Lexical Analysis (Scanning)

Part One

Lecture Two

Compiler (Why?)

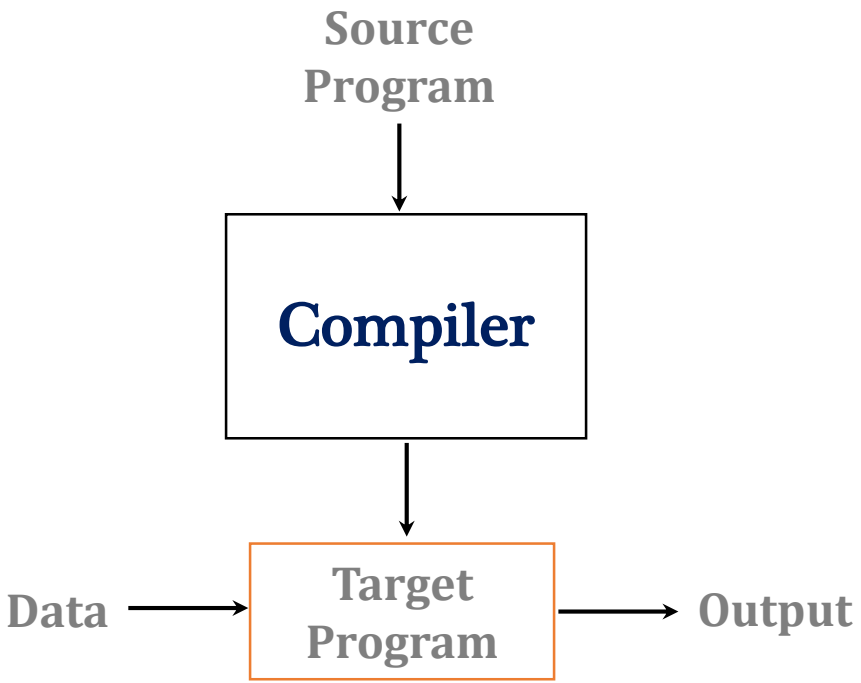
Machine Language

The instruction to move the number 2 to the location 0000.

C7 06 0000 0002

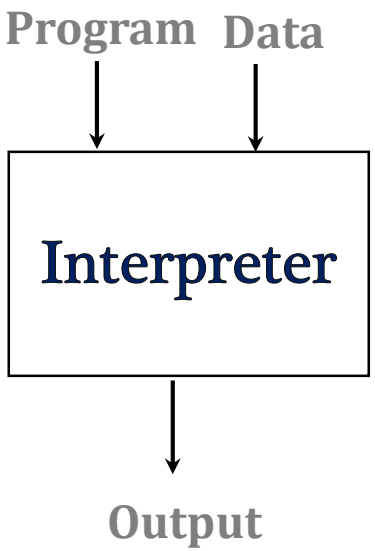
Compiler's definition

- A compiler is a software translates programs in a source language into target language.
- Offline



What is a n Interpreter

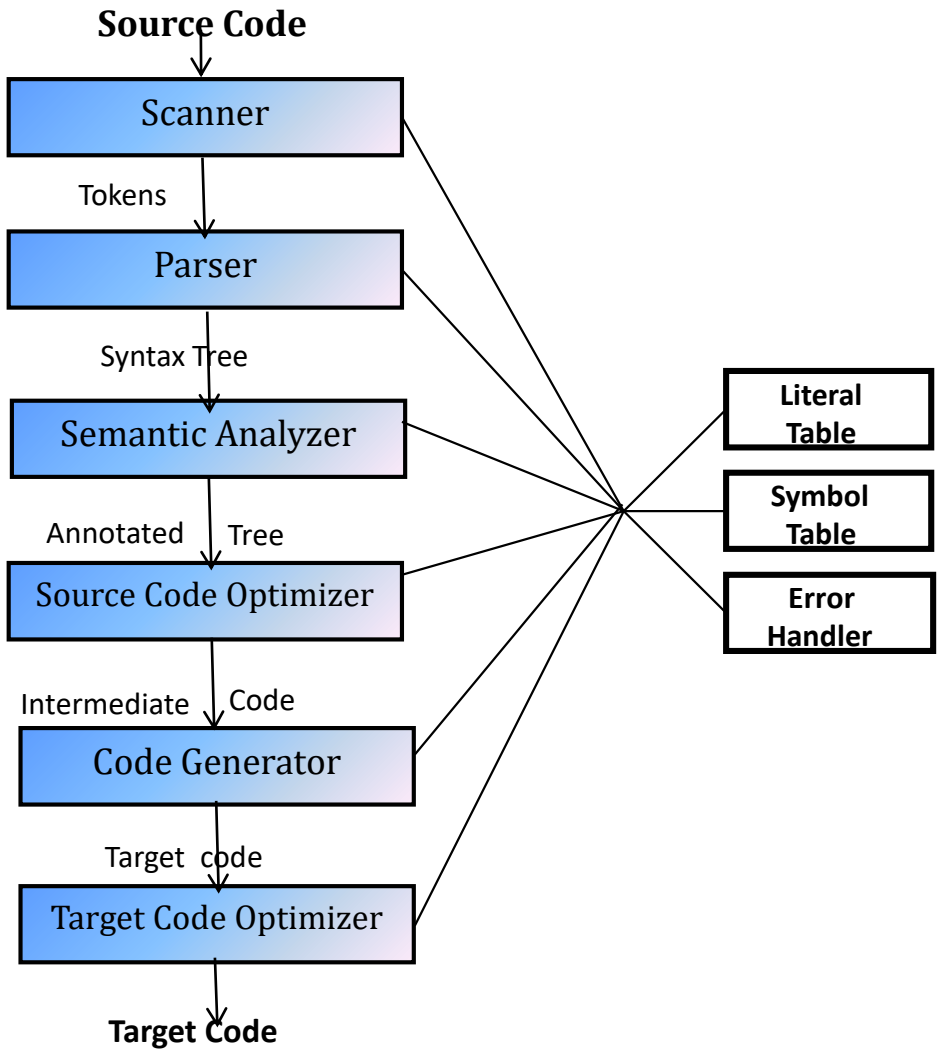
- **Unlike a compiler, an interpreter does not produce a target program. It reads an *executable* program and produces the results**
- **Online**



Hybrid Compilers

- **Source program is compiled into an intermediate program, which is later interpreted by an interpreter.**
- **For example, Java programs(*.java) are compiled into byte code(*.class), which is later interpreted by the just-in-time compiler (a virtual machine).**

Phases of the Compiler



Scanning Process

First step for humans to understand written text is recognizing Words.

This is a sentence

Ist his ase nte nce

*** Alex Hiken Stanford2014**

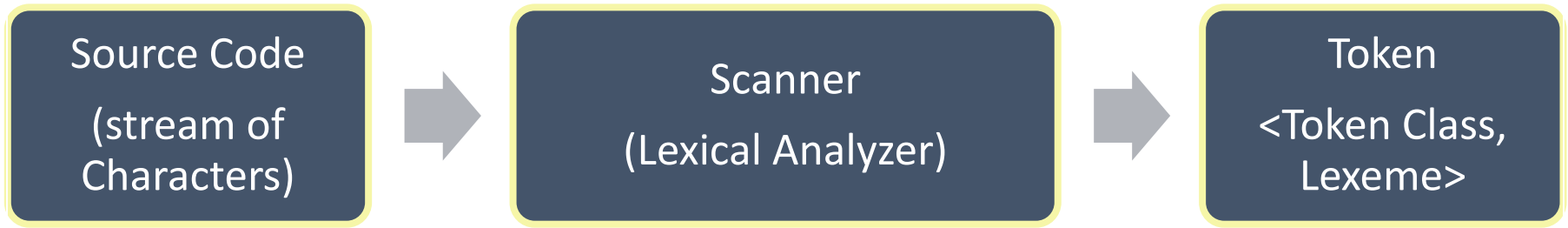
Scanning Process (Continued)

It is also helpful to identify a syntactic category

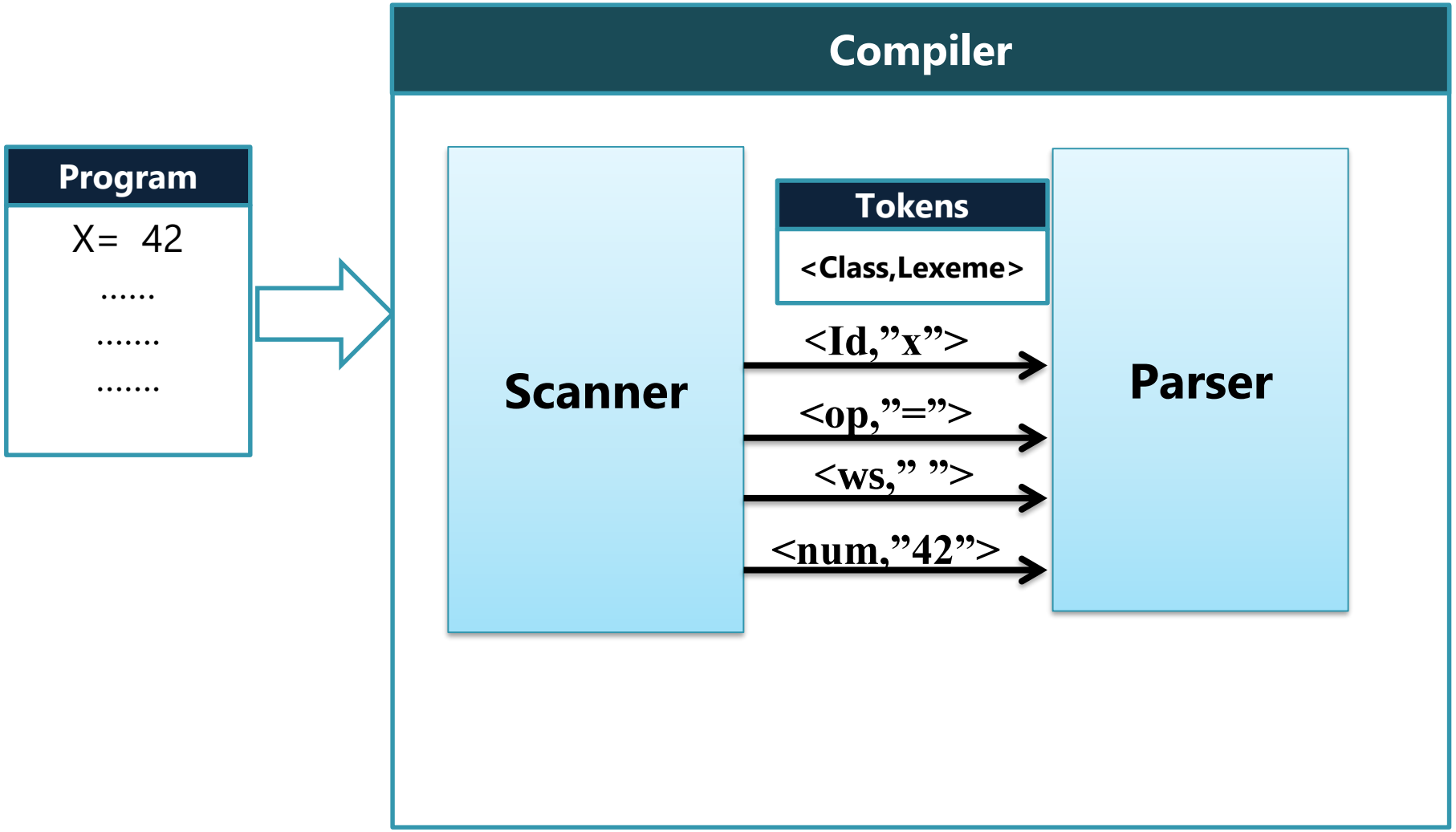
- **In English:**
  - **Noun, verb, adjective, …**
- **In a programming language:**
  - **Identifier, Integer, Keyword, White-space, …**

The Scanning Process (Continued)

- ***Lexical analyzer (scanner)*** **reads source program as a stream of characters (Character by Character from left-to-right) and classify it into tokens.**
- **Input: source code as a stream of characters (high level language program such as C++ program)**
- **Output: *Tokens* sent to the parser for syntax analysis**



The Scanning Process (Continued)



The Scanning Process

- **Source Code**
  - **float average = 273;**
- **Tokens:**
  - **<Token Class,Lexeme>**
  - **<reservedWord, float >**
  - **< identifier, average>**
  - **< assignment,“=“>**
  - **<num, 273>**
  - **< semi,“;”>**

Tokens, Lexemes and Patterns

- *Patterns* identify set of lexemes accepted by the token.

Tokens Class	Lexeme	Tokens	Pattern
Identifiers	Avg,balance, v12	<id,Avg>	Letter followed by letters and digits
Key words	If,while,for…	<kw,if>	“else” or “if” or “begin” or …
relop	<, <=,=,<>,>,>=	<relop,=>	< ,<= ,= ,<> ,> ,>=
num	31 , 28	<num,31>	a non-empty string of digits
op	+ , *, - ,/	<op,+>	Any arithmetic operator + or * or – or /
white spaces	\t ,\n	<ws,\t>	a non-empty sequence of spaces, newlines, and tabs

Implementing a scanner

- **Two primary methods for implementing a scanner:**

1) Ad hoc Scanners (Loop and Switch)

writing a program to perform the scanning tasks(suitable for small set of token types)

2) Regular Expression and Automata

Regular Expressions (RE)

Formal Languages

Regular Languages

- **Symbols** e.g. Letters,Digits,%,2,ΎΘ❶◎
- **alphabet Σ** :Finite, nonempty set of symbols.
  - A machine language Σ ={0,1}.
  - set off all English lower case character {a,b,c,d,….,z}.
- **string** :is a finite sequence of symbols over alphabet.
  - e.g. 0011001 is a string from machine language.
  - Empty string ε (not white space such as spaces or tabs).
- **Language** : set of strings over Σ
- **Powers of an Alphabet** Σ^k : the set of strings of length k from alphabet Σ. e.g.
  - Σ={0,1}.
  - Σ^1={0,1}.
  - Σ^2={00,01,10,11}.
  - Σ^*= set of all strings over alphabet Σ.
  - Σ^0={ε} empty string.

Regular Expressions' Operations

- **Alternation**

L or M = {s |s ∈ L or s ∈ M}. "|" or "+" can be used

e.g. (0|1) = 0 or 1.

e.g. a+b =a or b

- **Concatenation**

LM = {xy |x ∈ L and y ∈ M}

e.g. ab = ab

"hello" "there" = "hellothere"

- **Repetition (Kleene closure)**

$L^* = \cup_{i=0,\ldots,\infty} L^i$ → **L repeated 0 or more times**

e.g. 1* = {ε,1,11,111,1111,…….}

Regular Expressions

- **A *regular expression* is an expression that matches sets of strings.**
- **Regular expressions is used for pattern specification.**
- **Regular Expressions rules define the set of words that are valid tokens in a formal language.**
- **the "*language*" of the regular expressions is called Regular Languages.**
- **Parentheses for grouping (to change precedence, just as in arithmetic)**
- **Precedence of Operators : () ,*, concatenation, |**

Regular Expressions(Examples)

Let Σ = {a, b}

1. RE =(a|b) .

   language = {a,b} → *a or b.*

2. RE=(a|b)(a|b)

   language =*{aa, ab, ba, bb}.*

3.RE=a*

   language =*{ε, a, aa, aaa, . . . }*. all strings of zero or more a's

4. RE= (a|b)*

   language= *{ε, a, b, aa, ab, ba, bb, aaa, . . .}*. zero or more instances of a or b

5. RE= a|a*b

   language = *{a, b, ab, aab,aaab,. . .)*.all strings consisting of zero or more a's and ending in b.

Regular Expressions (Example)

a|b*

1. ε
2. a
3. b
4. bb
5. bbb

Solution : 1,2,3,4,5

Regular Expressions (Example)

Which of the following matches RE

a(ab)*a

1) **abababa**
2) **aaba**
3) **aabbaa**
4) **aba**
5) **aabababa**

Solution : 2,5

Regular Expressions (Examples)

- Examples of Regular Expressions
  - letter $\rightarrow$ A |B| ….|Z | a | b |...|z
  - digit $\rightarrow$ 0|1 |2| 3|….|9
  - digits $\rightarrow$ digit digit*
  - id $\rightarrow$ letter (letter | digit)*

Regular Expressions (Example)

- **Choose all the strings that match RE=[0-9]**
  a. 0
  b. 1
  c. 10
  d. 11
  e. 09
  f. 8
  g. ISSR

a,b,f

Regular Expressions (Examples)

- **RE =[0-9]**

  String =“1+2==3”

  Solution:

  1

  2

  3

- **RE =[a-c]**

  String =“Barbara Bosh”

  Solution:

  Barbara Bosh

Regular Expressions (Examples)

- **RE =[0-9]**

  String = “100-3=97”

- **RE=[A-Z]**

  String = “Mustafa Amin”

Solution

{1,0,0,3,9,7}

{“M”,”A”}

Regular Expressions (Examples)

- **Write all the strings in the language**

  [b-c][1-2]

- **RE=[0-9][0-9]**

  String=“Sep 15-1998”

  String =“12345”

Solution

{b1,c1,b2,c2}
{15,19,98}
{12,34}

Regular Expressions (Continued)

- **Common Extensions of R.E. operations:**
  - **+ : one or more repetitions of**

    (*r*+ is equivalent to *rr)**
  - **[] : range of characters**

    [a-d] = a|b|c|d
  - **. : any character**

    .*b.*: strings that contain at least one b.
  - **^ : negate a set**

    [^abc] = any character except a, b, or c.
  - **? : optional sub-expression**

    (+|-)?[0-9] = [0-9] | +[0-9] | -[0-9]
  - **\ : "escape" an operator or meta-symbol e.g. * \+ \? \| \.**

Regular Expressions (Continued)

- **Examples of Regular Expressions**
  - **letter →** A |B| ….|Z | a | b |...|z [A-Za-z]
  - **digit →** 0|1 |2| 3|….|9 [0-9]
  - **digits →** digit digit* digit+

Regular Expressions (Example)

Which of the following matches RE

a(bc)+

1) abc
2) abbbbbbbb
3) azc
4) abcbcbcbbc
5) ac
6) accbbbbcbcccc

Solution : 1

Regular Expression(Example)

- **RE=[a-z][0-9]**

  String="a12bcc344dX1"

- **RE=[0-9]+**

  String="13 August 1981"

  a. **{1,3,1,9,8,1}**
  b. **{13,1981}**

Solution
{a1}
{13,1981} Longest Match

Regular Expressions (Continued)

Write a regular expression for

$\Sigma = [A - Za - z0 - 9]$

1) **Positive integer percentage e.g. 9%,199% .**
2) **Quoted Strings : starts with " and ends with " and contains any number of characters except ".**
3) **All strings except EXCEPT < > or space.**

Solution

RE=[1-9][0-9]*%

RE=\"[^\"]*\"

RE=[^\s<>]*

\s-→Space