计蒜客内部课程

编译原理 - 词法分析

复习

- 编译的过程

复习

- 编译的过程
  - 词法分析
  - 语法分析
  - 语义分析
  - 代码优化
  - 目标代码生成

复习

- 编译的过程
  - 词法分析
  - 语法分析
  - 语义分析
  - 代码优化
  - 目标代码生成

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 保留字 if

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 左括号 (

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 变量 i

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 运算符 ==

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 标识符 j

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;
- 右括号)

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 标识符 z

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 运算符 =

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 整数字面量 0

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 分号；

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 保留字 else

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 标识符 z

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 运算符 =

词法分析

- if (i == j)
    z = 0;
  else
    z = 1;

- 整数字面量 1

词法分析

- if (i == j)
  z = 0;
  else
  z = 1;

- 分号；

词法分析

- 前面那个例子肉眼识别很容易

- 刚才那个例子实际上是:

词法分析

- 前面那个例子肉眼识别很容易

- 刚才那个例子实际上是:

- \tif (i == j)\n\t\tz = 0;\n\telse\n\t\tz = 1;

词法分析

- 前面那个例子肉眼识别很容易

- 刚才那个例子实际上是：

- \tif (i == j)\n\t\tz = 0;\n\telse\n\t\tz = 1;

- 词法分析的第一个工作：分词（将代码分成许多部分）

词法分析

- 前面那个例子肉眼识别很容易

- 刚才那个例子实际上是:

- \t|if| |(|i| |==| |j|)|\n\t\t|z| |=| |0|;|\n\t|else|\n\t\t|z| |=| |1|;|

- 词法分析的第一个工作：生成词素（或单词值）（将代码分成许多部分）

词法分析

- 词法分析 ≠ 生成词素

词法分析

- 词法分析 ≠ 生成词素
- 词法分析：词法单元（<token-name, attribute-value>）

词法单元类型

- 用于语法分析的词法单元的类型（token-name）

词法单元类型

- 用于语法分析的词法单元的类型（token-name）
- 自然语言中：名词、动词、形容词、副词……

词法单元类型

- 用于语法分析的词法单元的类型（token-name）

- 自然语言中：名词、动词、形容词、副词……

- 程序语言中：标识符、保留字、整数字面量、左括号、右括号……

COOL 词法单元类型

- 整数字面量:
- 标识符:
- 字符串:
- 注释:

COOL 词法单元类型

- 整数字面量：非空的由 0 到 9 组成的字符串
- 标识符：
- 字符串：
- 注释：

COOL 词法单元类型

- 整数字面量：非空的由 0 到 9 组成的字符串
- 标识符：字母开始、字母、数字或下划线组成的字符串
- 字符串：
- 注释：

COOL 词法单元类型

- 整数字面量：非空的由 0 到 9 组成的字符串
- 标识符：字母开始、字母、数字或下划线组成的字符串
- 字符串：双引号之间的字符串（有转义字符）
- 注释：

COOL 词法单元类型

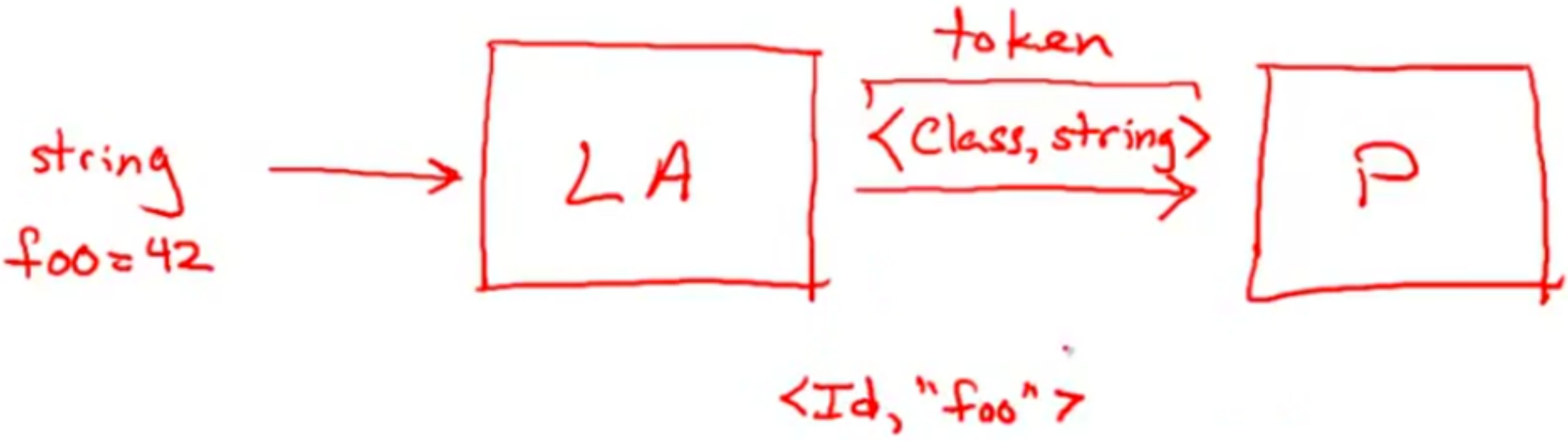
- 整数字面量：非空的由 0 到 9 组成的字符串
- 标识符：字母开始、字母、数字或下划线组成的字符串
- 字符串：双引号之间的字符串（有转义字符）
- 注释：
  - 单行：以两个'-'开始
  - 多行：在'(*'和'*)'之间

COOL 词法单元类型

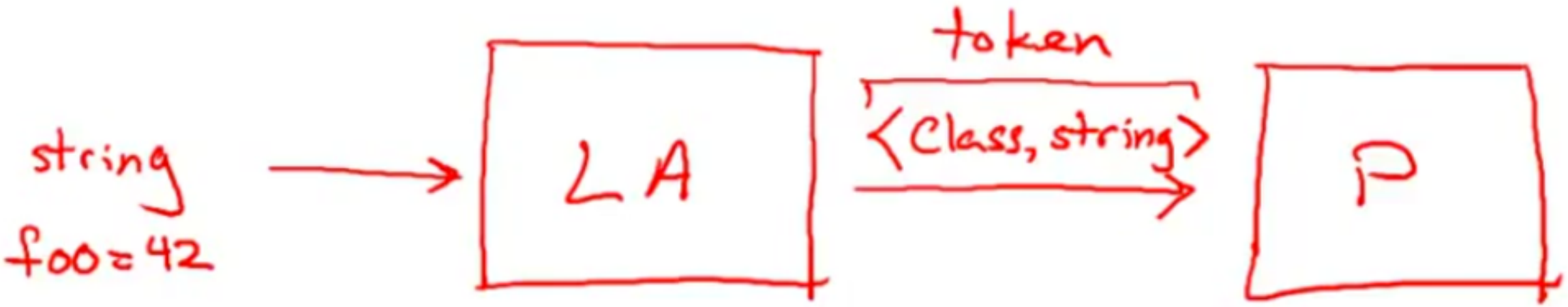
- 关键字：在保留字的集合中的词素
- 空白字符：连续的'\t'、'\n'、'\r'、' '

词法分析

词法分析

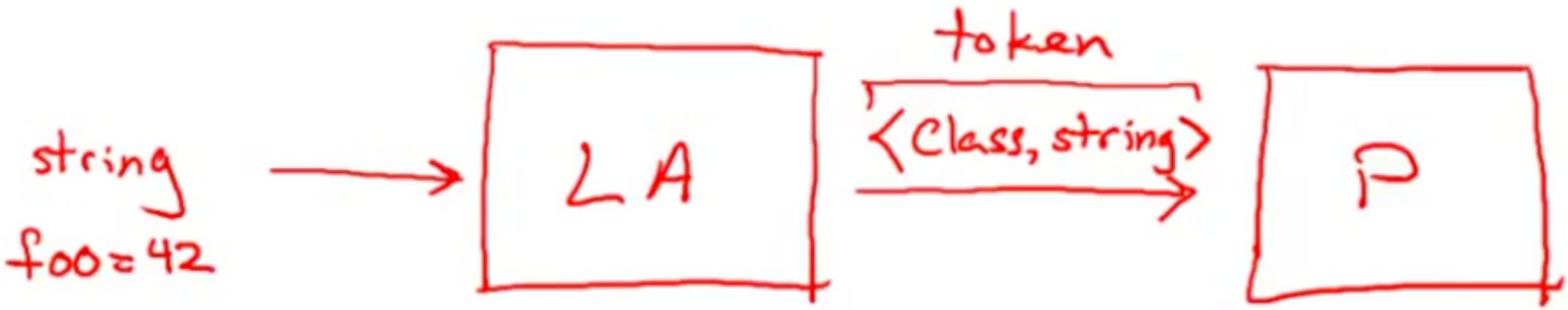


词法分析



<Id, "foo"> <Op, "="> .

词法分析



词法分析

- \tif (i == j)\n\t\tz = 0;\n\telse\n\t\tz = 1;

- Operators
  Whitespace
  Keywords
  Identifiers
  Numbers

词法分析

- x = 0;\n\twhile (x < 10) {\n\tx++;\n}

- Operators
  Whitespace
  Keywords
  Identifiers
  Numbers

词法分析

- 总结：

- 1. 将字符串分割为若干个“词素”（lexemes）

- 2. 识别每个“词素”的类型，生成词法单元序列

词法分析

- 总结：

- 1. 将字符串分割为若干个“词素”（lexemes）

- 2. 识别每个“词素”的类型，生成词法单元序列

- 从左向右顺次找就可以了？

PL/I

- DECLARE(ARG1, ARG2, …

- 可以是声明若干个变量，也可以是一个函数名

- 怎么处理?

PL/I

- DECLARE(ARG1, ARG2, ...

- 可以是声明若干个变量，也可以是一个函数名

- 怎么处理?

- lookahead

PL/I

- DECLARE(ARG1, ARG2, ...

- 可以是声明若干个变量，也可以是一个函数名

- 怎么处理?
- lookahead
- left-to-right scan => lookahead sometimes required

C++

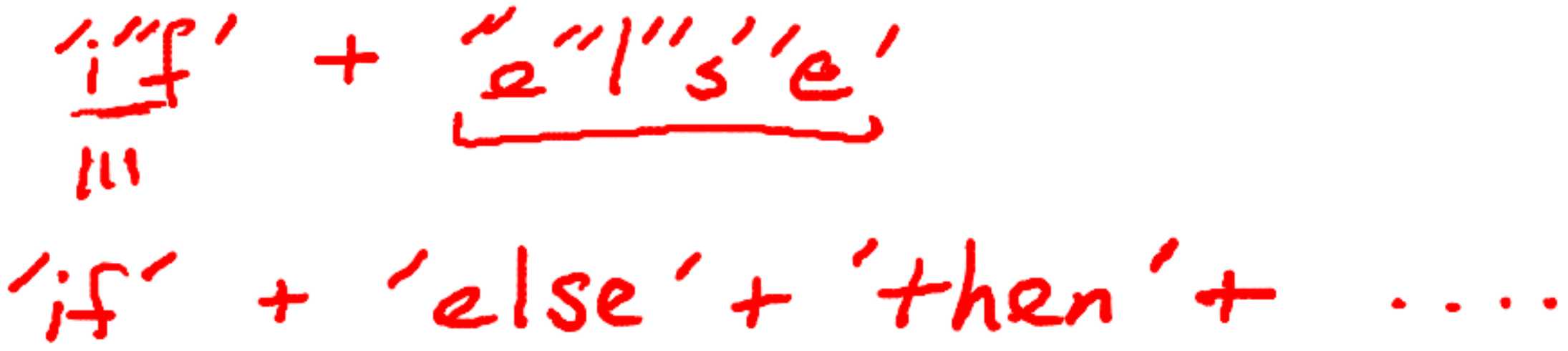
- cin >> a;
- vector<vector<int>> a;
- '>>'

词法分析

- 识别关键字：“if”或“else”或“then”
- 怎么用正则语言表示?

词法分析

- 识别关键字：“if”或“else”或“then”
- 怎么用正则语言表示?



词法分析

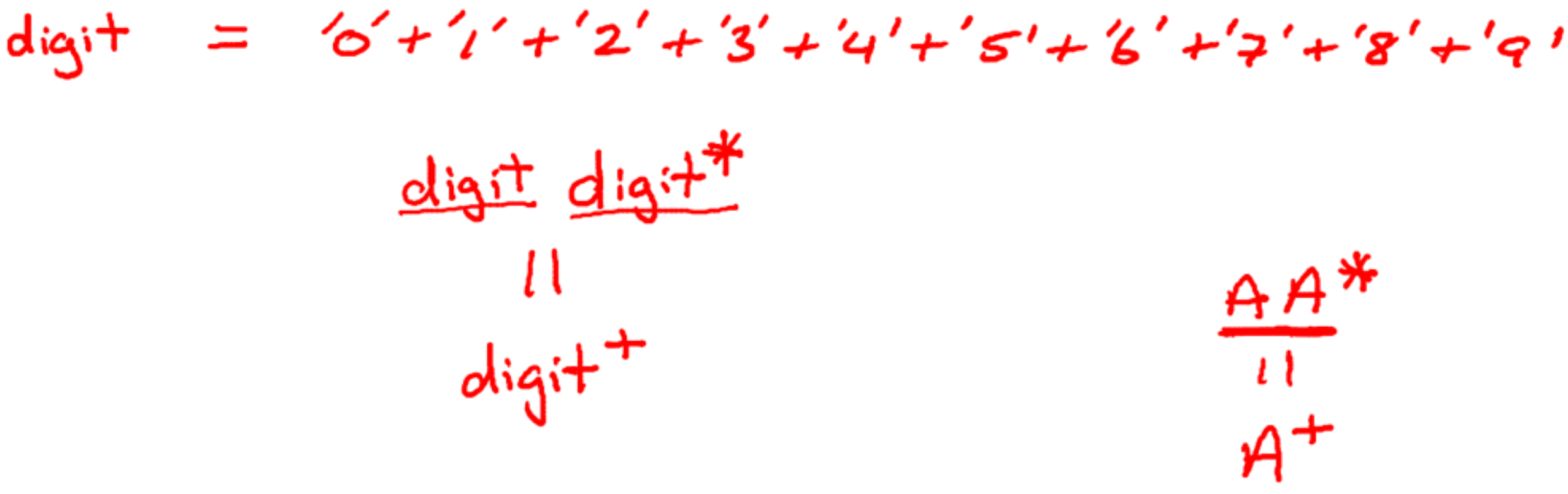
- 识别整数字面量：一个非空的数字串

- 怎么用正则语言表示?

词法分析

- 识别整数字面量：一个非空的数字串

- 怎么用正则语言表示?



词法分析

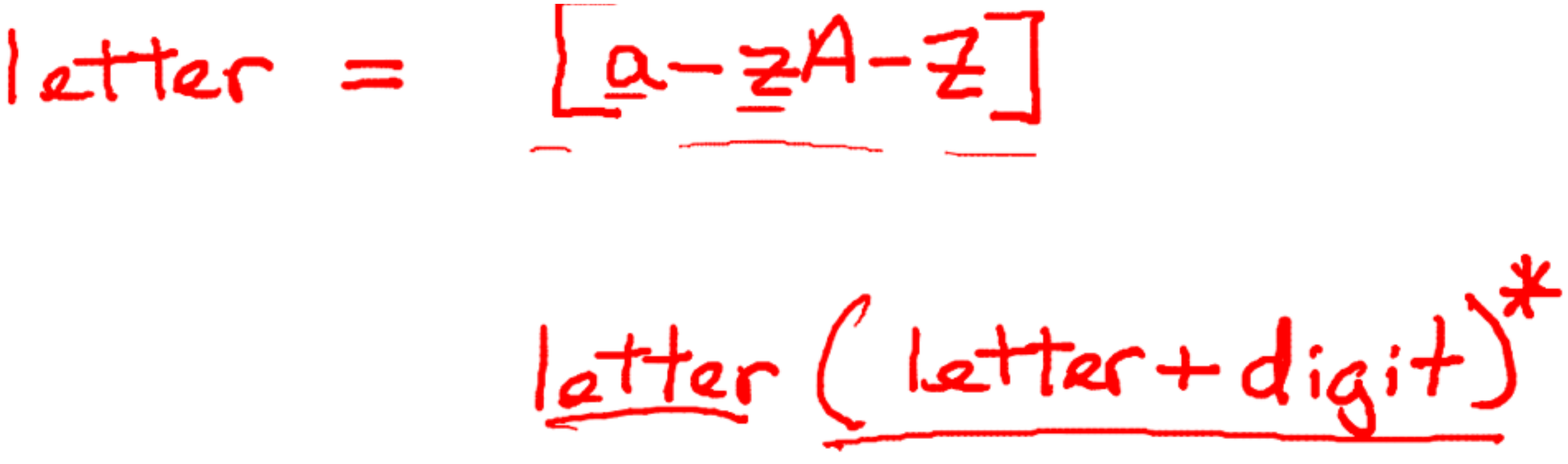
- 识别标识符：以字母开始，包含字母和数字

- 怎么用正则语言表示?

词法分析

- 识别标识符：以字母开始，包含字母和数字

- 怎么用正则语言表示?



词法分析

- 识别空白字符：一系列空格、换行、缩进

- 怎么用正则语言表示?

词法分析

- 识别空白字符：一系列空格、换行、缩进

- 怎么用正则语言表示?

正则表达式

- At least one: A+
- Union: A | B
- Range: [a-z]
- Excluded Range: [^a-z]

词法分析的算法流程

1. 写出每种词法单元类型的正则

- Number = digit+

1. 写出每种词法单元类型的正则

- Number = digit+
- Keyword = ‘if’ + ‘else’ + …

1. 写出每种词法单元类型的正则

- Number = digit+
- Keyword = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*

1. 写出每种词法单元类型的正则

- Number = digit+
- Keyword = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*
- OpenPar = ‘(‘

1. 写出每种词法单元类型的正则

- Number = digit+
- Keyword = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*
- OpenPar = ‘(‘
- …

2. 构造一个语言 R，匹配所有词法单元类型

- R = Keyword + Identifier + Number + ...

CONTINUE…

3. Let input be $x_1 \ldots x_n$

   For $1 \leq i \leq n$ check

   $x_1 \ldots x_i \in L(R)$

4. If success, then we know that

   $x_1 \ldots x_i \in L(R_j)$ for some j

5. Remove $x_1 \ldots x_i$ from input and go to (3)

最大适合规则（MAXIMAL MUNCH）

- if (a == b)
- == 怎么解析?

最大适合规则（MAXIMAL MUNCH）

- if (a == b)
- == 怎么解析?
- 贪心：最长的那个规则

最大适合规则（MAXIMAL MUNCH）

- if (a == b)
- == 怎么解析?
- 贪心：最长的那个规则
- 如果长度一样呢?

继续...

- Keywords = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*

继续...

- Keywords = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*
- ‘if’ 解析为?

HIGHEST PRIORITY MATCH

- Keywords = ‘if’ + ‘else’ + …
- Identifier = letter(letter + digit)*
- ‘if’ 解析为?
- 优先级（先被列出的规则优先级更高）

不合法 TOKEN

- 如果遇到不属于 R 语言的字符串，怎么处理?

不合法 TOKEN

- 如果遇到不属于 R 语言的字符串，怎么处理?
- 将所有不合法字符串的情况写作最后一条规则（优先级最低）。

总结

- 解决歧义的两种方法：

总结

- 解决歧义的两种方法：
  - matches as long as possible
  - highest priority match

总结

- 解决歧义的两种方法：
  - matches as long as possible
  - highest priority match
- 错误处理

总结

- 解决歧义的两种方法：
  - matches as long as possible
  - highest priority match
- 错误处理
- 优秀的词法分析器：只需要一遍扫描，更少的运算次数