LR(1) grammar ('' is ε):

```
(0) <STMT> -> <if_stmt> |
(1) <while_stmt> | <assign>
(2) <if_stmt> ->
(3) 'if''('<bool_expr>')'
(4) <stmt> ['else' <stmt>]
(5) <while_stmt> -> 'while'
(6) '('<bool_expr>')' <stmt>
    <assign> -> 'id' '='
    <expr>';'
    <expr> -> <term>{
    ('+'|'_') <term> }
    <term> -> <factor> {
    ('*'|'\'|'%') <factor>}
    <factor> -> 'id' |
    'int_lit' | 'float_lit' |
    '(' <expr> ')'
```

>>

LR(1) closure table

Goto	Kernel	State	Closure
	{[-> . | | , $]}	0	{[-> . | | , $]; [-> .'if''(')' ['else'], |]}
goto(0,)	{[-> .| | , $]}	1	{[-> .| | , $]}
goto(0, 'if''(')')	{[-> 'if''('). ['else'], |]}	2	{[-> 'if''('). ['else'], |]}
goto(1, |)	{[-> |. | , $]}	3	{[-> |. | , $]; [-> .'while' '('')' , |]}
goto(2,)	{[-> 'if''(')' .['else'], |]}	4	{[-> 'if''(')' .['else'], |]}
goto(3,)	{[-> | .| , $]}	5	{[-> | .| , $]}
goto(3, 'while')	{[-> 'while'.'('')' , |]}	6	{[-> 'while'.'('')' , |]}
goto(4, ['else')	{[-> 'if''(')' ['else'.], |]}	7	{[-> 'if''(')' ['else'.], |]}
goto(5, |)	{[-> | |., $]}	8	{[-> | |., $]; [-> .'id' '=' ';', $]}
goto(6, '('')')	{[-> 'while' '('')'., |]}	9	{[-> 'while' '('')'., |]}
goto(7,])	{[-> 'if''(')' ['else']., |]}	10	{[-> 'if''(')' ['else']., |]}
goto(8,)	{[-> | | ., $]}	11	{[-> | | ., $]}
goto(8, 'id')	{[-> 'id'.'=' ';', $]}	12	{[-> 'id'.'=' ';', $]}
goto(9,)	{[-> 'while' '('')' ., |]}	13	{[-> 'while' '('')' ., |]}
goto(12, '=')	{[-> 'id' '='.';', $]}	14	{[-> 'id' '='.';', $]}
goto(14, ';')	{[-> 'id' '=' ';'., $]}	15	{[-> 'id' '=' ';'., $]}

FIRST table

Nonterminal	FIRST
	{'if''(')'}
	{'if''(')'}
	{'while'}
	{'id'}
	{{}
	{'id'}
	{'id'}

LR table

State	ACTION																			GOTO					
	|	'if''(')'		['else'	]	'while'	'('')'	'id'	'='	';'	{	('+'|'_')	}	{	('*'|'\'|'%')	}	'int_lit'	'float_lit'	'('	')'	$						
0		s2																				1					
1	s3																										
2				s4																							
3							s6																5				
4					s7																						
5	s8																										
6								s9																			
7						s10																					
8									s12															11			
9				s13																							
10	r_1																										
11																					acc						
12										s14																	
13	r_2																										
14											s15																
15																					r_3						

Input (tokens): c d d

Maximum number of steps: 1

PARSE

Trace

Step	Stack	Input	Action
1	0	c d d $	