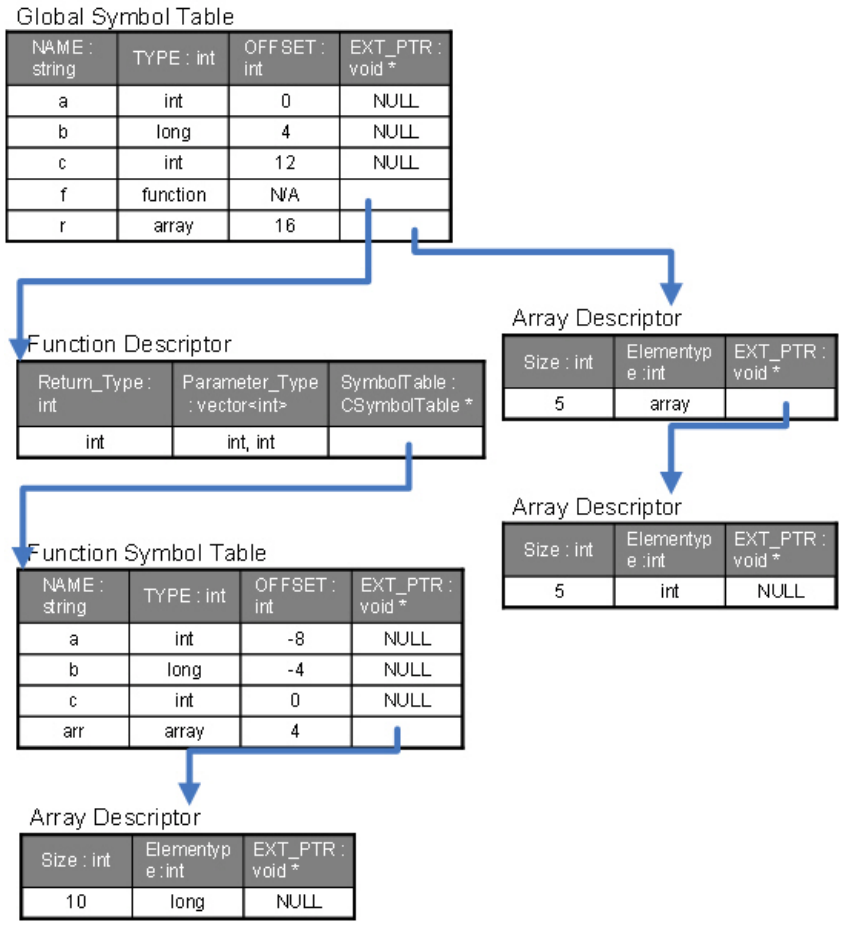
Chapter 3: Language Translation Issues

Symbol-Table Maintenance

Symbol-Table

A data structure that contains a record for each identifier.

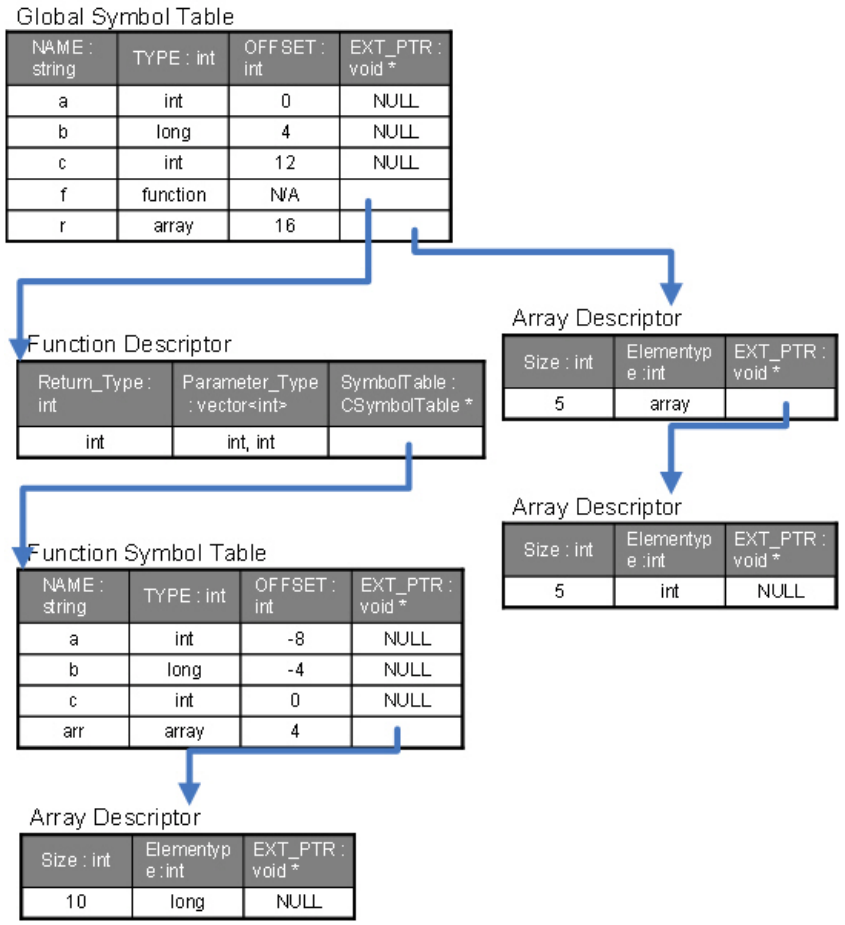
- Type
- Scope
- For Procedures:
  - Parameter type
  - No. of parameters
  - Method of passing



Chapter 3: Language Translation Issues

Symbol-Table Maintenance

- It is essential to record identifiers.
- Identifiers are entered during the lexical analysis phase.



Chapter 3: Language Translation Issues

Error Detection and Reporting

- Each phase can encounter an error.
- To proceed, a phase must somehow deal with the error.
- A compiler that stops when it finds the first error is not as helpful.

Chapter 3: Language Translation Issues

Error Detection and Reporting

- **Lexical Analysis Phase**
Characters do not form a valid token.

- **Syntax Analysis Phase**
Token streams violates the structure rules.

- **Semantic Analysis Phase**
Detect constructs that have the right syntax structure but no meaning to the operation involved.
    - Eg. Add – an array and a procedure

Something to Ponder

The 3 phases of compilation discussed are indeed essential, important. Why?

Something to Ponder

Chapter 3: Language Translation Issues

Review: Lexical Analysis (or Linear Analysis)

- A pattern matcher.
- It collects characters into logical groupings and assign internal groupings according to their structure.
- Character groupings are called **lexemes.**
- Internal codes are called **tokens.**

Eg:

```
result = oldsum - value / 100;
```

TOKEN	LEXEME
IDENT	**result**
ASSIGN_OP	**=**
IDENT	**oldsum**
SUBTRACT_OP	**-**
IDENT	**value**
DIVISION_OP	**/**
INT_LIT	**100**
SEMICOLON	**;**

Chapter 3: Language Translation Issues

Review: Lexical Analysis (or Linear Analysis)

- **Goal:** Extract lexemes from a given input string and produce tokens.
- The process includes skipping comments and blanks.
- It also inserts lexemes for user-defined names into the symbol-table.
- The lexical analyzer is responsible for the initial construction of the symbol-table.

TOKEN	LEXEME
IDENT	**result**
ASSIGN_OP	**=**
IDENT	**oldsum**
SUBTRACT_OP	**-**
IDENT	**value**
DIVISION_OP	**/**
INT_LIT	**100**
SEMICOLON	**;**

Chapter 3: Language Translation Issues

Review: Syntax Analysis (or Hierarchical Analysis)

- **Goals:**
  1. The syntax analyzer must check the input program to determine whether it is syntactically correct.
  2. Produce a complete parse tree, or at least trace the structure of the complete parse tree for syntactically correct input.

- The parse tree is the basis for translation.

Eg.

• Assume this is a statement in a certain PL: `([x, y])`

• These are the rules:

- `S ::= [A, S] | A`
- `A ::= C | (S)`
- `C ::= x | y | x`

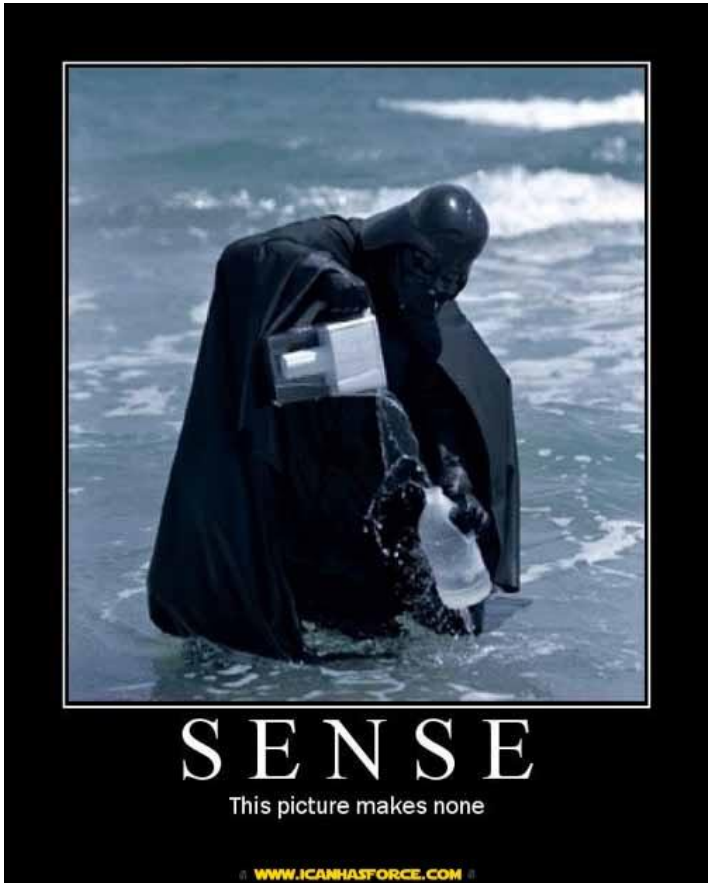
• **Generate** a parse tree?

• **Remember:** Grammars are used to describe syntax.

Chapter 3: Language Translation Issues

Review: Semantic Analysis

- Checks for semantic errors.
- Performs type checking.
- Identifies the operators and operands of expressions and statements.
- Checks for valid operands.



Intermediate Code Generation

- Some compilers generate an explicit intermediate representation or a program for an abstract machine.
- Intermediate representation have many forms.
- **Eg: Three Address Code**
  - Similar to assembly language.
  - Every memo location can act as a register.
  - Consists of a sequence of instructions.
  - Each instruction has at most three operands.

Chapter 3: Language Translation Issues

Compilation Phase #4: Intermediate Code Generation

Properties

- Each instruction has at most one operator plus assignment.
- Compiler must generate a temporary name to hold the value computed.
- Some instructions have fewer than three operands.

Eg:

```
position = initial + rate * 60
Remember: id1 = id2 + id3 * 60
```

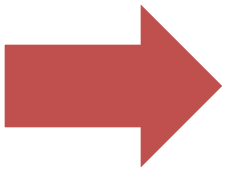
```
temp1 = inttoreal(60)
temp2 = id3 * temp1
temp3 = id2 + temp2
id1 = temp3
```

Chapter 3: Language Translation Issues

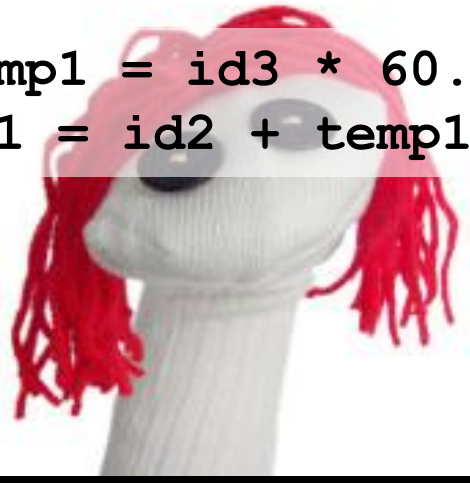
Compilation Phase #5: Code Optimization

- Improves the intermediate code to produce a faster running machine code.
- Compilers vary in the amount of time spent in this phase.
- **Eg:**

```
temp1 = inttoreal(60)
temp2 = id3 * temp1
temp3 = id2 + temp2
id1 = temp3
```

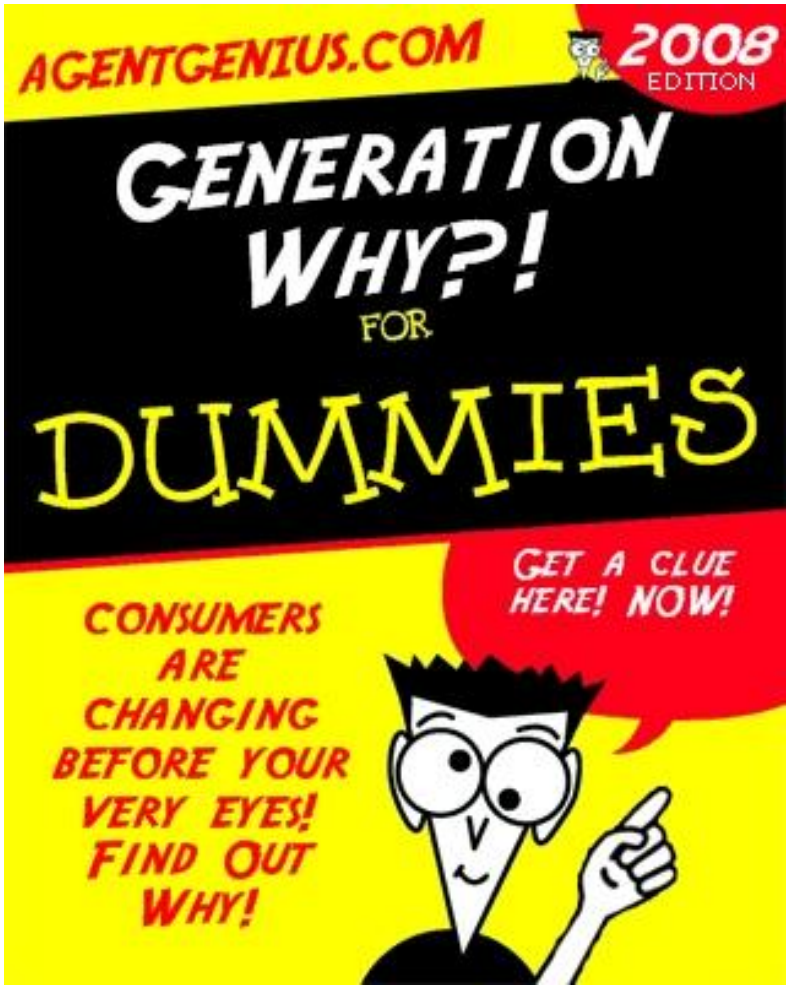


```
temp1 = id3 * 60.0
id1 = id2 + temp1
```



Chapter 3: Language Translation Issues

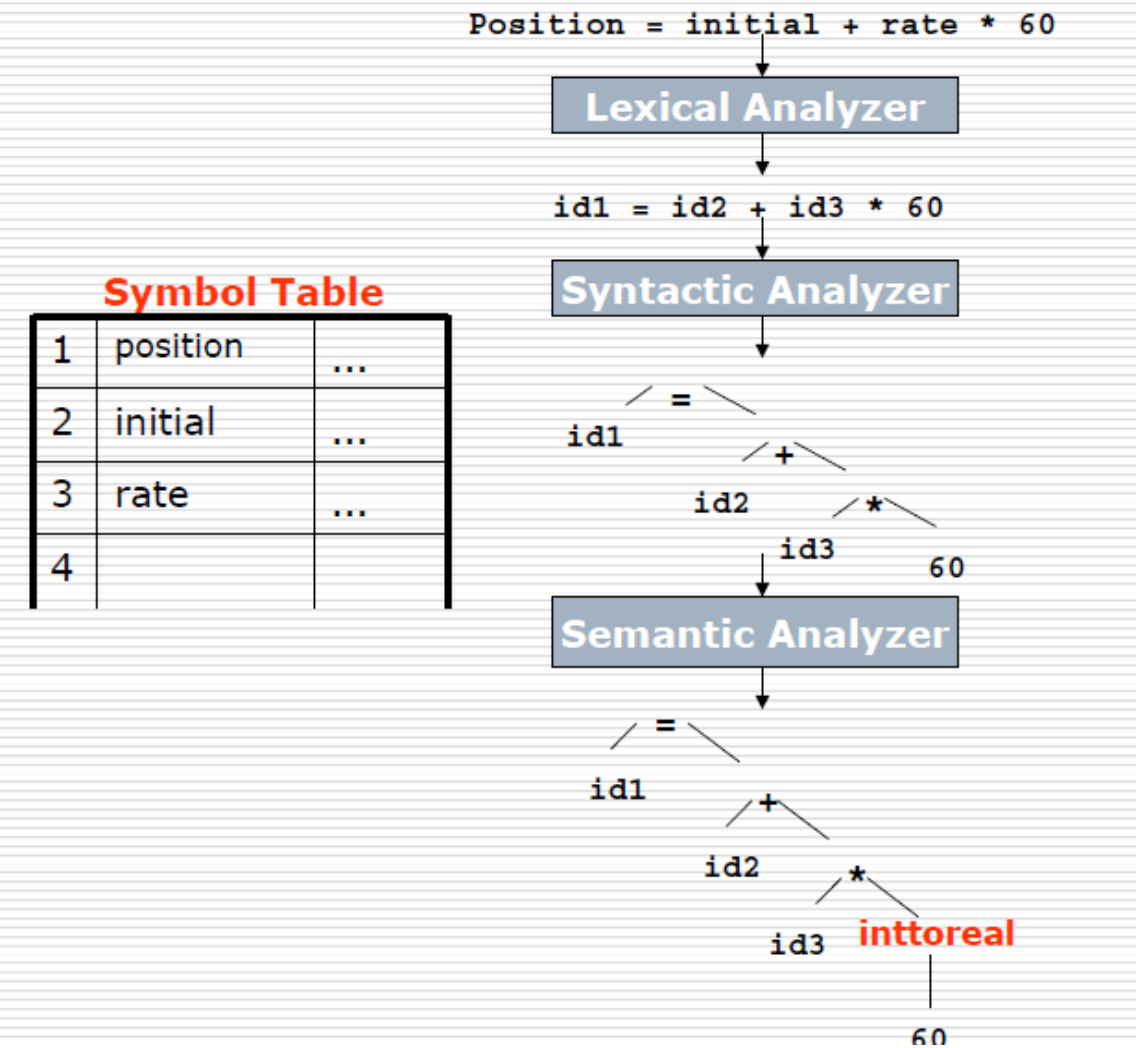
Compilation Phase #6: Code Generation



- Final phase.
- Generation of the target code from relocatable machine or assembly machine or assembly code.
  - Memory locations of variables are selected.
  - Each intermediate code are translated into an equiv. sequence of machine instructions.
- Assignment of variables to registers is **crucial**!

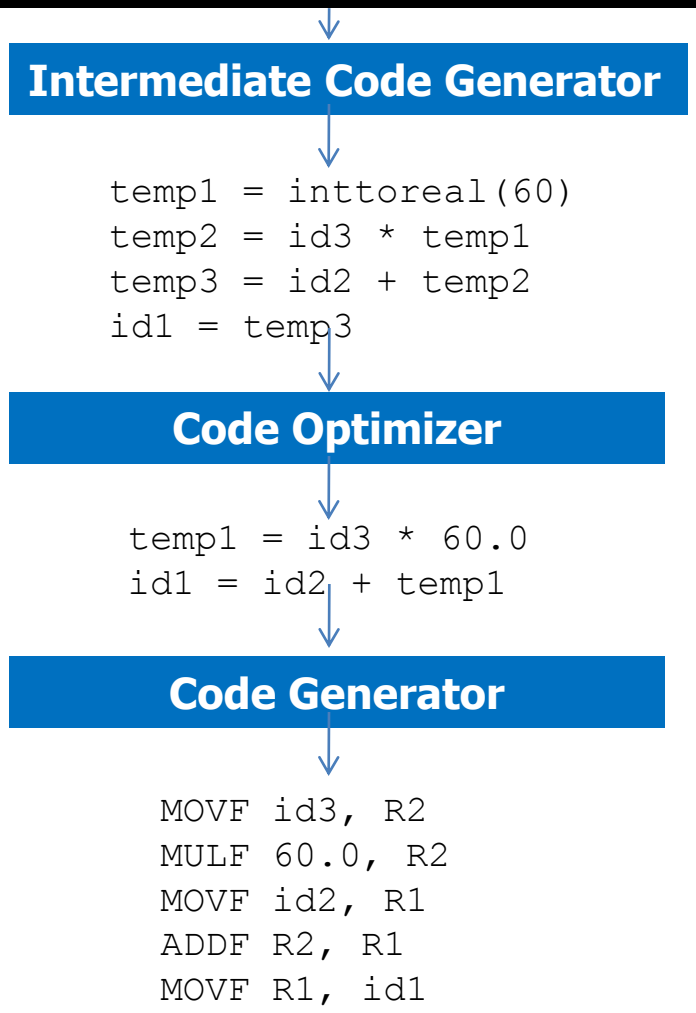
Chapter 3: Language Translation Issues

Translation of a Statement



Chapter 3: Language Translation Issues

Translation of a Statement



Chapter 3: Language Translation Issues

Language Specification

Syntax

- "What its program looks like."
- Refers to ways symbols may be combined to create well-formed sentences in the language.
- Formal methods include Context-Free Grammar or Backus Naur Form (BNF)

Semantics

- "What its program means."
- Much more difficult to describe.
- Informal methods and suggestive examples are normally used.