SLR grammar ('' is ε):

(0) CODE -> VDECL CODE
(1) CODE -> FDECL CODE
(2) CODE -> CDECL CODE
(3) CODE -> ''
(4) VDECL -> vtype id semi
(5) VDECL -> vtype ASSIGN semi
(6) ASSIGN -> id assign RHS
(7) RHS -> EXPR
(8) RHS -> literal
(9) RHS -> character
(10) RHS -> boolstr
(11) EXPR -> lparen EXPR addsub EXPR rparen
(12) EXPR -> lparen EXPR multdiv EXPR rparen
(13) EXPR -> lparen EXPR rparen
(14) EXPR -> id
(15) EXPR -> num
(16) FDECL -> vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace
(17) ARG -> vtype id MOREARGS
(18) ARG -> ''
(19) MOREARGS -> comma vtype id MOREARGS
(20) MOREARGS -> ''
(21) BLOCK -> STMT BLOCK
(22) BLOCK -> ''
(23) STMT -> VDECL
(24) STMT -> ASSIGN semi
(25) STMT -> if lparen COND rparen lbrace BLOCK rbrace ELSE
(26) STMT -> while lparen COND rparen lbrace BLOCK rbrace
(27) COND -> lparen COND comp COND rparen
(28) COND -> boolstr
(29) ELSE -> else lbrace BLOCK rbrace
(30) ELSE -> ''
(31) RETURN -> return RHS semi
(32) CDECL -> class id lbrace ODECL rbrace
(33) ODECL -> VDECL ODECL
(34) ODECL -> FDECL ODECL
(35) ODECL -> ''

FIRST / FOLLOW table

Nonterminal	FIRST	FOLLOW
CODE	{'',vtype,class}	{$}
VDECL	{vtype}	{$,vtype,class,return,id,if,while,rbrace}
ASSIGN	{id}	{semi}
RHS	{literal,character,boolstr,lparen,id,num}	{semi}
EXPR	{lparen,id,num}	{semi,addsub,rparen,multdiv}
FDECL	{vtype}	{$,vtype,class,rbrace}
ARG	{vtype,''}	{rparen}
MOREARGS	{comma,''}	{rparen}
BLOCK	{'',vtype,id,if,while}	{return,rbrace}
STMT	{vtype,id,if,while}	{return,vtype,id,if,while,rbrace}
COND	{lparen,boolstr}	{rparen,comp}
ELSE	{else,''}	{return,vtype,id,if,while,rbrace}
RETURN	{return}	{rbrace}
CDECL	{class}	{$,vtype,class}
ODECL	{vtype,''}	{rbrace}

>>

SLR closure table

Goto	Kernel	State	Closure
	{CODE -> .VDECL CODE}	0	{CODE -> .VDECL CODE; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi}
goto(0, VDECL)	{CODE -> VDECL.CODE}	1	{CODE -> VDECL.CODE; CODE -> .VDECL CODE; CODE -> .FDECL CODE; CODE -> .CDECL CODE; CODE -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace; CDECL -> .class id lbrace ODECL rbrace}
goto(0, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; ASSIGN -> .id assign RHS}
goto(1, CODE)	{CODE -> VDECL CODE.}	3	{CODE -> VDECL CODE.}
goto(1, VDECL)	{CODE -> VDECL.CODE}	1	
goto(1, FDECL)	{CODE -> FDECL.CODE}	4	{CODE -> FDECL.CODE; CODE -> .VDECL CODE; CODE -> .FDECL CODE; CODE -> .CDECL CODE; CODE -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace; CDECL -> .class id lbrace ODECL rbrace}
goto(1, CDECL)	{CODE -> CDECL.CODE}	5	{CODE -> CDECL.CODE; CODE -> .VDECL CODE; CODE -> .FDECL CODE; CODE -> .CDECL CODE; CODE -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace; CDECL -> .class id lbrace ODECL rbrace}
goto(1, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace; ASSIGN -> .id assign RHS}
goto(1, class)	{CDECL -> class.id lbrace ODECL rbrace}	7	{CDECL -> class.id lbrace ODECL rbrace}
goto(2, id)	{VDECL -> vtype id.semi; ASSIGN -> id.assign RHS}	8	{VDECL -> vtype id.semi; ASSIGN -> id.assign RHS}
goto(2, ASSIGN)	{VDECL -> vtype ASSIGN.semi}	9	{VDECL -> vtype ASSIGN.semi}
goto(4, CODE)	{CODE -> FDECL CODE.}	10	{CODE -> FDECL CODE.}
goto(4, VDECL)	{CODE -> VDECL.CODE}	1	
goto(4, FDECL)	{CODE -> FDECL.CODE}	4	
goto(4, CDECL)	{CODE -> CDECL.CODE}	5	
goto(4, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	
goto(4, class)	{CDECL -> class.id lbrace ODECL rbrace}	7	
goto(5, CODE)	{CODE -> CDECL CODE.}	11	{CODE -> CDECL CODE.}
goto(5, VDECL)	{CODE -> VDECL.CODE}	1	
goto(5, FDECL)	{CODE -> FDECL.CODE}	4	
goto(5, CDECL)	{CODE -> CDECL.CODE}	5	
goto(5, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	
goto(5, class)	{CDECL -> class.id lbrace ODECL rbrace}	7	
goto(6, id)	{VDECL -> vtype id.semi; FDECL -> vtype id.lparen ARG rparen lbrace BLOCK RETURN rbrace; ASSIGN -> id.assign RHS}	12	{VDECL -> vtype id.semi; FDECL -> vtype id.lparen ARG rparen lbrace BLOCK RETURN rbrace; ASSIGN -> id.assign RHS}
goto(6, ASSIGN)	{VDECL -> vtype ASSIGN.semi}	9	
goto(7, id)	{CDECL -> class id.lbrace ODECL rbrace}	13	{CDECL -> class id.lbrace ODECL rbrace}
goto(8, semi)	{VDECL -> vtype id semi.}	14	{VDECL -> vtype id semi.}
goto(8, assign)	{ASSIGN -> id assign.RHS}	15	{ASSIGN -> id assign.RHS; RHS -> .EXPR; RHS -> .literal; RHS -> .character; RHS -> .boolstr; EXPR -> .lparen EXPR addsub EXPR rparen; EXPR -> .lparen EXPR multdiv EXPR rparen; EXPR -> .lparen EXPR rparen; EXPR -> .id; EXPR -> .num}
goto(9, semi)	{VDECL -> vtype ASSIGN semi.}	16	{VDECL -> vtype ASSIGN semi.}
goto(12, semi)	{VDECL -> vtype id semi.}	14	
goto(12, lparen)	{FDECL -> vtype id lparen.ARG rparen lbrace BLOCK RETURN rbrace}	17	{FDECL -> vtype id lparen.ARG rparen lbrace BLOCK RETURN rbrace; ARG -> .vtype id MOREARGS; ARG -> .}
goto(12, assign)	{ASSIGN -> id assign.RHS}	15	
goto(13, lbrace)	{CDECL -> class id lbrace.ODECL rbrace}	18	{CDECL -> class id lbrace.ODECL rbrace; ODECL -> .VDECL ODECL; ODECL -> .FDECL ODECL; ODECL -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace}
goto(15, RHS)	{ASSIGN -> id assign RHS.}	19	{ASSIGN -> id assign RHS.}
goto(15, EXPR)	{RHS -> EXPR.}	20	{RHS -> EXPR.}
goto(15, literal)	{RHS -> literal.}	21	{RHS -> literal.}
goto(15, character)	{RHS -> character.}	22	{RHS -> character.}
goto(15, boolstr)	{RHS -> boolstr.}	23	{RHS -> boolstr.}
goto(15, lparen)	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen}	24	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen; EXPR -> .lparen EXPR addsub EXPR rparen; EXPR -> .lparen EXPR multdiv EXPR rparen; EXPR -> .lparen EXPR rparen; EXPR -> .id; EXPR -> .num}
goto(15, id)	{EXPR -> id.}	25	{EXPR -> id.}
goto(15, num)	{EXPR -> num.}	26	{EXPR -> num.}
goto(17, ARG)	{FDECL -> vtype id lparen ARG.rparen lbrace BLOCK RETURN rbrace}	27	{FDECL -> vtype id lparen ARG.rparen lbrace BLOCK RETURN rbrace}
goto(17, vtype)	{ARG -> vtype.id MOREARGS}	28	{ARG -> vtype.id MOREARGS}
goto(18, ODECL)	{CDECL -> class id lbrace ODECL.rbrace}	29	{CDECL -> class id lbrace ODECL.rbrace}
goto(18, VDECL)	{ODECL -> VDECL.ODECL}	30	{ODECL -> VDECL.ODECL; ODECL -> .VDECL ODECL; ODECL -> .FDECL ODECL; ODECL -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace}
goto(18, FDECL)	{ODECL -> FDECL.ODECL}	31	{ODECL -> FDECL.ODECL; ODECL -> .VDECL ODECL; ODECL -> .FDECL ODECL; ODECL -> .; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; FDECL -> .vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace}
goto(18, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	
goto(24, EXPR)	{EXPR -> lparen EXPR.addsub EXPR rparen; EXPR -> lparen EXPR.multdiv EXPR rparen; EXPR -> lparen EXPR.rparen}	32	{EXPR -> lparen EXPR.addsub EXPR rparen; EXPR -> lparen EXPR.multdiv EXPR rparen; EXPR -> lparen EXPR.rparen}
goto(24, lparen)	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen}	24	
goto(24, id)	{EXPR -> id.}	25	
goto(24, num)	{EXPR -> num.}	26	
goto(27, rparen)	{FDECL -> vtype id lparen ARG rparen.lbrace BLOCK RETURN rbrace}	33	{FDECL -> vtype id lparen ARG rparen.lbrace BLOCK RETURN rbrace}
goto(28, id)	{ARG -> vtype id.MOREARGS}	34	{ARG -> vtype id.MOREARGS; MOREARGS -> .comma vtype id MOREARGS; MOREARGS -> .}
goto(29, rbrace)	{CDECL -> class id lbrace ODECL rbrace.}	35	{CDECL -> class id lbrace ODECL rbrace.}
goto(30, ODECL)	{ODECL -> VDECL ODECL.}	36	{ODECL -> VDECL ODECL.}
goto(30, VDECL)	{ODECL -> VDECL.ODECL}	30	
goto(30, FDECL)	{ODECL -> FDECL.ODECL}	31	
goto(30, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	
goto(31, ODECL)	{ODECL -> FDECL ODECL.}	37	{ODECL -> FDECL ODECL.}
goto(31, VDECL)	{ODECL -> VDECL.ODECL}	30	
goto(31, FDECL)	{ODECL -> FDECL.ODECL}	31	
goto(31, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi; FDECL -> vtype.id lparen ARG rparen lbrace BLOCK RETURN rbrace}	6	
goto(32, addsub)	{EXPR -> lparen EXPR addsub.EXPR rparen}	38	{EXPR -> lparen EXPR addsub.EXPR rparen; EXPR -> .lparen EXPR addsub EXPR rparen; EXPR -> .lparen EXPR multdiv EXPR rparen; EXPR -> .lparen EXPR rparen; EXPR -> .id; EXPR -> .num}
goto(32, multdiv)	{EXPR -> lparen EXPR multdiv.EXPR rparen}	39	{EXPR -> lparen EXPR multdiv.EXPR rparen; EXPR -> .lparen EXPR addsub EXPR rparen; EXPR -> .lparen EXPR multdiv EXPR rparen; EXPR -> .lparen EXPR rparen; EXPR -> .id; EXPR -> .num}
goto(32, rparen)	{EXPR -> lparen EXPR rparen.}	40	{EXPR -> lparen EXPR rparen.}
goto(33, lbrace)	{FDECL -> vtype id lparen ARG rparen lbrace.BLOCK RETURN rbrace}	41	{FDECL -> vtype id lparen ARG rparen lbrace.BLOCK RETURN rbrace; BLOCK -> .STMT BLOCK; BLOCK -> .; STMT -> .VDECL; STMT -> .ASSIGN semi; STMT -> .if lparen COND rparen lbrace BLOCK rbrace ELSE; STMT -> .while lparen COND rparen lbrace BLOCK rbrace; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; ASSIGN -> .id assign RHS}
goto(34, MOREARGS)	{ARG -> vtype id MOREARGS.}	42	{ARG -> vtype id MOREARGS.}
goto(34, comma)	{MOREARGS -> comma.vtype id MOREARGS}	43	{MOREARGS -> comma.vtype id MOREARGS}
goto(38, EXPR)	{EXPR -> lparen EXPR addsub EXPR.rparen}	44	{EXPR -> lparen EXPR addsub EXPR.rparen}
goto(38, lparen)	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen}	24	
goto(38, id)	{EXPR -> id.}	25	
goto(38, num)	{EXPR -> num.}	26	
goto(39, EXPR)	{EXPR -> lparen EXPR multdiv EXPR.rparen}	45	{EXPR -> lparen EXPR multdiv EXPR.rparen}
goto(39, lparen)	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen}	24	
goto(39, id)	{EXPR -> id.}	25	
goto(39, num)	{EXPR -> num.}	26	
goto(41, BLOCK)	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK.RETURN rbrace}	46	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK.RETURN rbrace; RETURN -> .return RHS semi}
goto(41, STMT)	{BLOCK -> STMT.BLOCK}	47	{BLOCK -> STMT.BLOCK; BLOCK -> .STMT BLOCK; BLOCK -> .; STMT -> .VDECL; STMT -> .ASSIGN semi; STMT -> .if lparen COND rparen lbrace BLOCK rbrace ELSE; STMT -> .while lparen COND rparen lbrace BLOCK rbrace; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; ASSIGN -> .id assign RHS}
goto(41, VDECL)	{STMT -> VDECL.}	48	{STMT -> VDECL.}
goto(41, ASSIGN)	{STMT -> ASSIGN.semi}	49	{STMT -> ASSIGN.semi}
goto(41, if)	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}	50	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}
goto(41, while)	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}	51	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}
goto(41, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	
goto(41, id)	{ASSIGN -> id.assign RHS}	52	{ASSIGN -> id.assign RHS}
goto(43, vtype)	{MOREARGS -> comma vtype.id MOREARGS}	53	{MOREARGS -> comma vtype.id MOREARGS}
goto(44, rparen)	{EXPR -> lparen EXPR addsub EXPR rparen.}	54	{EXPR -> lparen EXPR addsub EXPR rparen.}
goto(45, rparen)	{EXPR -> lparen EXPR multdiv EXPR rparen.}	55	{EXPR -> lparen EXPR multdiv EXPR rparen.}
goto(46, RETURN)	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK RETURN.rbrace}	56	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK RETURN.rbrace}
goto(46, return)	{RETURN -> return.RHS semi}	57	{RETURN -> return.RHS semi; RHS -> .EXPR; RHS -> .literal; RHS -> .character; RHS -> .boolstr; EXPR -> .lparen EXPR addsub EXPR rparen; EXPR -> .lparen EXPR multdiv EXPR rparen; EXPR -> .lparen EXPR rparen; EXPR -> .id; EXPR -> .num}
goto(47, BLOCK)	{BLOCK -> STMT BLOCK.}	58	{BLOCK -> STMT BLOCK.}
goto(47, STMT)	{BLOCK -> STMT.BLOCK}	47	
goto(47, VDECL)	{STMT -> VDECL.}	48	
goto(47, ASSIGN)	{STMT -> ASSIGN.semi}	49	
goto(47, if)	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}	50	
goto(47, while)	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}	51	
goto(47, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	
goto(47, id)	{ASSIGN -> id.assign RHS}	52	
goto(49, semi)	{STMT -> ASSIGN semi.}	59	{STMT -> ASSIGN semi.}
goto(50, lparen)	{STMT -> if lparen.COND rparen lbrace BLOCK rbrace ELSE}	60	{STMT -> if lparen.COND rparen lbrace BLOCK rbrace ELSE; COND -> .lparen COND comp COND rparen; COND -> .boolstr}
goto(51, lparen)	{STMT -> while lparen.COND rparen lbrace BLOCK rbrace}	61	{STMT -> while lparen.COND rparen lbrace BLOCK rbrace; COND -> .lparen COND comp COND rparen; COND -> .boolstr}
goto(52, assign)	{ASSIGN -> id assign.RHS}	15	
goto(53, id)	{MOREARGS -> comma vtype id.MOREARGS}	62	{MOREARGS -> comma vtype id.MOREARGS; MOREARGS -> .comma vtype id MOREARGS; MOREARGS -> .}
goto(56, rbrace)	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace.}	63	{FDECL -> vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace.}
goto(57, RHS)	{RETURN -> return RHS.semi}	64	{RETURN -> return RHS.semi}
goto(57, EXPR)	{RHS -> EXPR.}	20	
goto(57, literal)	{RHS -> literal.}	21	
goto(57, character)	{RHS -> character.}	22	
goto(57, boolstr)	{RHS -> boolstr.}	23	
goto(57, lparen)	{EXPR -> lparen.EXPR addsub EXPR rparen; EXPR -> lparen.EXPR multdiv EXPR rparen; EXPR -> lparen.EXPR rparen}	24	
goto(57, id)	{EXPR -> id.}	25	
goto(57, num)	{EXPR -> num.}	26	
goto(60, COND)	{STMT -> if lparen COND.rparen lbrace BLOCK rbrace ELSE}	65	{STMT -> if lparen COND.rparen lbrace BLOCK rbrace ELSE}
goto(60, lparen)	{COND -> lparen.COND comp COND rparen}	66	{COND -> lparen.COND comp COND rparen; COND -> .lparen COND comp COND rparen; COND -> .boolstr}
goto(60, boolstr)	{COND -> boolstr.}	67	{COND -> boolstr.}
goto(61, COND)	{STMT -> while lparen COND.rparen lbrace BLOCK rbrace}	68	{STMT -> while lparen COND.rparen lbrace BLOCK rbrace}
goto(61, lparen)	{COND -> lparen.COND comp COND rparen}	66	
goto(61, boolstr)	{COND -> boolstr.}	67	
goto(62, MOREARGS)	{MOREARGS -> comma vtype id MOREARGS.}	69	{MOREARGS -> comma vtype id MOREARGS.}
goto(62, comma)	{MOREARGS -> comma.vtype id MOREARGS}	43	
goto(64, semi)	{RETURN -> return RHS semi.}	70	{RETURN -> return RHS semi.}
goto(65, rparen)	{STMT -> if lparen COND rparen.lbrace BLOCK rbrace ELSE}	71	{STMT -> if lparen COND rparen.lbrace BLOCK rbrace ELSE}
goto(66, COND)	{COND -> lparen COND.comp COND rparen}	72	{COND -> lparen COND.comp COND rparen}
goto(66, lparen)	{COND -> lparen.COND comp COND rparen}	66	
goto(66, boolstr)	{COND -> boolstr.}	67	
goto(68, rparen)	{STMT -> while lparen COND rparen.lbrace BLOCK rbrace}	73	{STMT -> while lparen COND rparen.lbrace BLOCK rbrace}
goto(71, lbrace)	{STMT -> if lparen COND rparen lbrace.BLOCK rbrace ELSE}	74	{STMT -> if lparen COND rparen lbrace.BLOCK rbrace ELSE; BLOCK -> .STMT BLOCK; BLOCK -> .; STMT -> .VDECL; STMT -> .ASSIGN semi; STMT -> .if lparen COND rparen lbrace BLOCK rbrace ELSE; STMT -> .while lparen COND rparen lbrace BLOCK rbrace; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; ASSIGN -> .id assign RHS}
goto(72, comp)	{COND -> lparen COND comp.COND rparen}	75	{COND -> lparen COND comp.COND rparen; COND -> .lparen COND comp COND rparen; COND -> .boolstr}
goto(73, lbrace)	{STMT -> while lparen COND rparen lbrace.BLOCK rbrace}	76	{STMT -> while lparen COND rparen lbrace.BLOCK rbrace; BLOCK -> .STMT BLOCK; BLOCK -> .; STMT -> .VDECL; STMT -> .ASSIGN semi; STMT -> .if lparen COND rparen lbrace BLOCK rbrace ELSE; STMT -> .while lparen COND rparen lbrace BLOCK rbrace; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; ASSIGN -> .id assign RHS}
goto(74, BLOCK)	{STMT -> if lparen COND rparen lbrace BLOCK.rbrace ELSE}	77	{STMT -> if lparen COND rparen lbrace BLOCK.rbrace ELSE}
goto(74, STMT)	{BLOCK -> STMT.BLOCK}	47	
goto(74, VDECL)	{STMT -> VDECL.}	48	
goto(74, ASSIGN)	{STMT -> ASSIGN.semi}	49	
goto(74, if)	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}	50	
goto(74, while)	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}	51	
goto(74, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	
goto(74, id)	{ASSIGN -> id.assign RHS}	52	
goto(75, COND)	{COND -> lparen COND comp COND.rparen}	78	{COND -> lparen COND comp COND.rparen}
goto(75, lparen)	{COND -> lparen.COND comp COND rparen}	66	
goto(75, boolstr)	{COND -> boolstr.}	67	
goto(76, BLOCK)	{STMT -> while lparen COND rparen lbrace BLOCK.rbrace}	79	{STMT -> while lparen COND rparen lbrace BLOCK.rbrace}
goto(76, STMT)	{BLOCK -> STMT.BLOCK}	47	
goto(76, VDECL)	{STMT -> VDECL.}	48	
goto(76, ASSIGN)	{STMT -> ASSIGN.semi}	49	
goto(76, if)	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}	50	
goto(76, while)	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}	51	
goto(76, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	
goto(76, id)	{ASSIGN -> id.assign RHS}	52	
goto(77, rbrace)	{STMT -> if lparen COND rparen lbrace BLOCK rbrace.ELSE}	80	{STMT -> if lparen COND rparen lbrace BLOCK rbrace.ELSE; ELSE -> .else lbrace BLOCK rbrace; ELSE -> .}
goto(78, rparen)	{COND -> lparen COND comp COND rparen.}	81	{COND -> lparen COND comp COND rparen.}
goto(79, rbrace)	{STMT -> while lparen COND rparen lbrace BLOCK rbrace.}	82	{STMT -> while lparen COND rparen lbrace BLOCK rbrace.}
goto(80, ELSE)	{STMT -> if lparen COND rparen lbrace BLOCK rbrace ELSE.}	83	{STMT -> if lparen COND rparen lbrace BLOCK rbrace ELSE.}
goto(80, else)	{ELSE -> else.lbrace BLOCK rbrace}	84	{ELSE -> else.lbrace BLOCK rbrace}
goto(84, lbrace)	{ELSE -> else lbrace.BLOCK rbrace}	85	{ELSE -> else lbrace.BLOCK rbrace; BLOCK -> .STMT BLOCK; BLOCK -> .; STMT -> .VDECL; STMT -> .ASSIGN semi; STMT -> .if lparen COND rparen lbrace BLOCK rbrace ELSE; STMT -> .while lparen COND rparen lbrace BLOCK rbrace; VDECL -> .vtype id semi; VDECL -> .vtype ASSIGN semi; ASSIGN -> .id assign RHS}
goto(85, BLOCK)	{ELSE -> else lbrace BLOCK.rbrace}	86	{ELSE -> else lbrace BLOCK.rbrace}
goto(85, STMT)	{BLOCK -> STMT.BLOCK}	47	
goto(85, VDECL)	{STMT -> VDECL.}	48	
goto(85, ASSIGN)	{STMT -> ASSIGN.semi}	49	
goto(85, if)	{STMT -> if.lparen COND rparen lbrace BLOCK rbrace ELSE}	50	
goto(85, while)	{STMT -> while.lparen COND rparen lbrace BLOCK rbrace}	51	
goto(85, vtype)	{VDECL -> vtype.id semi; VDECL -> vtype.ASSIGN semi}	2	
goto(85, id)	{ASSIGN -> id.assign RHS}	52	
goto(86, rbrace)	{ELSE -> else lbrace BLOCK rbrace.}	87	{ELSE -> else lbrace BLOCK rbrace.}

LR table

ACTION

State	vtype	id	semi	assign	literal	character	boolstr	lparen	addsub	rparen	multdiv	num	lbrace	rbrace	comma	if	while	comp	else	return	class	$
0	s2																					
1	s6																				s7	r_{3}
2		s8																				
3																						acc
4	s6																				s7	r_{3}
5	s6																				s7	r_{3}
6		s12																				
7		s13																				
8			s14	s15																		
9			s16																			
10																						r_{1}
11																						r_{2}
12			s14	s15				s17														
13													s18									
14	r_{4}	r_{4}												r_{4}		r_{4}	r_{4}			r_{4}	r_{4}	r_{4}
15		s25			s21	s22	s23	s24				s26										
16	r_{5}	r_{5}												r_{5}		r_{5}	r_{5}			r_{5}	r_{5}	r_{5}
17	s28									r_{18}												
18	s6													r_{35}								
19			r_{6}																			
20			r_{7}																			
21			r_{8}																			
22			r_{9}																			
23			r_{10}																			
24		s25						s24				s26										
25			r_{14}						r_{14}	r_{14}	r_{14}											
26			r_{15}						r_{15}	r_{15}	r_{15}											
27										s33												
28		s34																				
29														s35								
30	s6													r_{35}								
31	s6													r_{35}								
32									s38	s40	s39											
33													s41									
34										r_{20}					s43							
35	r_{32}																				r_{32}	r_{32}
36														r_{33}								
37														r_{34}								
38		s25						s24				s26										
39		s25						s24				s26										
40			r_{13}						r_{13}	r_{13}	r_{13}											
41	s2	s52												r_{22}		s50	s51			r_{22}		
42										r_{17}												
43	s53																					
44										s54												
45										s55												
46																				s57		
47	s2	s52												r_{22}		s50	s51			r_{22}		
48	r_{23}	r_{23}												r_{23}		r_{23}	r_{23}			r_{23}		
49			s59																			
50								s60														
51								s61														
52				s15																		
53		s62																				
54			r_{11}						r_{11}	r_{11}	r_{11}											
55			r_{12}						r_{12}	r_{12}	r_{12}											
56														s63								
57		s25			s21	s22	s23	s24				s26										
58														r_{21}						r_{21}		
59	r_{24}	r_{24}												r_{24}		r_{24}	r_{24}			r_{24}		
60							s67	s66														
61							s67	s66														
62										r_{20}					s43							
63	r_{16}													r_{16}							r_{16}	r_{16}
64			s70																			
65										s71												
66							s67	s66														
67										r_{28}								r_{28}				
68										s73												
69										r_{19}												
70														r_{31}								
71													s74									
72																		s75				
73													s76									
74	s2	s52												r_{22}		s50	s51			r_{22}		
75							s67	s66														
76	s2	s52												r_{22}		s50	s51			r_{22}		
77														s80								
78										s81												
79														s82								
80	r_{30}	r_{30}												r_{30}		r_{30}	r_{30}		s84	r_{30}		
81										r_{27}								r_{27}				
82	r_{26}	r_{26}												r_{26}		r_{26}	r_{26}			r_{26}		
83	r_{25}	r_{25}												r_{25}		r_{25}	r_{25}			r_{25}		
84													s85									
85	s2	s52												r_{22}		s50	s51			r_{22}		
86														s87								
87	r_{29}	r_{29}												r_{29}		r_{29}	r_{29}			r_{29}		

GOTO

State	CODE	VDECL	ASSIGN	RHS	EXPR	FDECL	ARG	MOREARGS	BLOCK	STMT	COND	ELSE	RETURN	CDECL	ODECL
0		1													
1	3	1				4								5	
2			9												
3															
4	10	1				4								5	
5	11	1				4								5	
6			9												
7															
8															
9															
10															
11															
12															
13															
14															
15				19	20										
16															
17							27								
18		30				31									29
19															
20															
21															
22															
23															
24					32										
25															
26															
27															
28															
29															
30		30				31									36
31		30				31									37
32															
33															
34								42							
35															
36															
37															
38					44										
39					45										
40															
41		48	49						46	47					
42															
43															
44															
45															
46													56		
47		48	49						58	47					
48															
49															
50															
51															
52															
53															
54															
55															
56															
57				64	20										
58															
59															
60											65				
61											68				
62								69							
63															
64															
65															
66											72				
67															
68															
69															
70															
71															
72															
73															
74		48	49						77	47					
75											78				
76		48	49						79	47					
77															
78															
79															
80												83			
81															
82															
83															
84															
85		48	49						86	47					
86															
87															

Input (tokens): id + id * id

Maximum number of steps: 100

PARSE

Trace				Tree
Step	Stack	Input	Action	
1	0	id + id * id $		