1. 源语言：书写源程序所使用的语言
2. 源程序：用程序设计语言书写的程序
3. 目标语言：计算机的机器指令。目标语言可以是机器语言，也可以是汇编语言，或者是其他中间语言，但最终结果必是机器语言。
4. 目标程序:由机器指令构成的程序。 目标程序是经过翻译程序加工后用目标语言表示的程序。
5. 翻译程序：能够把某一种语言程序（源程序）改造成另一种语言程序（目标程序）将源程序译成逻辑上等价的目标程序的程序。翻译程序有两种工作方式：编译和解释。
6. 编译程序：也称翻译程序
7. 解释程序：有些翻译程序在翻译过程中并不产生完整的目标程序，而是翻译一句，解释执行一句，这样的称为解释程序。
8. 汇编程序：由汇编语言写成的程序
9. 词法分析：执行词法分析的程序成为词法分析器，词法分析依据的是语言构词规则。词法分析器从文件读入源程序，由字符拼接单词。每当识别出一个单词，词法分析器就输出这个单词的内部码。
10. 语法分析：执行语法分析的程序叫做语法分析器。语法分析的任务就是根据语言的规则，将词法分析器所提供的单词种别分成各类语法范畴。
11. 中间代码生成：中间代码产生有时称为语义分析，执行中间代码产生的程序称为中间代码生成器。他的任务时按照语法分析器所识别出的语法范畴产生相应的中间代码，并建立符号表、常数表，等各种表格。
12. 目标代码生成：执行目标代码生成的程序称为目标代码生成器。他的任务是根据中间代码和表格信息，确定各类数据在内存中的位置，选择合适的指令代码，将中间代码翻译成汇编语言或机器指令，这部分工作与计算机硬件有关。
13. 符号表：用于记录源程序中出现的标识符，一个标识符往往具有一系列的语义值，她包括标识符的名称、种属、类型、值存放的地址等等。
14. 常数表：用于记录在源程序中出现的常数。
15. 编译程序前端：是由词法分析器、语法分析器和中间代码产生器组成的。她的特点是依赖于被编译的源程序，输出结果用中间代码描述，和目标机器无关。
16. 编译程序后端：是由目标代码生成器组成，他的特点是和源程序无关，以中间代码形式的源程序为输入进行处理，输出结果依赖于目标机器。
17. 文本文件：文本文件的内容由 94 个图形字符‘！‘-’～‘(33-126）和 4 个控制字符换行（10)、回车（13)、空格（32)、TAB（9）构成，文本文件又称为 ASCII 码文件，扩展名通常为 TXT，文件尾用控制字符 EOF（26）指示。
18. 二进制文件：由机器指令即二进制数构成，因二进制数可能是 26（文件结束控制符)，故文件尾用文件长度（文件的字节数）指示，扩展名通常为ＥＸＥ。
19. 源代码 (source code) → 预处理器 (preprocessor) → 编译器 (compiler) → 汇编程序 (assembler) → 目标代码 (object code) → 链接器 (Linker) → 可执行程序 (executables)
20. 编译程序的流程是：
    源程序－》词法分析－》语法分析－》语义分析（中间代码产生）－》目标

代码生成一》目标程序

21. 二元式编码表：

单词	二元式
b e g i n	（’ {’ , ” N U L ”）
e n d	（’ } ’ ,” NUL”）
real	（‘c’ ,” NUL”）
integer	（ ‘a’ ,” NUL”）
标识符	（ ‘i’ ,” abc”）
无符号整数	（ ‘x’ ,” 223”）
无符号实数	（ ‘y’ ,” 1.23”）

22. 词法分析的各种正规式所代表的含义

(1)a(a|b)* 描述标识符的正规式

（2）bb* 描述无符号整数的正规式

（3)bb*.b* .bb* bb*.b*(E|e)(+|-| ε)bb* 描述的是无符号实数的正规式

（4）(0|1)(0|1)* 描述二进制数的正规式

23. 左递归的消除

文法：PPα｜β 消除左递归的公式是PβP’ P’ αP’ ｜ε

24. 提取左因子

文法：Pδβ1｜δβ2｜δβ3|…｜δβn 提取左因子的公式是

PP’ P’ β1｜β2｜β3|…｜βn

25. First 集和 Follow 集规律【E】

First 集：（1）aB 为ε，则 E 终结符的这种，则 b 在 Fisrt(E)中（2）a 在 First（E）中，此时的a 可以是+，-，*，/，.等（3）a 为ε，则First(B)/ε添加到First(E)中

Follow 集：（1）文法的开始符号，那么#在 Follow（E）中（2）看紧跟在所要求的那个非终结符后面的元素，将 first（b）/ε添加到 Follow（B)(3)若 b 为ε，或者文法式为 E，则 Follow（E）添加到 Follow（B）中

26. LL（1）分析表的构造

将非终结符的 first 集中的符号列下填上相对应的文法规则

若将非终结符的 first 集中含有ε，则在 Follow 集中的符号列下填上推出ε的文法规则

27. LR（0）分析表的构造

（1）A rk（K 为文法规则的编号）

(2)A 数字 m(m 为 Ij 的 j)

(3)S Acc

(4)A sj(j 为 Ij 的 j)

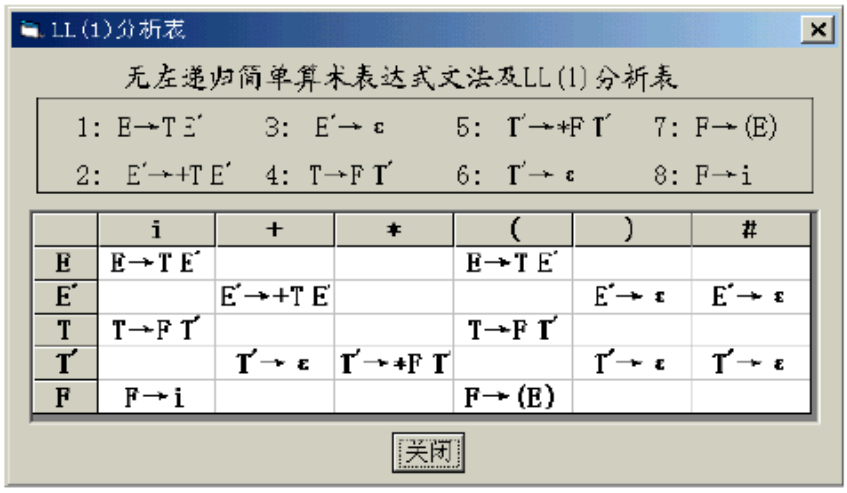
28. SLR 分析表的构造

删除非终结符的 Follow 集中的不存在的那些列中的值

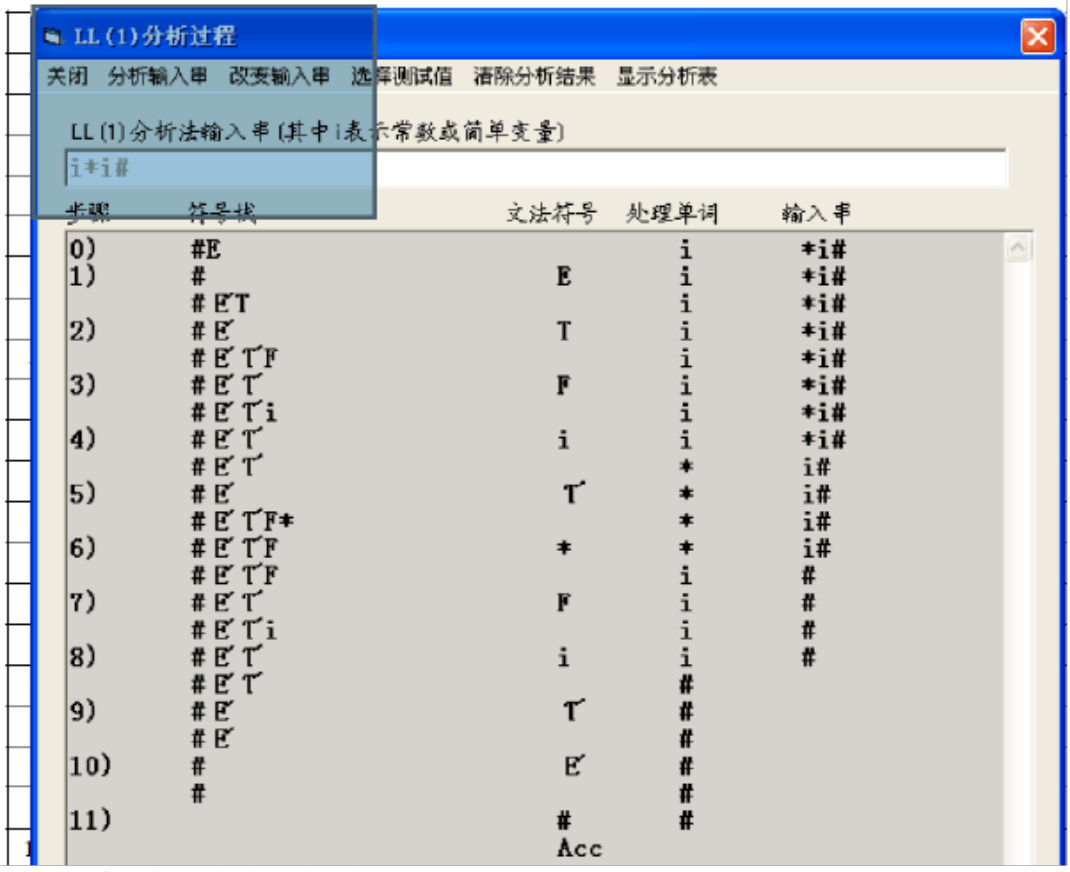
28. 文法分析过程

(10 分)已知文法 G

1. E→TE'
2. E'→+TE'
3. E'→ ε
4. T→FT'
5. T'→*FT'
6. T'→ ε
7. F→(E)
8. F→i



LL(1)分析表见左图。假定输入串为 i*i#，写出预测分析执行过程。



29. LR 语法分析器的控制程序

例如：a*b+c

经词法分析，单词的二元式为(‘i’,”a”),(‘*’,”NUL”),(‘i’,”b”),(‘+’,”NUL”),(‘i’,”c”),(‘#’,”NUL”)

因此单词的种别序列为 i*i+i#

step	状态栈	符号栈	输入串	动作
0)	0	#	i*i+i#	初始

1）	05	#i	*i+i#	移进
2）	03	#F	*i+i#	归约【1】
3）	02	#T	*i+i#	归约【2】
4）	027	#T*	i+i#	移进
5）	0275	#T*i	+i#	移进
6）	02710	#T*F	+i#	归约【3】
7）	02	#T	+i#	归约【4】
8）	01	#E	+i#	归约【5】
9）	016	#E+	i#	移进
10）	0165	#E+i	#	移进
11）	0163	#E+F	#	归约【6】
12）	0169	#E+T	#	归约【7】
13）	01	#E	#	归约【8】
14）		Acc		接受

注：【1】i
【2】F
【3】i
【4】T*F
【5】T
【6】i
【7】F
【8】E+T

30. aVbVc 语法制导翻译过程如下所示

step	symbol	wval	.addr	.tc	.fc	输入串	nxq=1
0	#	-	-	-	-	(i,”a”)	
1	#i	-a	--	--	--	(V,”NUL”)	
2	#X	--	-&a	--	--	(V,”NUL”)	
3	#E	--	--	-1	-2	(V,”NUL”)	(1)(jnz,&a,0,0) (2)(jmp,0,0,3) nxq=3
4	#EV	---	---	-1-	-2-	(i,”b”)	
5	#E₀	--	--	-1	--	(i,”b”)	

6	#E∘i	--b	---	-1-	--	(V,”NUL”)	
7	#E∘X	---	---	-1-	---	(V,”NUL”)	
8	#E∘E	---	---	-13	--4	(V,”NUL”)	(3)(jnz,&b,0,1) (4)(jmp,0,0,5) nxq=5
9	#E	--	--	-3	-4	(V,”NUL”)	
10	#EV	---	---	-3-	-4-	(i,”c”)	
11	#E∘	--	--	-3	--	(i,”c”)	
12	#E∘i	--c	---	-3-	---	(#,”NUL”)	
13	#E∘X	---	--&c	-3-	---	(#,“”NUL)	
14	#E∘E	---	---	-35	--6	(#,”NUL”)	(5)(jnz,&c,0,3) (6)(jmp,0,0,0) nxq=7
15	#E	--	--	-5	-6	(#,”NUL”)	
	Acc						E.tc=5 E.fc=6

31.设源程序为a∧b∧c,经词法分析,他的二元式序列为:(i,”a”)

(∧,”NUL”)(i,”b”) (∧,”NUL”)(i,”c”)(#,”NUL”)

step	symbol	wval	.addr	.tc	.fc	输入串	nxq=1
0	#	-	-	-	-	(i,”a”)	
1	#i	-a	--	--	--	(∧,”NUL”)	
2	#X	--	-&a	--	--	(∧,”NUL”)	
3	#E	--	--	-1	-2	(∧,”NUL”)	(1)(jnz,&a,0,3)

							(2)(jmp,0,0,0) nxq=3
4	#E Λ	---	---	-1-	-2-	(i,” b”)	
5	#EA	--	--	--	-2	(i,” b”)	
6	#EAi	--b	---	---	-2-	(Λ,” NUL”)	
7	#EAX	---	--&b	---	-2-	(Λ,” NUL”)	
8	#EAE	---	---	--3	-24	(Λ,” NUL”)	(3)(jnz,&b,0,5) (4)(jmp,0,0,2) nxq=5
9	#E	--	--	-3	-4	(Λ,” NUL”)	
10	#EV	---	---	-3-	-4-	(i,” c”)	
11	#EA	--	--	--	-4	(i,” c”)	
12	#EAi	--c	---	---	-4-	(#,” NUL”)	
13	#EAX	---	--&c	---	-4-	(#,“” NUL)	
14	#EAE	---	---	--5	-46	(#,” NUL”)	(5)(jnz,&c,0,3) (6)(jmp,0,0,0) nxq=7
15	#E	--	--	-5	-6	(#,” NUL”)	
	Acc						E.tc=5 E.fc=6