

※<习题一>

填空题:

- 1、编译阶段按前后端组合,可分为编译前端和编译后端,其中与目标机有关的阶段一般属于分析阶段,而与源语言相关的阶段一般属于综合阶段。
- 2、设文法 $G = (V_N, V_T, P, S)$, 若 P 中的每一个规则 $A \rightarrow \beta$ 满足: $A \in V_N, \beta \in (V_N \cup V_T)^*$, 则称此文法为 0 型文法。
- 3、已知 M 为一个确定的有穷自动机, $M = (Q, \Sigma, q_0, F, \delta)$, 则 Q 表示一个有穷的状态集合; Σ 表示字母表, δ 表示状态转换函数, q_0 是唯一的初始状态。
- 4、规范推导是指最右推导, 每步推导只变换符号串中最右边的非终结符, 其逆过程即最左规约, 称为规范归约。
- 5、LR(0) 项目集规范族中的项目可分为四类, 即移进项目、待约项目、归约项目和接受项目, 其中归约项目和归约项目或移进项目共存于一个项目集中会引起冲突。
- 6、表达式 $s:=a+b*c/d+(b-d)$ 的逆波兰式表示为 sabc*d/bd-++:=。

判断题:

- 1、从功能上看, 一个编译程序就是一个语言翻译程序。T
- 2、LEX 是一个语法分析程序的生成系统。F
- 3、一个句型的最左(直接)简单短语称为句柄。T
- 4、已证明文法的二义性是可判定的。F
- 5、一个 NFA 一定能转换为 DFA。T
- 6、递归下降分析法是一种不确定的自顶向下分析法。F

简答:

1、文法 G 是 LL(1) 文法的充要条件是什么?

答: (1). 不能有左递归;

(2). LL(1) 文法的分析表不出现多重定义

即: 对于文法 G 的每个非终结符 A 的任何两条不同规则 $A \rightarrow \alpha \mid \beta$, 下面条件成立:

• $FIRST(\alpha) \cap FIRST(\beta) = \emptyset$ 即头符号集不相交。

• 假若 $\beta \Rightarrow^* \epsilon$, 那么, $FIRST(\alpha) \cap FOLLOW(A) = \emptyset$, 即 α 所能推出的符号串的头符号集中的元素不能出现在 $FOLLOW(A)$ 中。

2、将表达式 $A:=B*(C-D)/D$: 翻译成波兰后缀表达式。

答: $ABCD-D/*:=$

※<习题二>

填空题:

- 1、对给定文法 $G[E]$, 由推导序列 $E \Rightarrow E+T \Rightarrow T+T \Rightarrow i+T \Rightarrow i+i$ 可知: 该推导为(最左)推导, 从该推导序列可得到(5)个句型, 其中的($i+i$)同时也是句子。
- 2、用四元组 $G = (V_N, V_T, P, S)$ 表示文法, 则其元素 V_N 表示(非终结符)集; 元素 V_T 表示(终结符)集; 元素 P 表示规则集; 元素 S 表示开始符号, 它必须是一个(至少在某个产生式的左部出现一次的非终结符)符号。
- 3、YACC 是一种(语法)分析程序的自动构造工具; 而 LEX 是一种(词法)分析程序的自动构造工具。
- 4、对一个文法 G , 在其 LR(0) 项目集规范族 DFA 中, 当有归约项目和(规约)项目或(移

进) 项目共存于同一个状态中时, 该文法就不是 LR(0) 文法。

5、编译程序是一种语言翻译程序, 它将源语言程序翻译成功能等价的 (目标语言程序)。

6、所谓规则或产生式是指形如 $\alpha \rightarrow \beta$ 或 $\alpha ::= \beta$ 的 (α, β) 有序对, 其中 α 是字母表 V 的 (正闭包元素), 而 β 是字母表 V 的 (闭包元素)。

7、设文法 $G = (V_N, V_T, P, S)$, 若 P 中的每一个规则都是 $A \rightarrow aB$ 或 $A \rightarrow a$, 其中 A 和 B 是非终结符, 而 a 是终结符, 则称此文法 G 为正规文法或 (3) 型文法。

8、实用文法中不得含有形如 $U \rightarrow U$ 的 (有害规则), 也不得含有由不可达或不可终止的非终结符所构成的 (多余规则)。

9、对任意给定的一个正则表达式 R , 都可以将它转换为与之功能等价的 (正则) 文法, 或与之功能等价的 (有穷自动机)。

判断题:

1、在语法分析过程中, 随着分析的步步进展, 根据每个规则所对应的语义子程序或语义动作进行翻译的办法, 称为语法制导翻译, 它被现代很多编译程序所采用。T

2、可归前缀本身就是活前缀, 它是包含句柄在内的活前缀。T

简答:

一、现有文法 $G[E]$:

$E \rightarrow EE+$

$E \rightarrow EE*$

$E \rightarrow F$

$F \rightarrow i$

1、证明 $ii*i+$ 是文法的一个句子。

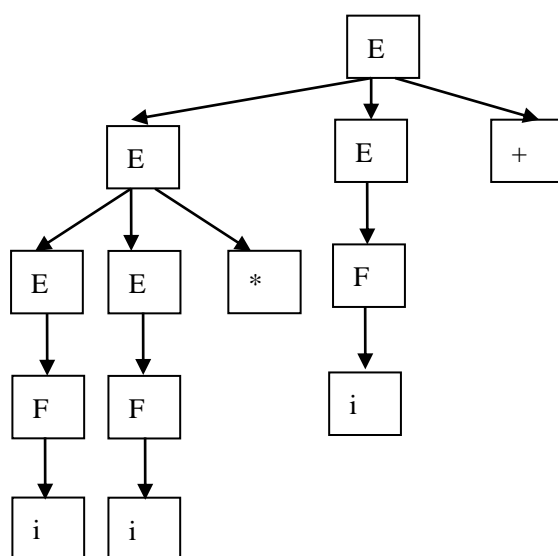
2、构造句型 $ii*i+$ 的语法推导树。

3、指出该句型所有短语、简单短语和句柄。

答:

1、因为存在一个最左推导过程: $E \rightarrow EE+ \rightarrow FE+ \rightarrow FE+ \rightarrow iE+ \rightarrow iE*E+ \rightarrow iF*E+ \rightarrow ii*E+ \rightarrow ii*F+ \rightarrow ii*i+$ 并且, $ii*i+$ 只含有终结符。

2、



3、短语: $ii*i+$ $ii* \quad i \quad +$

简单短语: $i \quad * \quad +$

句柄: i

二、现有文法 $G[E]$:

$E \rightarrow E+T \mid E-T \mid T$

$T \rightarrow T*F \mid T/F \mid F$

$F \rightarrow i \mid (E)$

1、证明: $F+T*(E)$ 是文法的一个句型。

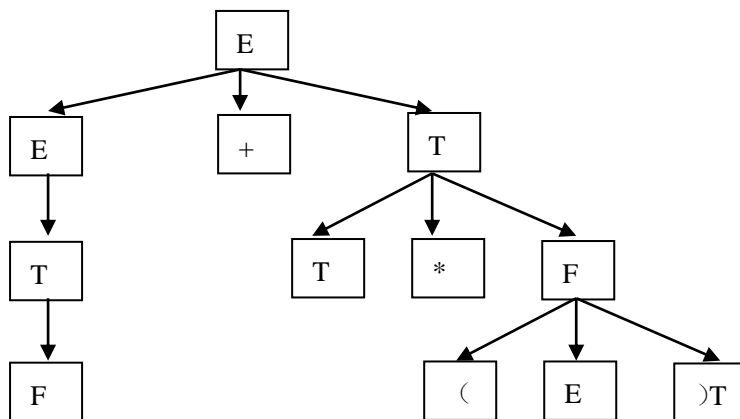
2、构造句型 $F+T*(E)$ 的语法推导树。

3、指出该句型所有短语、直接短语和句柄。

答: 1、因为存在这样的一个推导过程:

$E \rightarrow E+T \rightarrow T+T \rightarrow F+T \rightarrow F+T*F \rightarrow F+T*(E)$

2、



3、短语: $F+T*(E)$ $T*(E)$ (E)

简单短语: (E)

句柄: (E)

※<习题三>

填空题:

1、在语法分析过程中,随着分析的步步进展,根据每个规则所对应的语义子程序或语义动作进行翻译的办法,称为(语法制导)翻译方法,它被现代很多编译程序所采用。

2、(YACC) 是一种语法分析程序的自动构造工具,用它可以直接构造各种语言的语法分析器;而 (LEX) 是一种 (词法分析) 程序的自动构造工具,用它可以直接构造各种语言的词法分析器。3、所谓 2 型文法就是指 (上下文无关) 文法,若用 $G = (VN, VT, P, S)$ 表示它,则它要求 G 中的所有规则 $\alpha \rightarrow \beta$ 都满足: α 是 (VN) , 而 β 属于 $(VN \cup VT)^*$ 。

4、文法中形如 $U \rightarrow U$ 的规则称为 (有害) 规则;由不可达的非终结符或不可终止的非终结符作为左部的规则称为 (无用) 规则。在实用文法中一般不允许含有这两类规则。

6、语法分析方法分为自上而下与自下而上两类,自上而下的分析方法方要有递归子程序分析法和 (11 (1));而自下而上的分析方法主要有 (算符优先方法) 和 (LR 分析方法)。

7、LR(0) 项目集规范族中的项目可分为四类，它们是移进项目、(待约)、归约项目和接受项目。其中，接受项目是 (规约) 的一种特例。

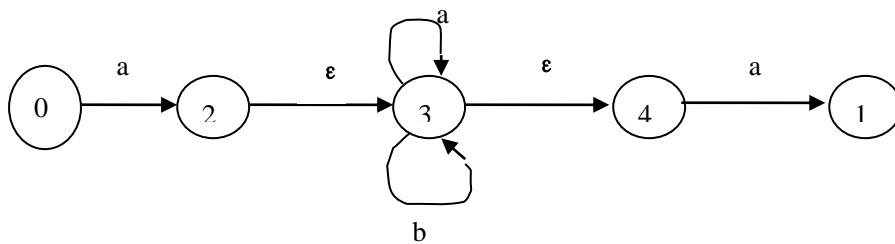
8、将非 LL(1) 文法转换为等价的 LL(1) 文法所采用的两种方法是 (左递归转换为有递归) 和 (提取公因子解决分析表多重定义)。但这两种方法并不能保证所有的非 LL(1) 文法都能转换为等价的 LL(1) 文法。

9、编译阶段按前后端组合，可分为编译前端和编译后端，其中与目标机有关的阶段一般属于 分析阶段，而与源语言相关的阶段一般属于 综合阶段

简答：

求出正规式 $a(a|b)^*a$ 对应的 NFA，并将之确定化，如有可能，将其最小化。

答：先构造等价的 NFA 如下图所示：



然后确定化，将其转换为 DFA 如下图所示::

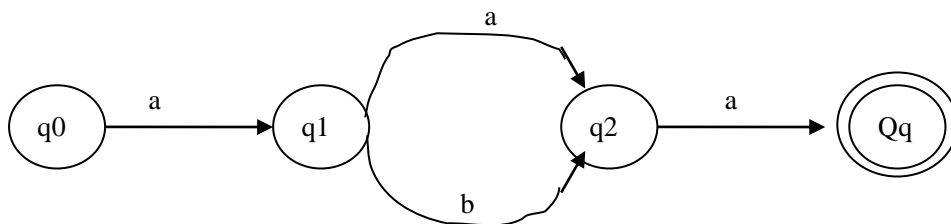
$q_0 = \epsilon\text{-closure}(\{0\}) = \{0\}$

$\delta(q_0, a) = \epsilon\text{-closure}(\delta(0, a)) = \{2, 3, 4\} = q_1$

$\delta(q_1, a) = \{3, 4\} = q_2$

$\delta(q_1, b) = \{3, 4\} = q_2$

$\delta(q_2, a) = \{1\} = q_3$



※<习题四>

填空题：

1、(2) 型文法又称为 (上下文无关) 文法，是描述程序设计语言语法部分的主要文法；高级程序设计语言的单词符号，如标识符、无符号整数常用 (3) 型文法来描述。

2、从 0 型文法到 3 型文法对规则的限制逐渐 (增加)，产生的语言类却逐步 (缩小)。

简答：

一、现有文法 $G[S]$ ：

$S \rightarrow aB$

$S \rightarrow Bb$

$B \rightarrow Sc$

$B \rightarrow d$

1、消除文法 $G[S]$ 的左递归；

2、指出消除了左递归之后的文法是否是 LL(1) 文法, 为什么?

二、已知 $G[S]$:

$S \rightarrow b|+|(T)$

$T \rightarrow T, S|S$

1、给出 $(+, (b, +))$ 的最左推导。

2、证明 $G[S]$ 不是 LL(1) 文法。

3、将 $G[S]$ 改写为 LL(1) 文法, 再给出它的分析表;

4、给出输入串 $(b, +)\#$ 的分析过程。

三、已知 $G[S]$:

$S \rightarrow (A) | a | b$

$A \rightarrow A, S | S$

1、给出 $(a, (b, b))$ 的最左推导。

2、判断该文法是否为 LL(1) 文法。若是, 直接给出它的分析表; 若不是, 先将其改写为 LL(1) 文法, 再给出它的分析表;

3、给出输入串 $(b, b)\#$ 的分析过程, 并说明该串是否为 $G[S]$ 的句子。

四、对给定文法 $G[S']$:

0) $S' \rightarrow S$

1) $S \rightarrow A$

2) $S \rightarrow B$

3) $A \rightarrow aAc$

4) $A \rightarrow a$

5) $B \rightarrow bBd$

6) $B \rightarrow b$

1、构造 $G[S']$ 的 LR(0) 项目集规范族 DFA, 并据此判断 $G[S']$ 是否为 LR(0) 文法。

五、现有文法 $G[S]$:

(0) $S' \rightarrow S$

(1) $S \rightarrow A - B$

(2) $S \rightarrow B$

(3) $A \rightarrow + B$

(4) $A \rightarrow a$

(5) $B \rightarrow b$

试判断 $G[S]$ 是否为: LR(0) 文法并说明原因。