

词法分析

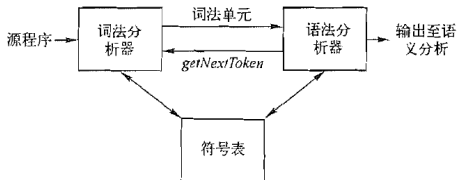
魏恒峰

hfwei@nju.edu.cn

2020 年 11 月 3 日



输入: 程序文本/字符串 s & **词法单元 (token) 的规约**



输出: 词法单元流

token : \langle token-class, attribute-value \rangle

词法单元	非正式描述	词素示例
if	字符 i, f	if
else	字符 e, l, s, e	else
comparison	< 或 > 或 <= 或 >= 或 == 或 !=	<=, !=
id	字母开头的字母 / 数字串	pi, score, D2
number	任何数字常量	3.14159, 0, 6.02e23
literal	在两个 " 之间, 除 " 以外的任何字符	"core dumped"

token : \langle token-class, attribute-value \rangle

词法单元	非正式描述	词素示例
if	字符 i, f	if
else	字符 e, l, s, e	else
comparison	< 或 > 或 <= 或 >= 或 == 或 !=	<=, !=
id	字母开头的字母 / 数字串	pi, score, D2
number	任何数字常量	3.14159, 0, 6.02e23
literal	在两个 "之间, 除" 以外的任何字符	"core dumped"

int/if 关键词

ws 空格、制表符、换行符

comment “//” 开头的一行注释或者 “/* */” 包围的多行注释

```
int main(void)
{
    printf("hello, world\n");
}
```

```
int main(void)
{
    printf("hello, world\n");
}
```

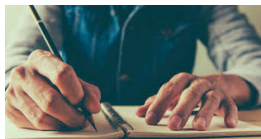
int ws **main/id** LP void RP ws
 LB ws
 ws id LP literal RP SC_{ws}
 RB

```
int main(void)
{
    printf("hello, world\n");
}
```

int ws main/id LP void RP ws
 LB ws
 ws id LP literal RP SCws
 RB

本质上, 这就是一个**字符串 (匹配/识别) 算法**

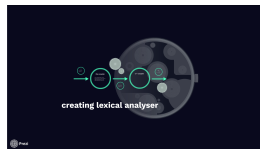
词法分析器的三种设计方法



手写词法分析器



词法分析器的生成器



自动化词法分析器

生产环境下的编译器 (如 gcc) 通常选择**手写词法分析器**



识别字符串 s 中符合某种词法单元模式的所有词素

```
if ab42>=42
    xyz =3.14
else xyz = 2.718
```

ws if else id integer real relop

识别字符串 s 中符合某种词法单元模式的所有词素

```
if ab42>=42
    xyz =3.14
else xyz = 2.718
```

ws if else id integer real relop

识别字符串 s 中符合某种词法单元模式的**第一个词素**

识别字符串 s 中符合某种词法单元模式的所有词素

```
if ab42>=42
    xyz =3.14
else xyz = 2.718
```

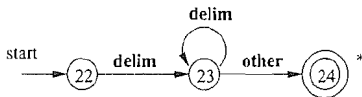
ws if else id integer real relop

识别字符串 s 中符合某种词法单元模式的**第一个词素**

识别字符串 s 中符合**特定词法单元模式**的第一个词素

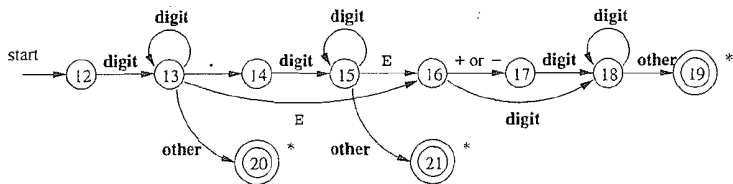
识别字符串 s 中符合**特定词法单元模式**的第一个词素

ws: blank tab newline



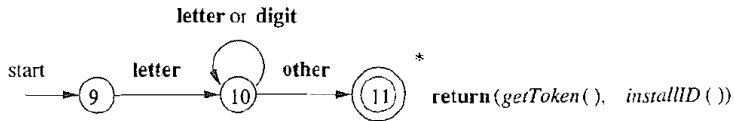
识别字符串 s 中符合**特定词法单元模式**的第一个词素

num: 整数 (允许以 0 开头)



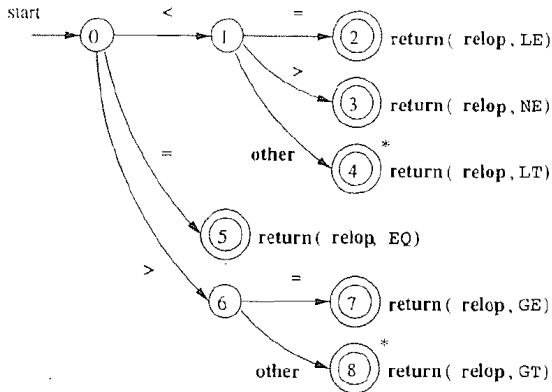
识别字符串 s 中符合**特定词法单元模式**的第一个词素

id: 字母开头的字母/数字串

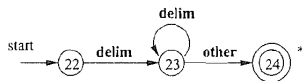
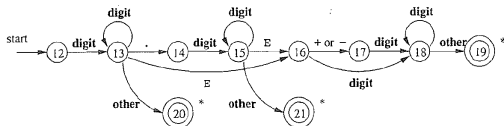
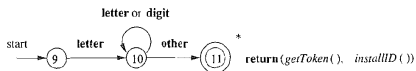
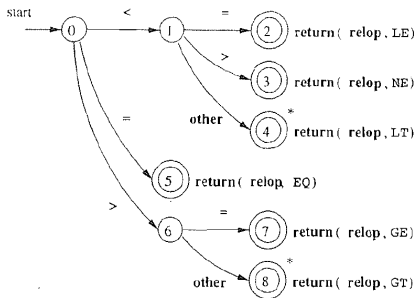


识别字符串 s 中符合**特定词法单元模式**的第一个词素

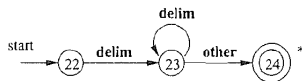
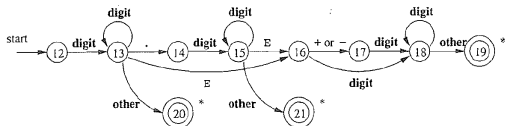
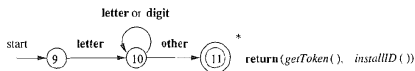
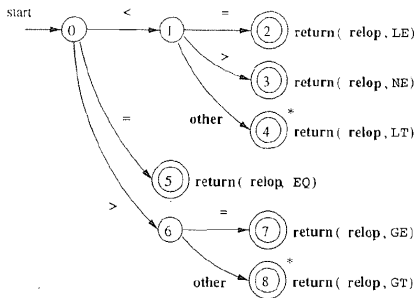
relop: < > <= >= == <>



识别字符串 s 中符合某种词法单元模式的第一个词素 (SCAN())

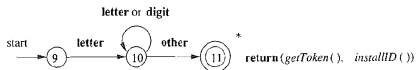
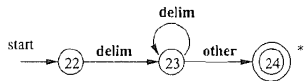
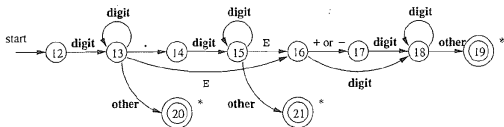
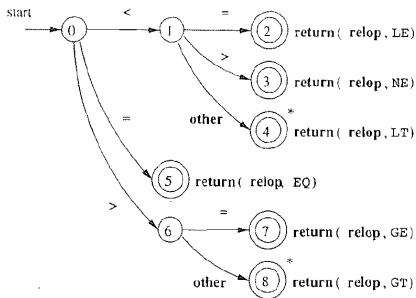


识别字符串 s 中符合**某种词法单元模式**的第一个词素 (SCAN())



根据**下一个字符**即可判定词法单元的类型

识别字符串 s 中符合**某种词法单元模式**的第一个词素 (SCAN())



根据**下一个字符**即可判定词法单元的类型

否则, 报告**该字符有误**, 并忽略该字符

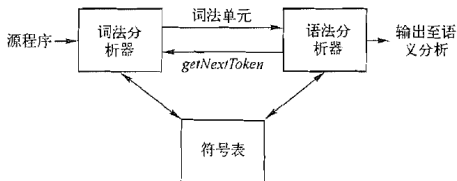
识别字符串 s 中符合某种词法单元模式的所有词素

最外层循环调用 `SCAN()`

或者, 由语法分析器按需调用 `SCAN()`

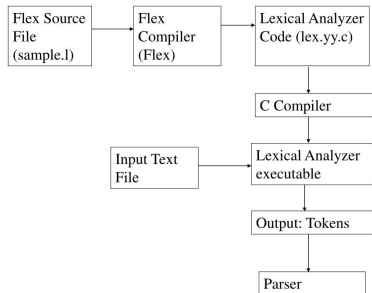


输入: 程序文本/字符串 s & 词法单元的规约



输出: 词法单元流

输入：词法单元的规约

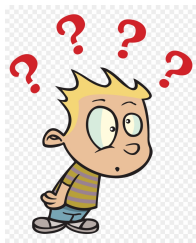


输出：词法分析器

词法单元的规约

词法单元	非正式描述	词素示例
if	字符 i, f	if
else	字符 e, l, s, e	else
comparison	< 或 > 或 <= 或 >= 或 == 或 !=	<=, !=
id	字母开头的字母 / 数字串	pi, score, D2
number	任何数字常量	3.14159, 0, 6.02e23
literal	在两个 " 之间, 除" 以外的任何字符	"core dumped"

词法单元的规约



词法单元	非正式描述	词素示例
if	字符 i, f	if
else	字符 e, l, s, e	else
comparison	< 或 > 或 <= 或 >= 或 == 或 !=	<=, !=
id	字母开头的字母 / 数字串	pi, score, D2
number	任何数字常量	3.14159, 0, 6.02e23
literal	在两个 "之间, 除" 以外的任何字符	"core dumped"

我们需要词法单元的**形式化**规约

id: 字母开头的字母/数字串

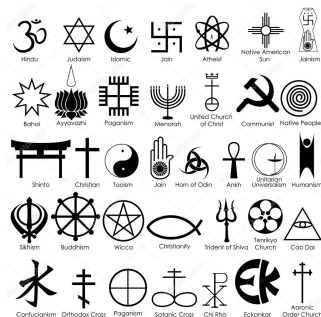
id 定义了一个集合, 我们称之为**语言 (Language)**

它使用了字母与数字等符号集合, 我们称之为**字母表 (Alphabet)**

该语言中的每个元素 (即, 标识符) 称为**串 (String)**

Definition (字母表)

字母表 Σ 是一个有限的符号集合。



Definition (串)

字母表 Σ 上的串 (s) 是由 Σ 中符号构成的一个**有穷**序列。

ϵ

空串 : $|\epsilon| = 0$

Definition (串上的“连接”运算)

$$x = \text{dog}, y = \text{house} \quad xy = \text{doghouse}$$

$$s\epsilon = \epsilon s = s$$

Definition (串上的“连接”运算)

$$x = \text{dog}, y = \text{house} \quad xy = \text{doghouse}$$

$$s\epsilon = \epsilon s = s$$

Definition (串上的“指数”运算)

$$s^0 \triangleq \epsilon$$

$$s^i \triangleq ss^{i-1}, i > 0$$

Definition (语言)

语言是给定字母表 Σ 上一个任意的**可数的**串集合。

$$\emptyset \quad \{\epsilon\}$$

Definition (语言)

语言是给定字母表 Σ 上一个任意的**可数的**串集合。

$$\emptyset \quad \{\epsilon\}$$

$$\mathbf{id} : \{a, b, c, a1, a2, \dots\}$$

$$\mathbf{ws} : \{\mathbf{blank}, \mathbf{tab}, \mathbf{newline}\}$$

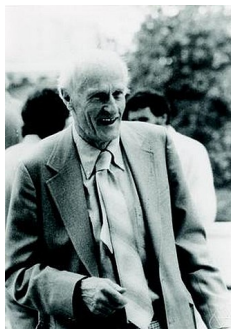
$$\mathbf{if} : \{if\}$$

语言是串的集合

因此, 我们可以通过集合操作**构造**新的语言。

运算	定义和表示
L 和 M 的并	$L \cup M \doteq \{s \mid s \text{ 属于 } L \text{ 或者 } s \text{ 属于 } M\}$
L 和 M 的连接	$LM = \{st \mid s \text{ 属于 } L \text{ 且 } t \text{ 属于 } M\}$
L 的 Kleene 闭包	$L^* = \bigcup_{i=0}^{\infty} L^i$
L 的正闭包	$L^+ = \bigcup_{i=1}^{\infty} L^i$

L^* 允许我们构造**无穷集合**



Stephen Kleene
(1909 ~ 1994)

$$L = \{A, B, \dots, Z, a, b, \dots, z\}$$

$$D = \{0, 1, \dots, 9\}$$

运算	定义和表示
L 和 M 的并	$L \cup M \doteq \{s \mid s \text{ 属于 } L \text{ 或者 } s \text{ 属于 } M\}$
L 和 M 的连接	$LM = \{st \mid s \text{ 属于 } L \text{ 且 } t \text{ 属于 } M\}$
L 的 Kleene 闭包	$L^* = \bigcup_{i=0}^{\infty} L^i$
L 的正闭包	$L^+ = \bigcup_{i=1}^{\infty} L^i$

$$L = \{A, B, \dots, Z, a, b, \dots, z\}$$

$$D = \{0, 1, \dots, 9\}$$

运算	定义和表示
L 和 M 的并	$L \cup M \doteq \{s \mid s \text{ 属于 } L \text{ 或者 } s \text{ 属于 } M\}$
L 和 M 的连接	$LM = \{st \mid s \text{ 属于 } L \text{ 且 } t \text{ 属于 } M\}$
L 的Kleene 闭包	$L^* = \bigcup_{i=0}^{\infty} L^i$
L 的正闭包	$L^+ = \bigcup_{i=1}^{\infty} L^i$

$$L \cup D \quad LD \quad L^4 \quad L^* \quad D^+$$

$$L(L \cup D)^*$$

id : $L(L \cup D)^*$

如何更简洁地描述该 **id** 语言?

$\text{id} : L(L \cup D)^*$

如何更简洁地描述该 id 语言?



下面向大家隆重介绍简洁、优雅、强大的**正则表达式**

每个正则表达式 r 对应一个正则语言 $L(r)$



正则表达式是**语法**, 正则语言是**语义**

Definition (正则表达式)

给定字母表 Σ , Σ 上的正则表达式由且仅由以下规则定义:

- (1) ϵ 是正则表达式;
- (2) $\forall a \in \Sigma, a$ 是正则表达式;
- (3) 如果 r 是正则表达式, 则 (r) 是正则表达式;
- (4) 如果 r 与 s 是正则表达式, 则 $r|s, rs, r^*$ 也是正则表达式。

运算优先级: $() \succ * \succ \text{连接} \succ |$

$$(a)|((b)^*(c)) \equiv a|b^*c$$

每个正则表达式 r 对应一个正则语言 $L(r)$

Definition (正则表达式对应的正则语言)

$$L(\epsilon) = \{\epsilon\} \quad (1)$$

$$L(a) = \{a\}, a \in \Sigma \quad (2)$$

$$L((r)) = L(r) \quad (3)$$

$$L(r|s) = L(r) \cup L(s) \quad L(rs) = L(r)L(s) \quad L(r^*) = (L(r))^* \quad (4)$$

$$\Sigma = \{a, b\}$$

$$L(a|b) = \{a, b\}$$

$$\Sigma = \{a, b\}$$

$$L(a|b) = \{a, b\}$$

$$L((a|b)(a|b))$$

$$\Sigma = \{a, b\}$$

$$L(a|b) = \{a, b\}$$

$$L((a|b)(a|b))$$

$$L(a^*)$$

$$\Sigma = \{a, b\}$$

$$L(a|b) = \{a, b\}$$

$$L((a|b)(a|b))$$

$$L(a^*)$$

$$L((a|b)^*)$$

$$\Sigma = \{a, b\}$$

$$L(a|b) = \{a, b\}$$

$$L((a|b)(a|b))$$

$$L(a^*)$$

$$L((a|b)^*)$$

$$L(a|a^*b)$$



表达式	匹配	例子
c	单个非运算符字符 c	a
$\backslash c$	字符 c 的字面值	$\backslash *$
$"s"$	串 s 的字面值	$"**"$
$.$	除换行符以外的任何字符	$a.*b$
\wedge	一行的开始	$\wedge abc$
$\$$	行的结尾	$abc\$$
$[s]$	字符串 s 中的任何一个字符	$[abc]$
$[^s]$	不在串 s 中的任何一个字符	$[^abc]$
r^*	和 r 匹配的零个或多个串连接成的串	a^*
r^+	和 r 匹配的一个或多个串连接成的串	a^+
$r^?$	零个或一个 r	$a^?$
$r\{m, n\}$	最少 m 个, 最多 n 个 r 的重复出现	$a\{1, 5\}$
$r_1 r_2$	r_1 后加上 r_2	ab
$r_1 r_2$	r_1 或 r_2	$a b$
(r)	与 r 相同	$(a b)$
r_1 / r_2	后面跟有 r_2 时的 r_1	$abc/123$

$[0 - 9] \quad [a - zA - Z]$

num

C 语言的注释对应的正则表达式?

C 语言的注释对应的正则表达式?

`//(^\\n)*\\n`

C 语言的注释对应的正则表达式?

$//(^{\backslash}n)^{*}\backslash n$

$/\ast (^{\ast} | \ast^{\ast} \wedge /)^{\ast} \ast /$

$$(0|(1(01^*0)^*1))^*$$



Thank
You!



Office 926

hfwei@nju.edu.cn