第二章　词法分析

词法分析的基本功能

正则表达式

PART 1

词法分析的基本功能

词法分析的基本功能

词法分析程序是编译程序的一部分，是整个编译过程的第一步工作。

词法分析器 读取源程序的字符序列，逐个拼出单词并构造相应的内部表示。同时检查源程序中的词法错误。它的核心作用即为将字符序列转化为计算机内部表示。

抽取单词序列的例子 

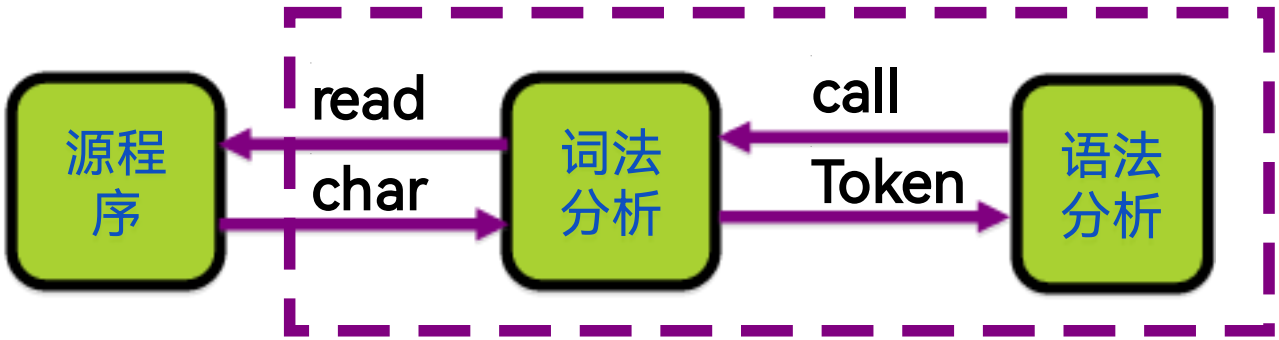
```
if (position > 10)  rate = 3.14 * initial;
```


<$if,->,<$open,->,<$id,position>,<$gt, ->,
<$num, 10>,<$close, ->, <$id, rate>,<$eq,->,
<$num, 3.14>,<$mult,->,<$id, initial>, <$semi,
->

词法分析器的接口

词法分析器有两类

一类是仅作为语法分析的子程序：



另一类是作为编译器的独立一遍处理器：

单词及单词类型的划分

单词

是指语言中具有独立含义的最小的语义单位。

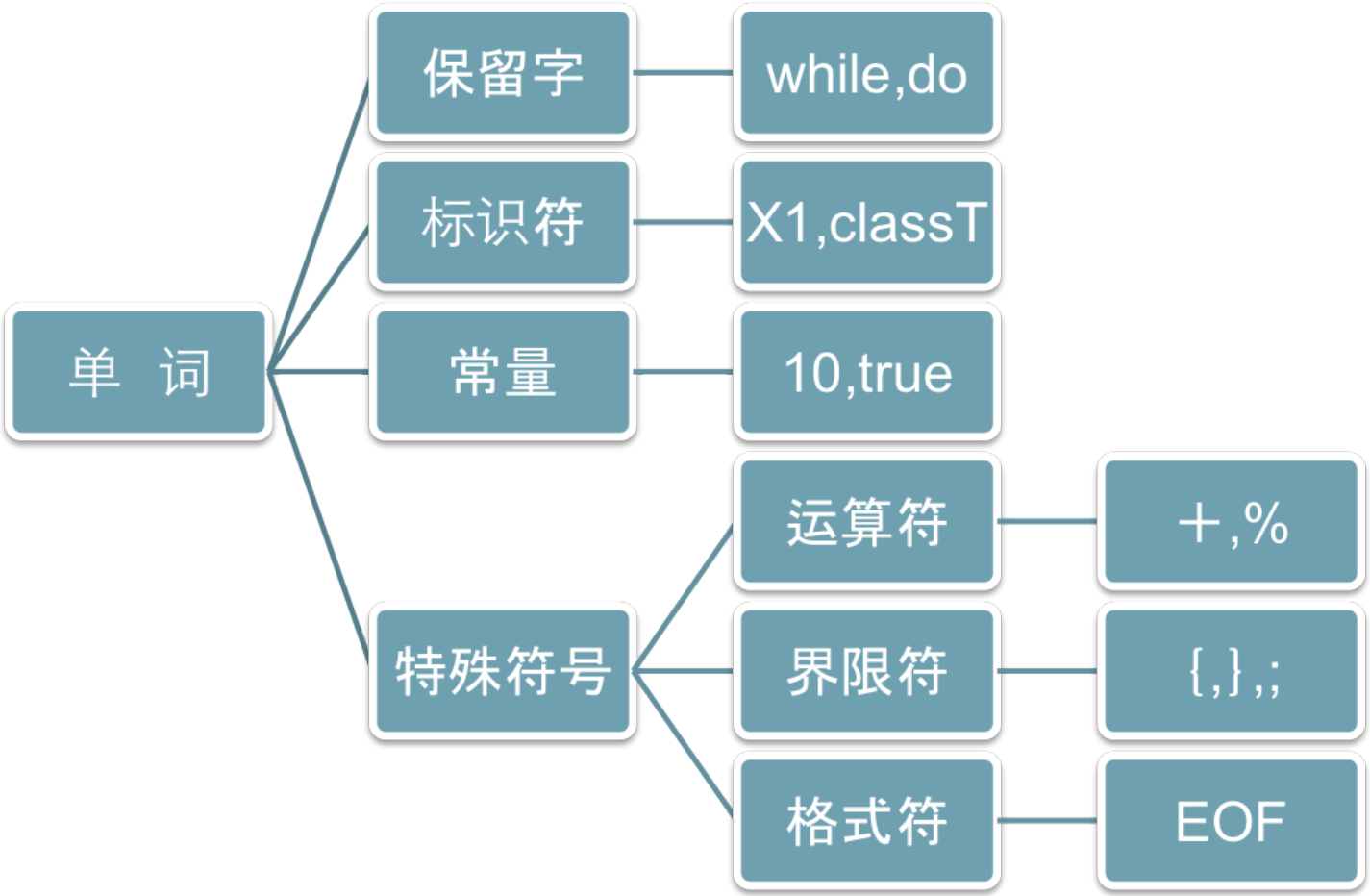
```
if (position > 10)  rate = 3.14 * initial;
```

例如 3.14*initial就可以划分成:

3.14, *,initial这三个单词。

但是3.14不可以继续划分成3,.,14

单词及单词类型的划分



思考如何实现词法分析

你怎么做?

怎么做合理?

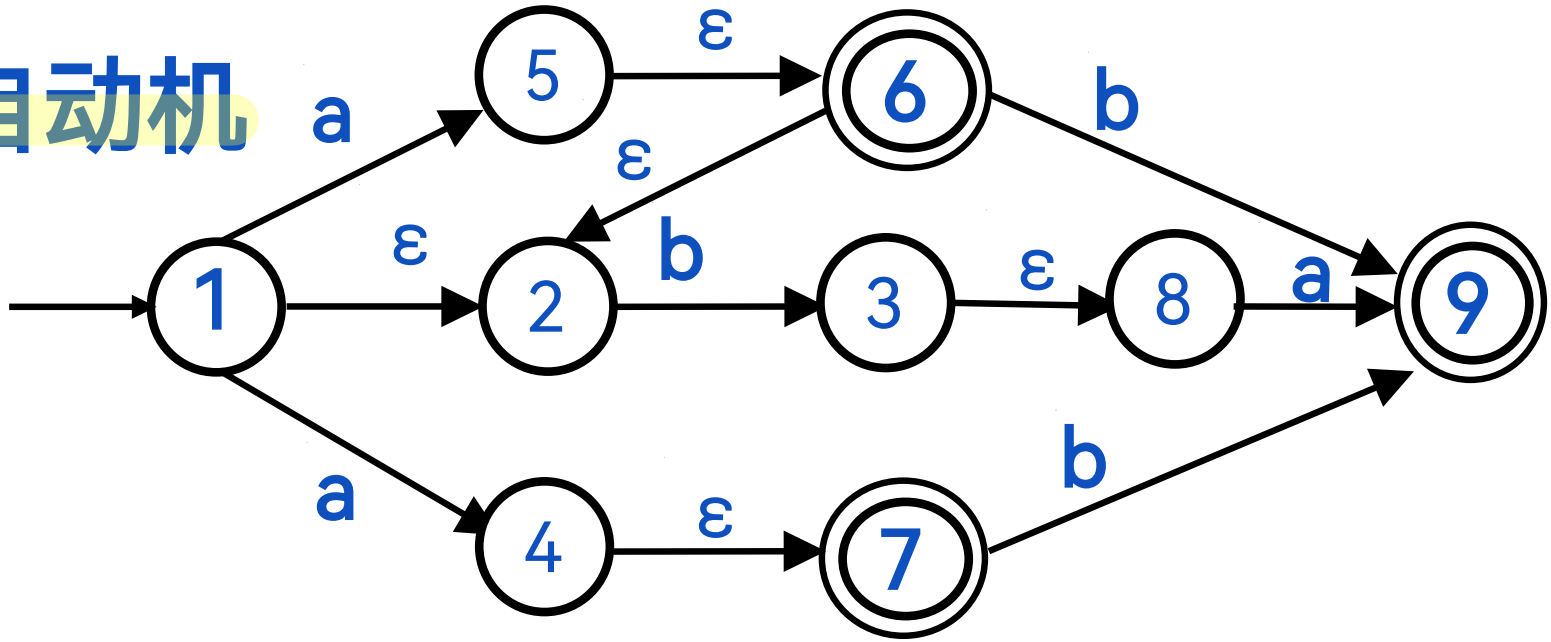
- 把问题分析清楚
- 采用何种描述方式
- 设计算法

单词的描述工具

- **正则表达式**

  $((y|z)^*x(y|z)^*x)^*(y|z)^*$

- **自动机**



基本概念

NEW

字母表 Σ，元素的非空有穷集合。

符号串 由字母表中的符号组成的任何有穷序列。或者如下定义：

1. 空符号串（用ε表示）是Σ上的符号串
2. 若α是Σ上的符号串,x是Σ的元素,则xα是Σ上的符号串
3. β是Σ上的符号串,当且仅当它可以由1和2导出

基本概念

符号串的连接

设α和β均是字母表Σ上的符号串，它们的连接是把β的所有符号顺序接在α的符号之后所得到的符号串。

例如：

有：α=abc　β=def

则：αβ=abcdef

特殊情况 对于空串ε来说

εα=αε

基本概念

符号串的方幂

设α是字母表Σ上的符号串，把α自身连接n次得到的符号串α'，称作符号串α的n 次幂，记作 $\alpha'=\alpha^n$。

$\alpha^0=\varepsilon$

$\alpha^1=\alpha$

$\alpha^2=\alpha\alpha$

$\alpha^3=\alpha^2\alpha=\alpha\alpha^2=\alpha\alpha\alpha$

…

$\alpha^n=\alpha^{n-1}\alpha=\alpha\alpha^{n-1}=\alpha\alpha\ldots\alpha$ (n个α)

基本概念

从这开始最集合.

符号串集合的乘积

设A、B 是两个符号串集合，AB表示A与B的乘积，则定义$AB=\{xy|(x\in A)\wedge(y\in B\}$

例如：

A={ab,cd}, B={ef,gh}

AB={abef,abgh,cdef,cdgh}

特殊：

$A\varnothing=\varnothing A=\varnothing$

基本概念

符号串集合的方幂

设A是符号串集合，则称A^i是符号串集合A的方幂，其中i是非负整数。

$A^0=\{\varepsilon\}$, $A^1=A$, $A^2=AA$, …, $A^n=AA\ldots A$

例如：

A={a,b} A^3就是由a,b组成的任意长度为3的串集。如aab、aba 等等。

{a,b}{a,b}{a,b}分别从每个里取一个元素

基本概念

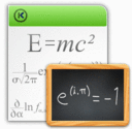
符号串集合的正闭包

设A是符号串集合，则称A^+是符号串集合A的正闭包 $A^+=A^1 \cup A^2 \cup A^3 \ldots \cup A^n \ldots$

符号串集合的星闭包

设A是符号串集合，则称A^*是符号串集合A的星闭包 $A^*=A^0 \cup A^1 \cup A^2 \cup A^3 \ldots \cup A^n \ldots=A^0 \cup A^+$

正则表达式



定义

Σ上的所有正则表达式的集合

设Σ为字母表，RE是定义在Σ上的正则表达式集，则有：

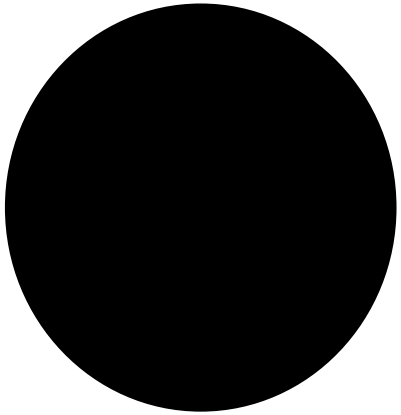
定义

1. ∅,ε∈RE;
2. 对于任意符号a∈Σ，则a∈RE;
3. 若r,s∈RE，则r|s∈RE，
   r·s∈RE，r*∈RE;

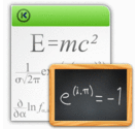
正则表达式

什么是解释，什么是语义？

1+1=?



定义的三点就是正则表达式的形式，它有什么含义需要我们赋予它相应的解释

正则表达式 

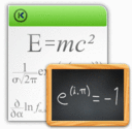
正则表达式对应的语义解释被称作**正则集**

正则集也是正则表达式所表示的**语言**。

在词法分析中，正则表达式是针对单词进行描述的。为此，我们要建立一种由正则表达式到字符串集合的**映射关系**，使得正则表达式的语义解释被描述成字符串的形式。

正则表达式 → 字符串集

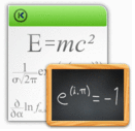
正则表达式



若设e、e1、e2为Σ上的正则表达式，则e所对应的正则集L(e)取值如下：

- 当e=∅时，L(e)=∅；
- 当e=ε时，L(e)={ε}；
- 对于Σ中一个字符a，若e=a，则L(e)={a}；
- 当e=e1·e2 时，L(e)=L(e1)L(e2)；
- 当e=e1|e2时，L(e)=L(e1)∪L(e2)；
- $L(e^*)=L(e)^*$；
- $L(e^+)=L(e)^+$.

正则表达式



若用RE表示∑上的正则表达式，L(RE)表示RE的正则集，且A、B都表示正则表达式，a表示字母表中的任意符号。有：

1)∅∈RE L(∅)={} 2)ε∈RE L(ε)={ε}

3)a∈RE L(a)={a} 4)(A)∈RE L((A))=L(A)

5)A|B∈RE L(A|B)=L(A) U L(B)

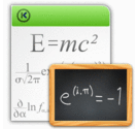
6)A·B∈RE L(A·B)=L(A)L(B)

7)A*∈RE L(A*)=L(A)*;

8)A^+∈RE L(A^+)=L(A)$^+$;

.....如A?,[chi..chk],[abc]

正则表达式



正则表达式的性质

+ = * > . > |　　运算优先级

A | B = B | A　　| 的可交换性

A | (B | C) = (A | B) C　　| 的可结合性

A (B C) = (A B)C　　连接的可结合性

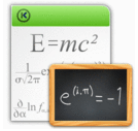
A (B | C) = A B | A C　　连接的可分配性

(A | B) C = A C | B C　　连接的可分配性

$A^{**} = A^{*}$　　幂的等价性

$A\varepsilon = \varepsilon A = A$　　同一律

正则表达式



用正则表达式描述词法

L =A|B|...|a|b|...|z;

D =0|1|...|9; D1=1|2|...|9;

标识符：L(L|D)*

常数：整数：(+|-|ε)(D1D*)|0

正.负.无符号

实数：(+|-|ε)(D1D*|0).D*

特殊符号：用枚举的方式来表示

保留字：while|if|for|...

运算符：+|-|*|...

分界符：{|}|;|...

控制符：\t|\0|...

正则表达式的应用 

手机中常用的号码地区识别软件

软件的安全监测方法

程序分析技术

正则表达式的局限性

缺乏**对称性字符串**的表达能力

例1： $A=\{a^nba^n \mid n>0\}$

例2：

$n \in AE$;

$(AE) \in AE$;

$AE+AE \in AE$

例子

∑={ a,b }

ab* ∑上所有以a为首后跟任意多个（包括0个）b的符号串集

abbb…

a(a|b)* ∑上所有以a为首的符号串集

ab aab abb…

∑={0，1}

所有∑上定义的串的正则表达式 （1|0）*

二进制数 1（1|0）*|0

能被二整除的二进制数 1（1|0）*0|0

习 题

设字母表Σ={x，y，z}，求:

1、包含偶数个x的所有符号串。

2、不包含连续两个y的所有符号串集合。

限定符 (Quantifier)

```
a*  a出现0次或多次
a+  a出现1次或多次
a?  a出现0次或1次
a{6}  a出现6次
a{2,6}  a出现2-6次
a{2,}  a出现两次以上
```

或运算符 (OR Operator)

```
(a|b)  匹配a或者b
(ab)|(cd)  匹配ab或者cd
```

字符类 (Character Classes)

```
[abc]  匹配a或者b或者c
[a-c]  同上
[a-fA-F0-9]  匹配小写+大写英文字符以及数字
[^0-9]  匹配非数字字符
```

元字符 (Meta-characters)

```
\d  匹配数字字符
\D  匹配非数字字符
\w  匹配单词字符(英文、数字、下划线)
\W  匹配非单词字符
\s  匹配空白符(包含换行符、Tab)
\S  匹配非空白字符
.  匹配任意字符(换行符除外)
\bword\b  \b标注字符的边界 (全字匹配)
^  匹配行首
$  匹配行尾
```

贪婪/懒惰匹配 (Greedy / Lazy Match)

```
<.+>  默认贪婪匹配“任意字符”
<.+?>  懒惰匹配“任意字符”
```

断言 (Lookbehind Assertions)