

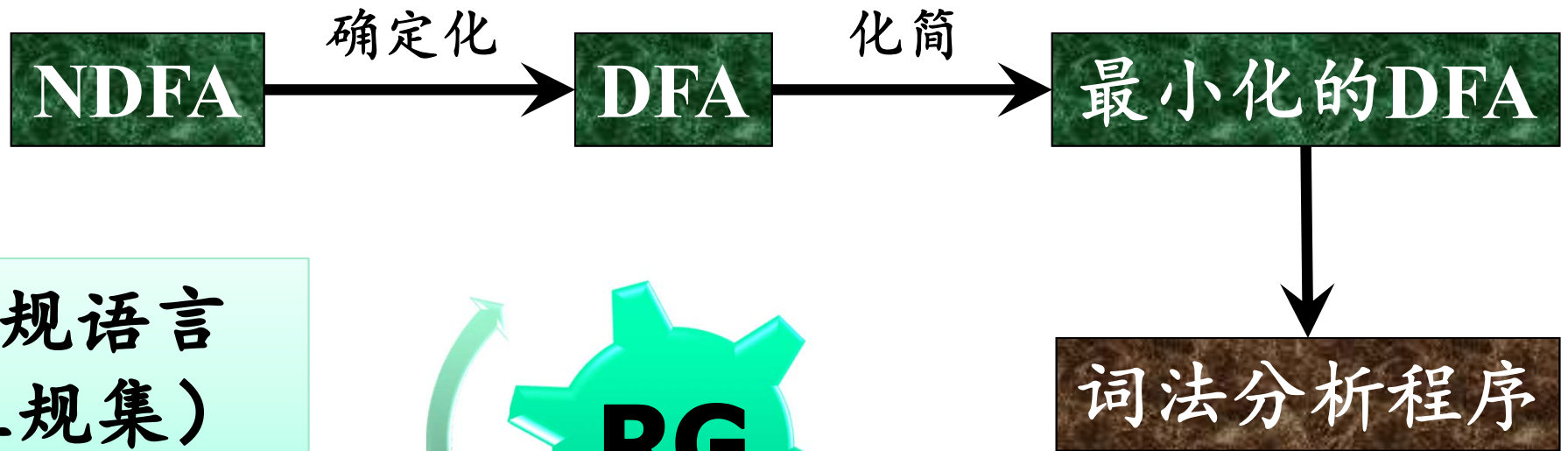


# 编译原理

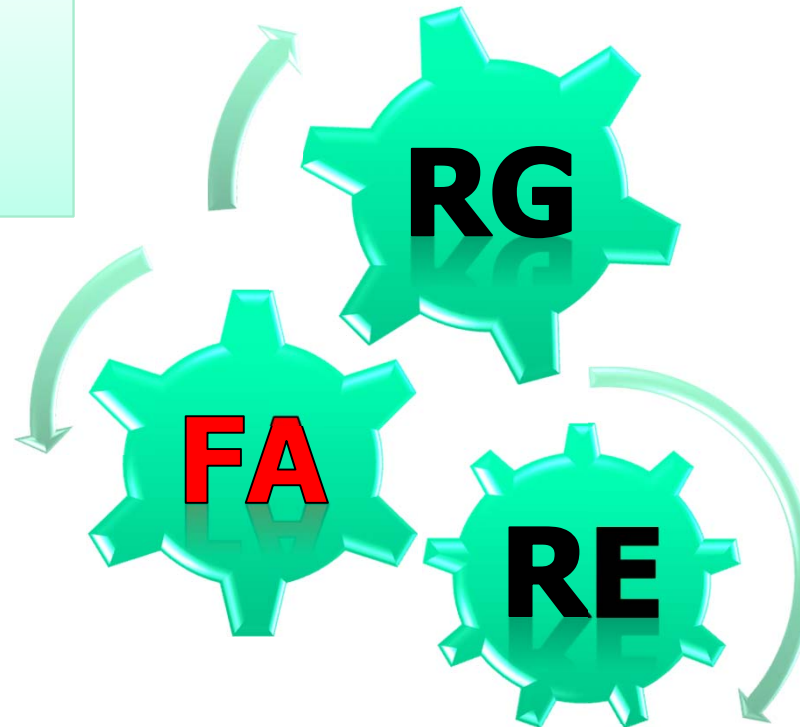
---

武汉大学计算机学院  
编译原理课程组

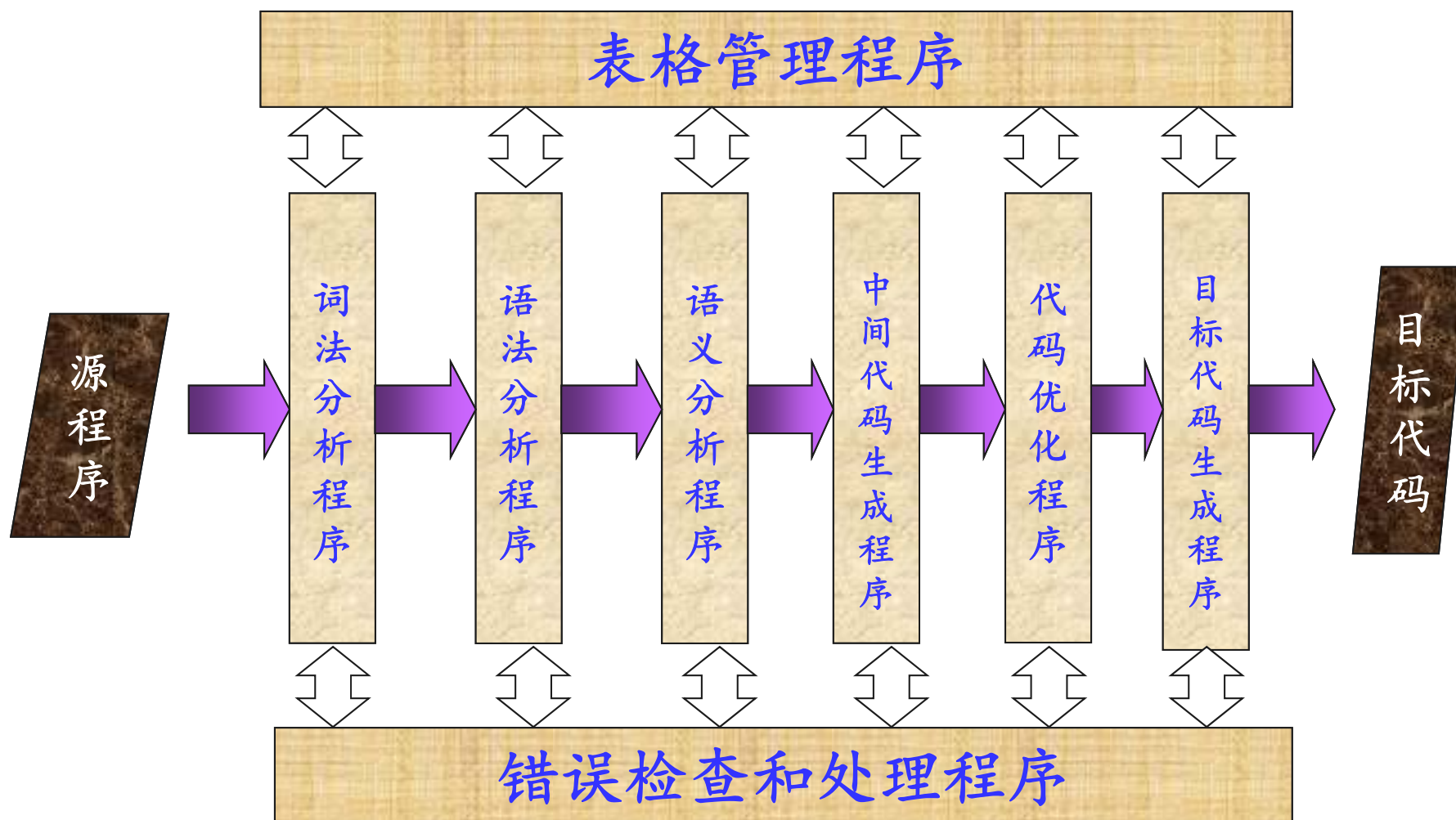
# 前述内容回顾



正规语言  
(正规集)



# 编译程序的结构





## 第4章 词法分析

---

- 1 词法分析器
- 2 单词符号
- 3 词法分析程序设计
- 4 词法分析器的自动生成



# 词法分析

属性字: <类别号, 自身值>  
class value

符号表

position := initial + rate \* 60

Scanner

<id,1> <:=> <id,2> <+> <id,3> <\*> <number,4>

	NAME	TYPE	VAL	ADDR	...
1	position	...			
2	initial	...			
3	rate	...			
4	60	...			



## 4.1 词法分析器与单词符号

### 1. 词法分析程序的作用

词法分析程序依据语言**词法规则**，分析由字符组成的源程序，把它识别为一个一个具有独立意义的最小语法单位，即“**单词**”，并识别出与其相关的属性(如是标识符，是界限符，还是数，等等)，再转换成长度上统一的标准形式——**属性字**，把字符串形式的源程序改造成成为单词符号串(属性字)形式的**中间程序**，以供其它部分使用。

如删除注解、空格、回车符、换行符之类非必要信息等**预加工**处理，以及把标识符登录入符号表等语义处理工作。



## 4.1 词法分析器与单词符号

---

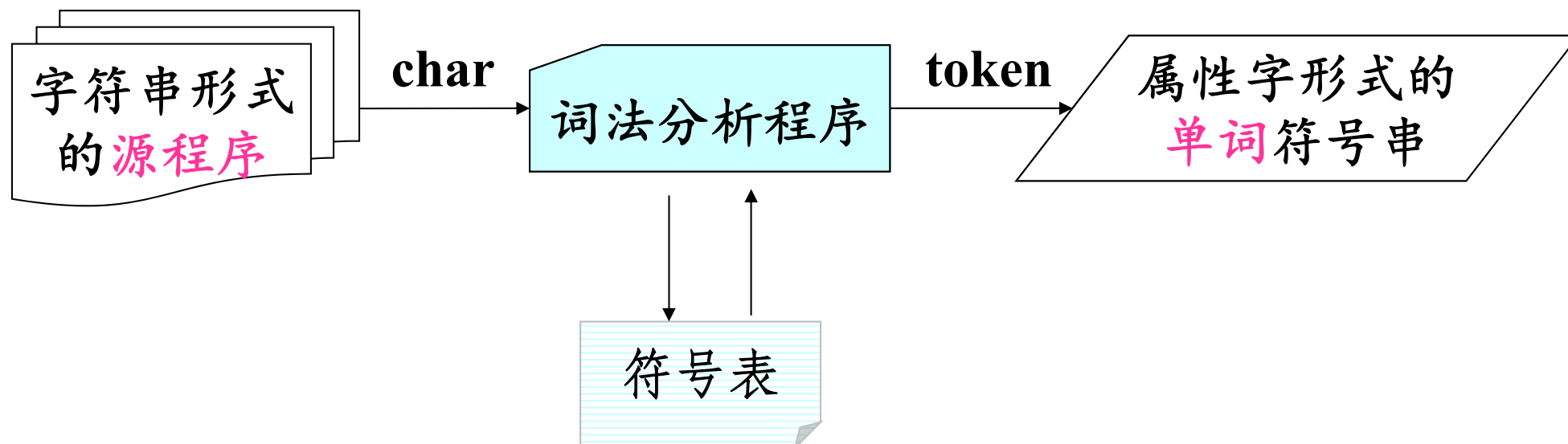
### 2. 词法分析程序的两种处理结构

- 作为主程序，单独一趟
- 作为子程序，供Parser调用

## 4.1 词法分析器与单词符号

### 2. 词法分析程序的两种处理结构

□ 作为主程序，单独一趟







## 4.1 词法分析器与单词符号

---

### 2. 词法分析程序的两种处理结构

□ 作为主程序，单独一趟的特点

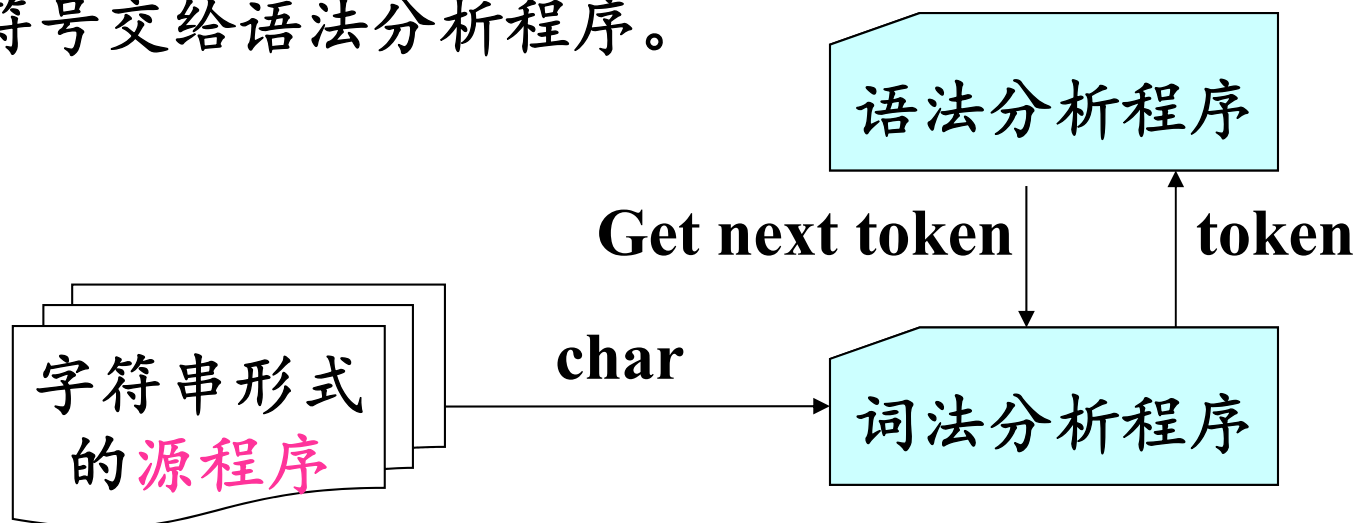
- 降低Parser的复杂性，使Compiler结构清晰、简洁
- 转换成统一的属性字形式的单词串，增强Compiler的可移植性
- 可分开构造Scanner generator / Parser generator
- 增加了属性字序列存储、内外存调入调出等重复工作

## 4.1 词法分析器与单词符号

### 2. 词法分析程序的两种处理结构

□ 作为子程序，供Parser调用

每当语法分析程序需要一个单词符号时就调用词法分析子程序，每一次调用，词法分析子程序就从源程序中识别出一个单词符号交给语法分析程序。





## 4.1 词法分析器与单词符号

### 3. 单词符号

#### (1) 类别

关键字、标识符、常数、运算符、界限符

#### (2) 表示形式

类别号	属性值
-----	-----

- 类别号

一类一码；一符一码。

- 属性值（自身值）

反映单词符号特征或特性的值。（内码/指针）  
如标识符的符号表指针，常数的常数表指针等。

# 属性字的表示形式

<类别号, 自身值>  
class value

符号表

标识符 **position**

21	10
----	----

10

14

18

NAME	TYPE	VAL	ADDR	...
position	...			
initial	...			
rate	...			
	...			

实常数 **3.14**

22	100
----	-----

100

104

108

实常数 **0.005**

22	104
----	-----

$0.314 \times 10$
$0.5 \times 10^{-2}$
...

常数表



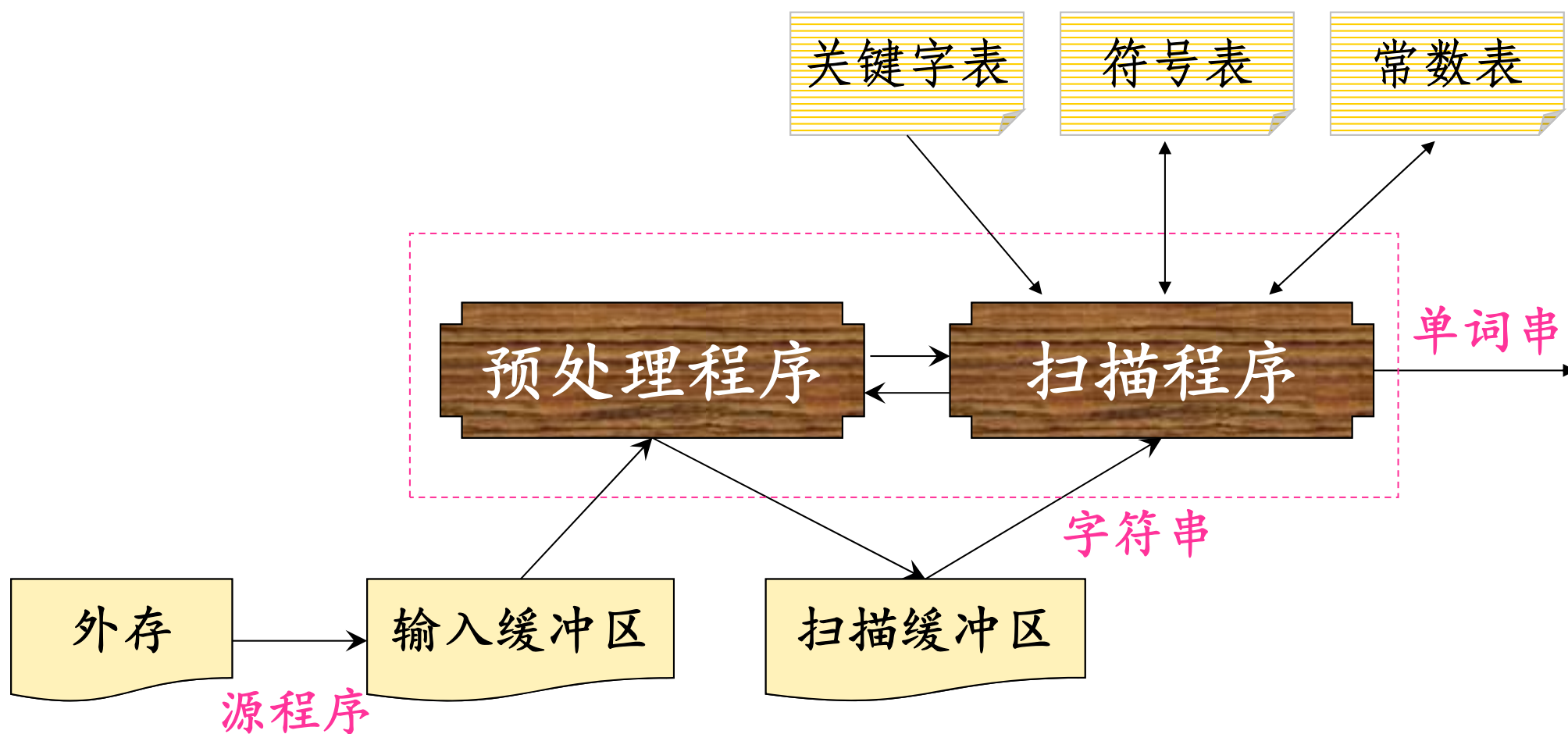
## 部分语义处理工作（并非必要的）

---

- 查填符号表
- 部分语义检查

如对标识符的处理：

- 定义性标识符
- 使用性标识符

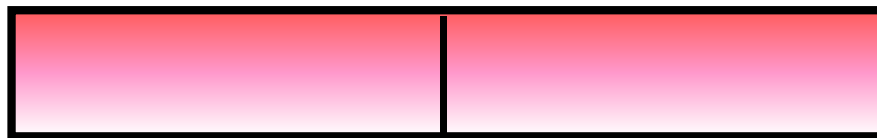


## 4.2 词法分析程序的设计

### 1. 预处理

如删除注解、空格、回车符、换行符之类**非必要信息**。

从源程序中处理出一串确定长度的输入字符，并将其装进词法分析程序指定的缓冲区——**扫描缓冲区**中。



扫描缓冲区

## 4.2 词法分析程序的设计

### 2. 单词符号的识别 —— 超前搜索

- 关键字的识别
- 标识符的识别
- 常数的识别
- 运算符/界限符的识别

IF(5.EQ.M)I=10

IF=100

IF(5)=55

5.EQ.M

5.E08

<>

<=

<

逻辑条件语句

简单变量赋值语句

下标变量赋值语句

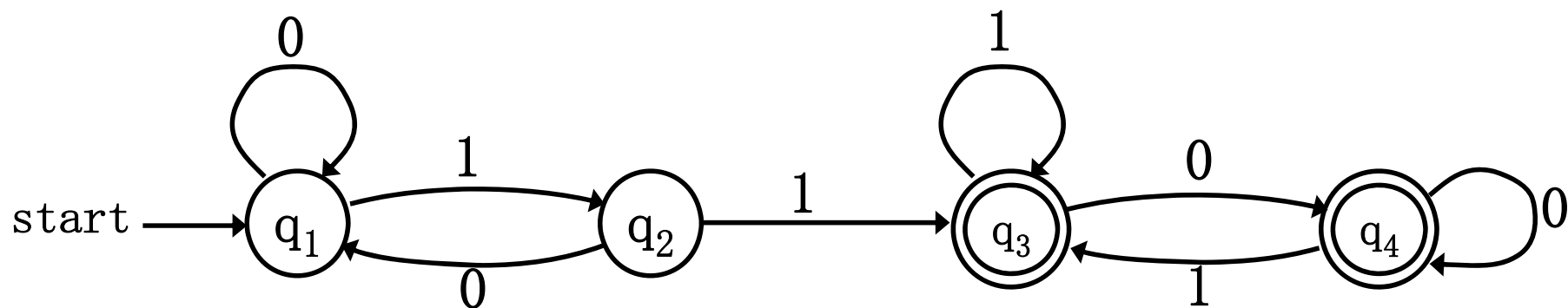
数

最长匹配原则  
最先匹配原则



## 4.2 词法分析程序的设计

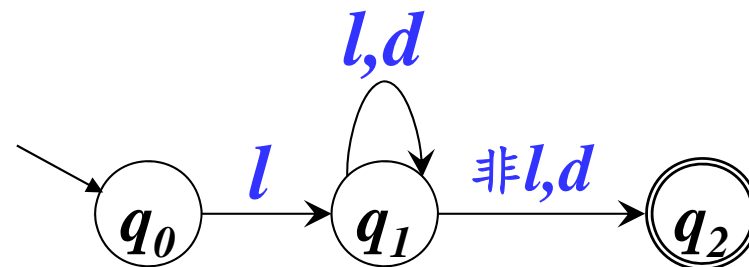
### 3. 词法分析程序的设计与实现（状态转换图）



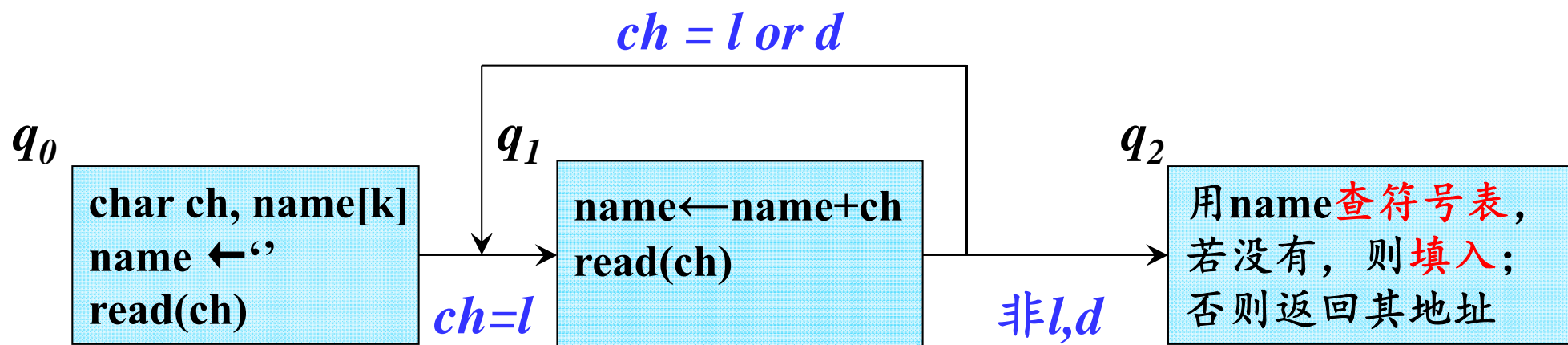
状态转换图中的每一个状态实际上对应着分析过程的某一个时刻。

**Scanner 的主要工作：实现状态转换**（同时执行相应的语义动作）

## 4.2 词法分析程序的设计

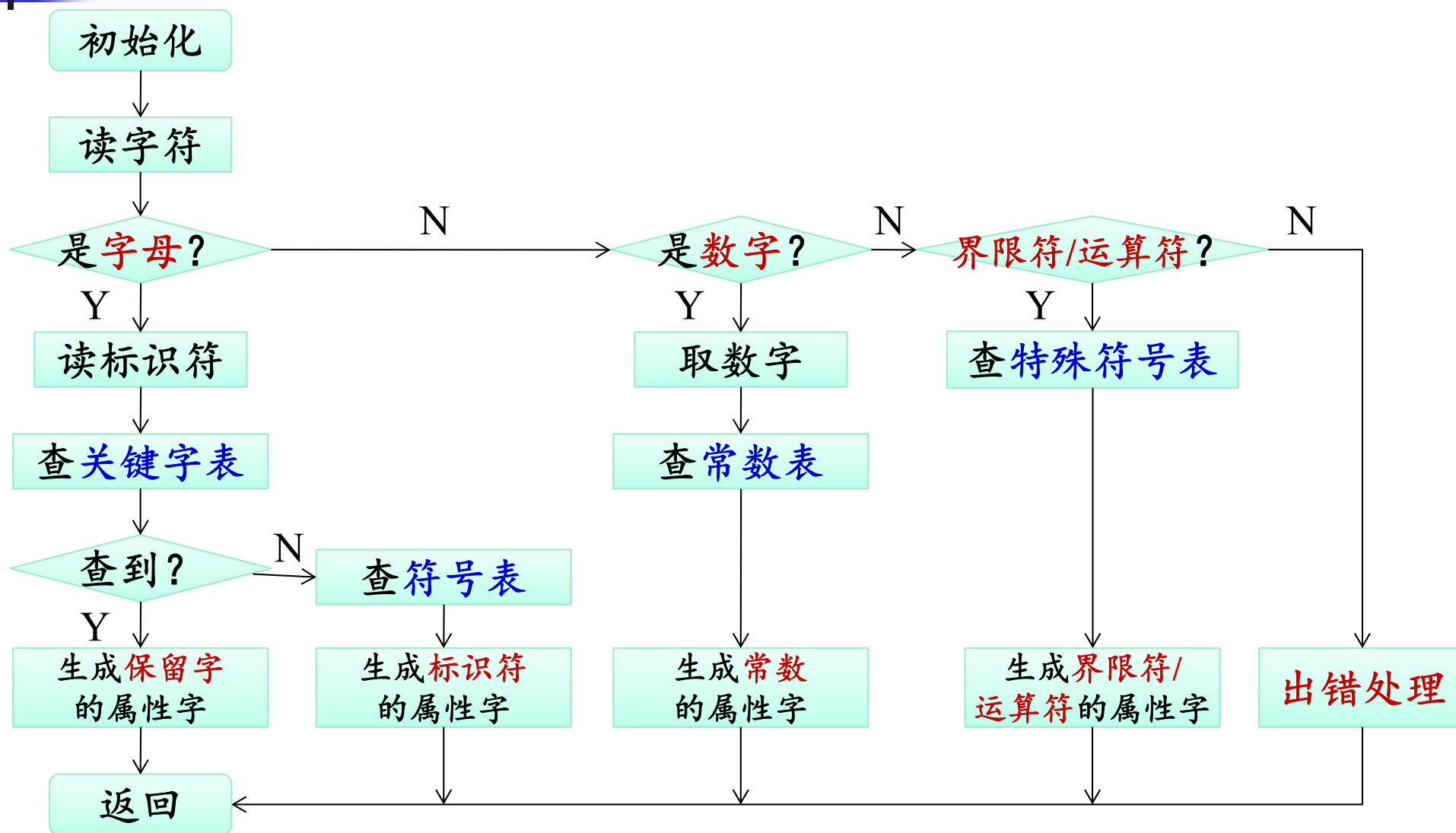


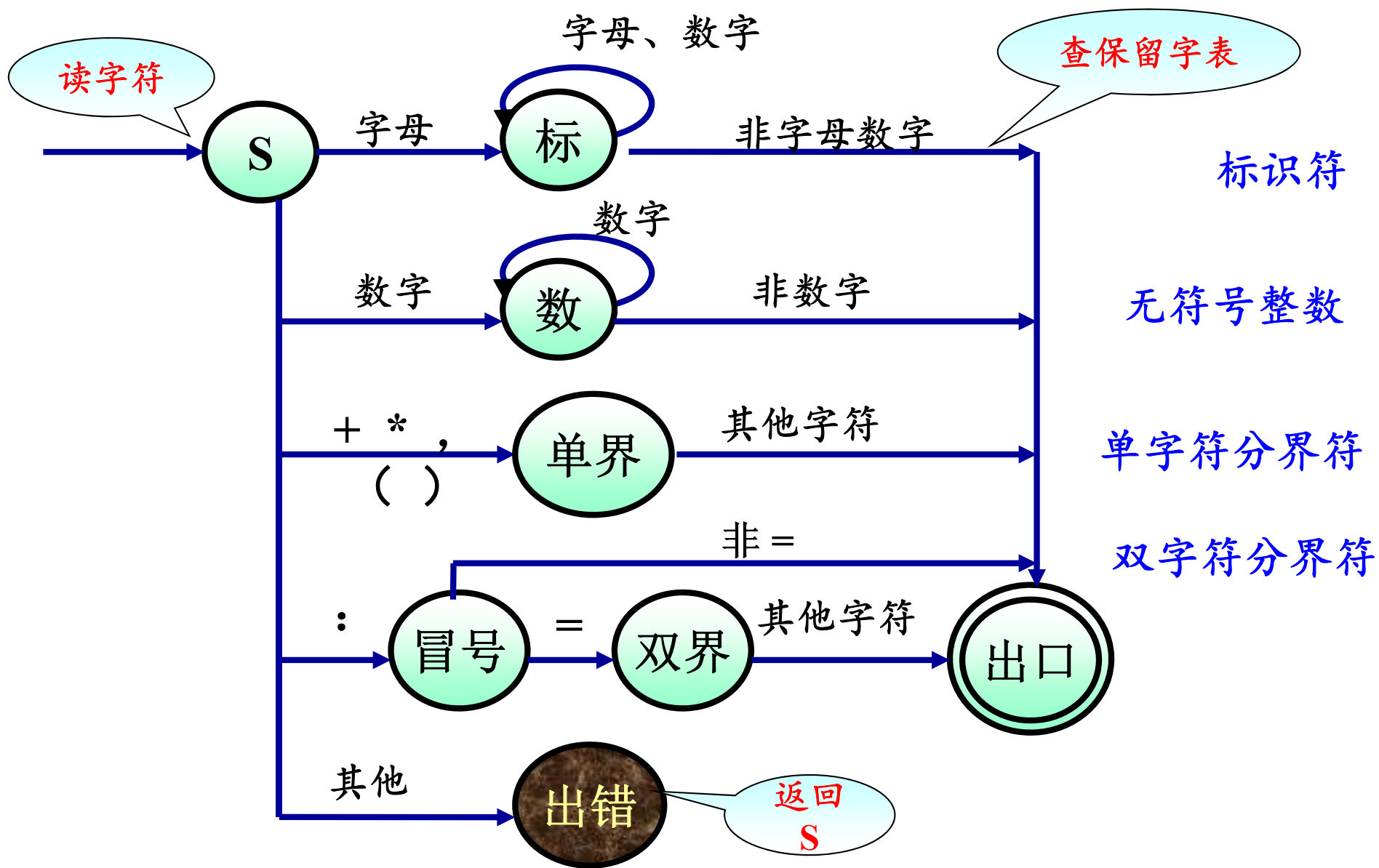
标识符DFA



识别标识符的程序流程图

# 单独成趟的词法分析程序的流程图

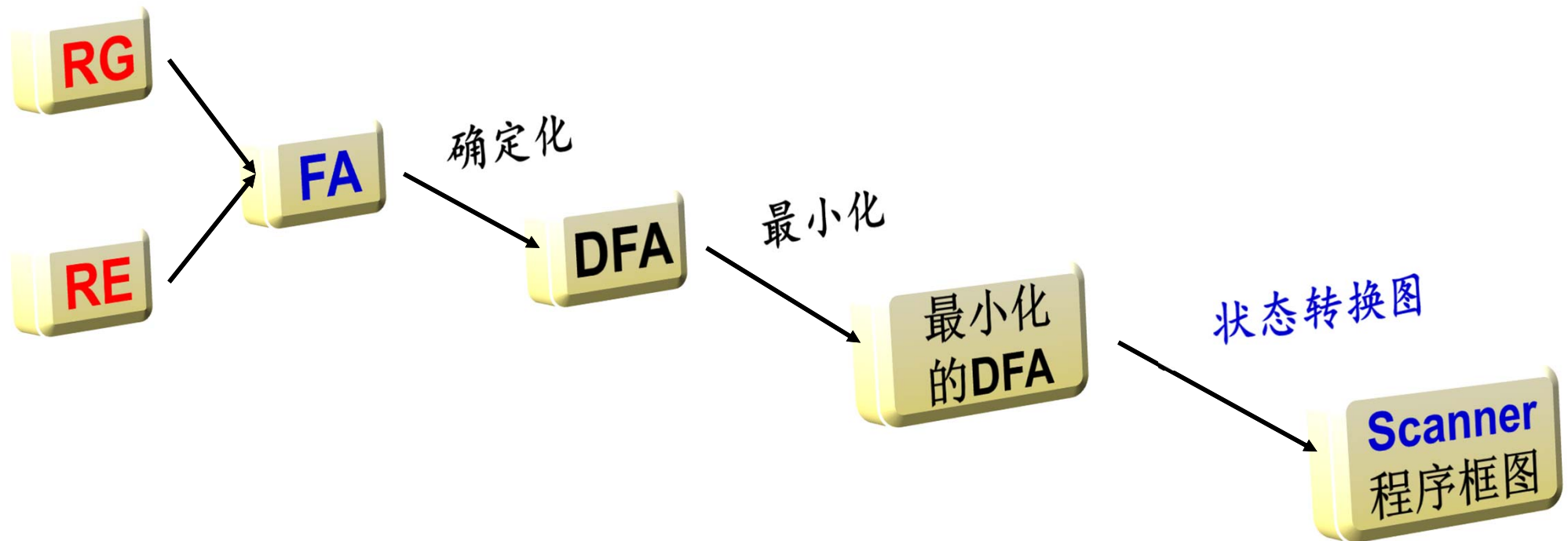




识别各类单词符号的FA的状态转换图

## 4.2 词法分析程序的设计

### 3. 词法分析程序的设计与实现（状态转换图）



通过两种方式设计



## 4.2 词法分析程序的设计

**状态转换图的实现：**将状态转换图看作是词法分析程序的程序框图。

每个**状态**构造一段**代码**，代码的功能为：

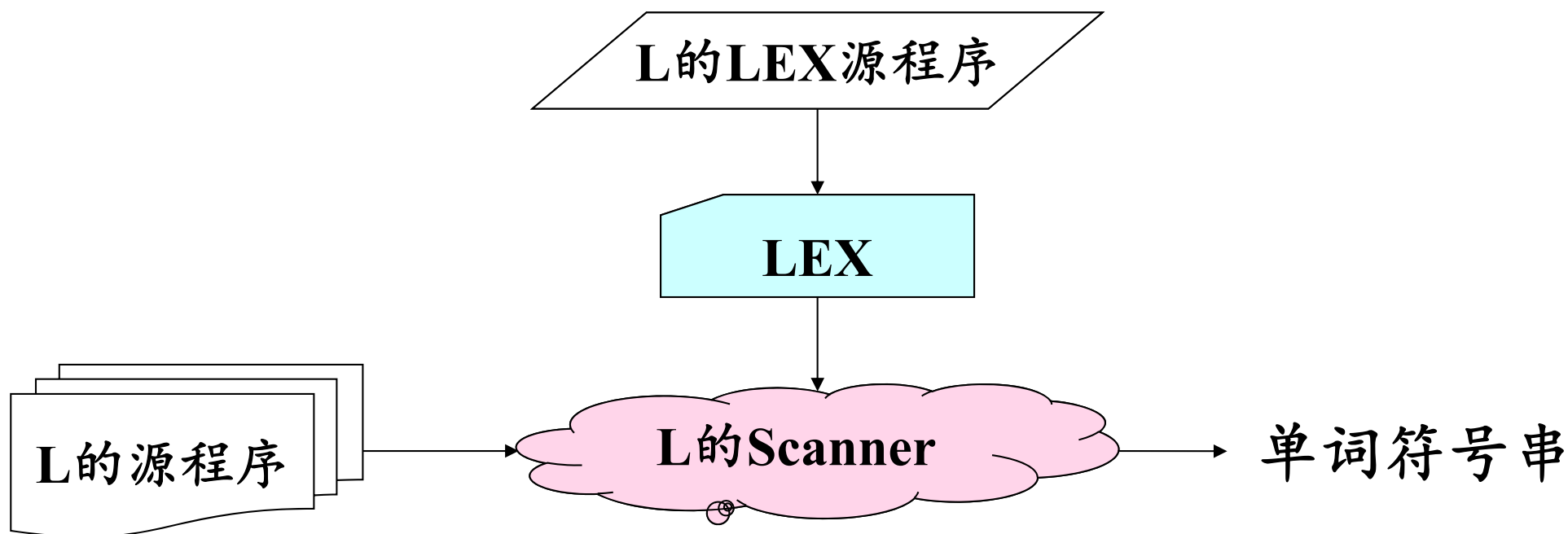
- ①从输入串中**读一个字符**；
- ②判明读入的字符与由此状态出发的哪条弧上的标记相匹配，  
便**转**至相**匹配**的那条弧所指向的状态；
- ③均不匹配时便**失败**(不能到达正常出口)。

有些词法分析程序还会在这些代码段中加上一些其它**语义处理**（如对数值进行10进制到2进制的转换等）。



## 4.3 词法分析程序自动生成

LEX是一个词法分析程序生成器。





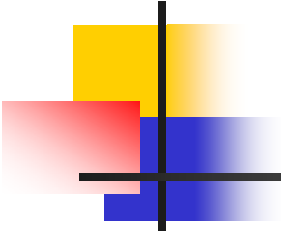
# LEX语句形式

LEX接受一组正规表达式以及每个正规表达式相应的一组动作。

“动作”本身是一小段程序代码，它指出了按正规表达式识别出一个单词后应采取的动作。

LEX语句形式:	正规式	语义规则
	$P_1$	$\{A_1\}$
	$P_2$	$\{A_2\}$
	...	...
	$P_n$	$\{A_n\}$





# LEX语句形式

LEX语句形式:

正规式

语义规则

and	{RETURN(0,-)}
begin	{RETURN(1,-)}
...	...
write	{RETURN(20,-)}
letter{letter digit}	{RETURN(21,TOKEN)}
digit{digit}	{RETURN(22,DTB)}
:	{RETURN(25,-)}
...	...
+	{RETURN(34,-)}
-	{RETURN(35,-)}
...	...
:=	{RETURN(44,-)}



# LEX的构造思想

---

构造步骤为：

①对token按类构造状态转换图；

②增加初态，合并状态转换图；

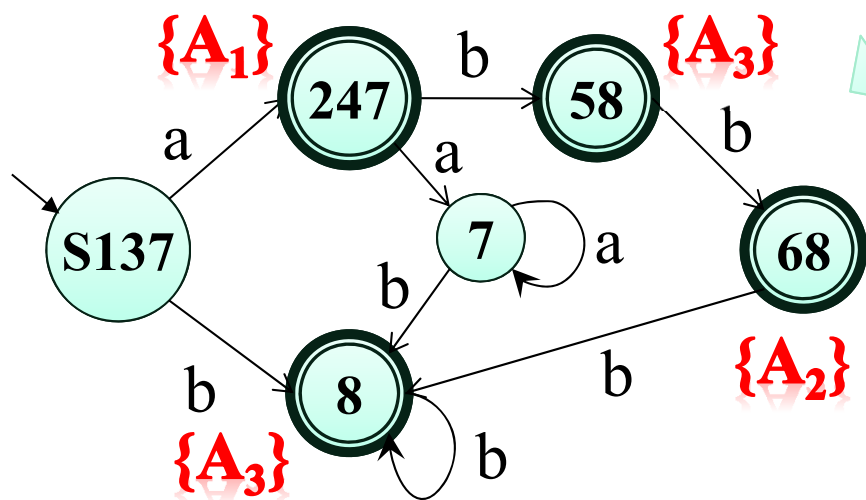
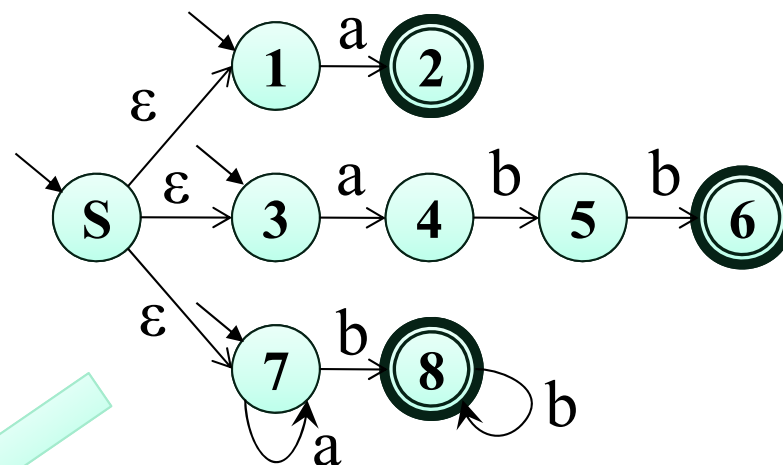
③化简状态转换图；

（注意：DFA最小化为MFA时，等价状态的语义动作也必须相同）

④增加出错处理终态。

# LEX的构造思想

a                     $\{A_1\}$   
 abb                 $\{A_2\}$   
 a\*bb\*             $\{A_3\}$



最长匹配原则  
 最先匹配原则

举例：输入串的认可过程

abb... 68  $\{A_2\}$

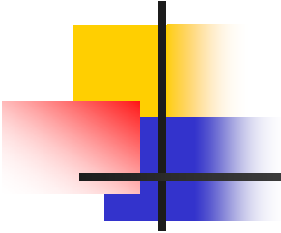
aac... 247  $\{A_1\}$



## 第4章 内容小结

---

- 词法分析程序的任务和作用
- 单词符号的表示
- 词法分析程序的构造



# 实 习

---

实习题：构造一个小（Mini）语言的词法分析程序。

设计一个包含简单算术表达式、赋值语句、IF语句的小语言的文法。

根据此文法，构造一词法分析程序。输入以“#”为结束符的源程序，输出为各类单词表和单词串文件。

要求：源程序和输出的单词串均以文件的形式存放。单词的自身值均为其对应的表的指针，如标识符表的指针、常数表的指针等。

词法错误类型：词法中未定义的字符及任何不符合词法单元定义的字符。