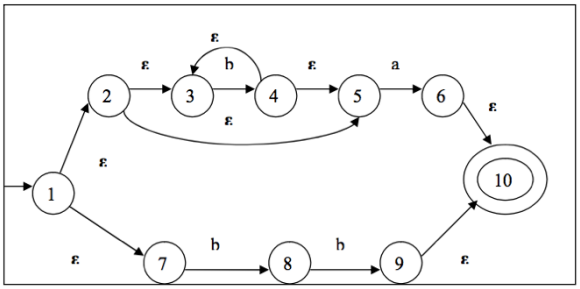
1. 【词法分析】考虑定义在字母表Σ = { **a, b** }正则语言：b*a|bb：

（1） 画出识别该语言的不确定有限自动机(NFA)。

【参考答案】



（2） 将得到的NFA转化为等价的DFA，给出转换表和状态转换图。

【参考答案】

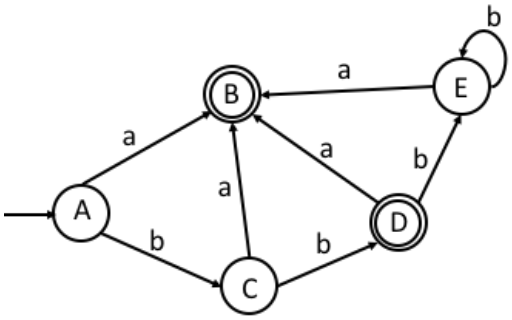
转换过程：

ε-closure({1}) = **{1,2,3,5,7}**
move({1,2,3,5,7}, a) = {6} ε-closure({6}) = **{6, 10}** move({1,2,3,5,7}, b) = {4, 8} ε-closure({4,8}) = **{3,4,5,8}**
move({6,10}, a) = ∅ move({6,10}, b) = ∅
move({3,4,5,8}, a) = {6} ε-closure({6}) = {6, 10} move({3,4,5,8}, b) = {4, 9} ε-closure({4,9}) = **{3,4,5,9,10}**
move({3,4,5,9,10}, a) = {6} ε-closure({6}) = {6, 10} move({3,4,5,9,10}, b) = {4} ε-closure({4,8}) = **{3,4,5}**
move({3,4,5}, a) = {6} ε-closure({6}) = {6, 10} move({3,4,5}, b) = {4} ε-closure({4,8}) = {3,4,5}

最终结果：

状态别名	DFA状态	a	b
A	{1,2,3,5,7}	{6,10}	{3,4,5,8}
B	{6,10}	-	-
C	{3,4,5,8}	{6,10}	{3,4,5,9,10}

D	{3,4,5,9,10}	{6,10}	{3,4,5}
E	{3,4,5}	{6,10}	{3,4,5}



（3） 判断（2）中所得到的DFA状态是否已最小化。若是，请简述理由；若否，请将其最小化[3分]

【参考答案】

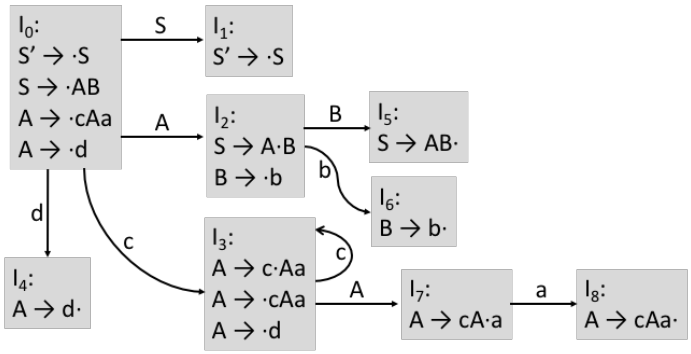
已经最小化。从（2）中得到的转换表可以看出，只有D和E在对输入a/b的转换是完全相同的，但这两个状态并不能合并，因为D是终结状态，而E不是。

2. 【语法分析】对于如下文法G，增广后文法为G’：

文法G	增广文法G’
(1) S -> AB	(0) S’ -> S
(2) A -> cAa | d	(1) S -> AB
(3) B -> b	(2) A -> cAa | d
	(3) B -> b

（1） 对文法G’构建LR(0)解析的有穷自动机(FA)，包括状态和转换。提示：项目集闭包（closure of item sets)。

【参考答案】



（2） 构建LR(0)解析表。注：如有需要，请自行添加更多行。

State	ACTION					GOTO		
	a	b	c	d	$	S	A	B

【参考答案】

State	ACTION					GOTO		
	a	b	c	d	$	S	A	B
0			s3	s4		1	2	
1					acc			
2		s6						5
3			s3	s4			7	
4	r3	r3	r3	r3	r3			
5	r1	r1	r1	r1	r1			
6	r4	r4	r4	r4	r4			
7	s8							
8	r2	r2	r2	r2	r2			

（3） 列出解析输入串cdab的过程，包括每一步输入串和解析栈变化及采取的具体动作。注：如有需要，请自行添加更多行。

Stack	Input	Action

【参考答案】

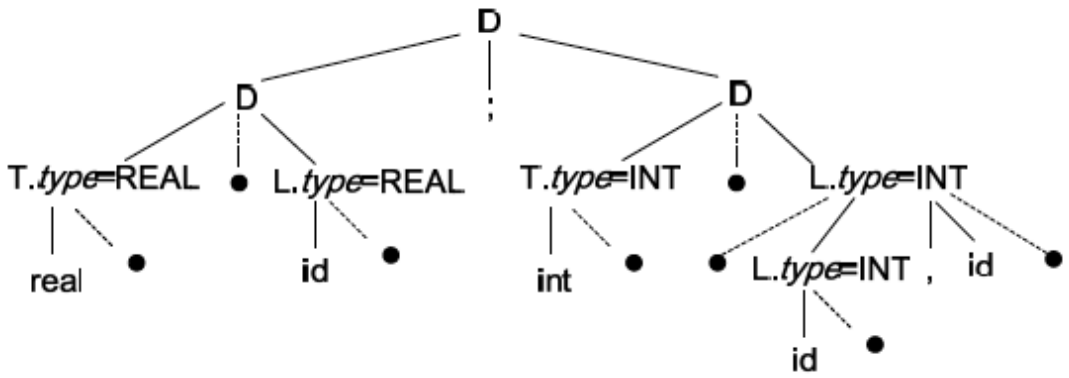
Stack	Input	Action
0 $	cdab$	[0, c]: shift, to state 3
0 3 $ c	dab$	[3, d]: shift, to state 4
0 3 4 $ c d	ab$	[4, a]: reduce, A -> d
0 3 $ c A	ab$	[GOTO: [3, A] => 7
0 3 7 $ c A	ab$	[7, a]: shift, to state 8
0 3 7 8 $ c A a	b$	8, b]: reduce, A -> cAa
0 $ A	b$	GOTO: [0, A] => 2
0 2 $ A	b$	[2, b]: shift, to state 6
0 2 6 $ A b	$	[6, $]: reduce, B -> b
0 2 $ A B	$	GOTO: [2, B] => 5
0 2 5 $ A B	$	[5, $]: reduce, S -> AB
0 $ S	$	GOTO: [0, S] => 1
0 1 $ S	$	acc

3. 【语义分析】根据如下定义的翻译模式：

D → D ; D	
D → T	{ L.type:=T.type }
L	
T → **int**	{ T.type:=INT }
T → **real**	{ T.type:=REAL }
L →	{ L_1.type:=L.type }
L_1 , id	{ enter (id.name, L.type)}
L → id	{ enter (id.name, L.type)}

画出对输入串real x ; int y, z 构造的带属性和带动作的分析树。（提示：所谓带属性和带动作的分析树，是指根据翻译模式在同一颗分析树上挂上语义动作并标注各符号的属性值。）

【参考答案】



注：花括号括起来的语义动作在图中简化为•,请大家自行补充。