INFO-F-403 : Language theory and compiling
Rapport projet partie 2 - Grammaire

Simon Picard
Arnaud Rosette

17 décembre 2014

1 Grammaire

[1]	<Program>	→	<InstructionList>
[2]	<InstructionList>	→	<IdentifierInstruction> END_OF_INSTRUCTION <InstructionList>
[3]		→	<ConstDefinition> END_OF_INSTRUCTION <InstructionList>
[4]		→	<Block> END_OF_INSTRUCTION <InstructionList>
[5]		→	<Loop> END_OF_INSTRUCTION <InstructionList>
[6]		→	<BuiltInFunctionCall> END_OF_INSTRUCTION <InstructionList>
[7]		→	<FunctionDefinition> END_OF_INSTRUCTION <InstructionList>
[8]		→	END_OF_INSTRUCTION <InstructionList>
[9]		→	EPSILON_VALUE
[10]	<IdentifierInstruction>	→	IDENTIFIER <IdentifierInstructionTail>
[11]	<IdentifierInstructionTail>	→	<AssignationTail>
[12]		→	TYPE_DEFINITION <Type>
[13]		→	<FunctionCallTail>
[14]	<AssignationTail>	→	ASSIGNATION <Expression>
[15]		→	COMMA IDENTIFIER <AssignationTail> COMMA <Expression>
[16]	<ConstDefinition>	→	CONST IDENTIFIER <AssignationTail>
[17]	<Block>	→	LET IDENTIFIER <AssignationTail> END_OF_INSTRUCTION <InstructionList> END
[18]	<Loop>	→	<If>
[19]		→	WHILE <Expression> END_OF_INSTRUCTION <InstructionList> END
[20]		→	FOR IDENTIFIER ASSIGNATION <Expression> TERNARY_ELSE <Expression> <ForTail>
[21]	<ForTail>	→	END_OF_INSTRUCTION <InstructionList> END
[22]		→	TERNARY_ELSE <Expression> END_OF_INSTRUCTION <InstructionList> END
[23]	<Type>	→	BOOLEAN_TYPE
[24]		→	REAL_TYPE
[25]		→	INTEGER_TYPE
[26]	<Expression>	→	<BinaryExpression> <TernaryIfExpression>
[27]	<TernaryIfExpression>	→	TERNARY_IF <Expression> <TernaryElseExpression>
[28]		→	EPSILON_VALUE
[29]	<TernaryElseExpression>	→	TERNARY_ELSE <Expression>
[30]	<AtomicExpression>	→	<AtomicIdentifierExpression>

[31]		→	INTEGER
[32]		→	REAL
[33]		→	BOOLEAN
[34]		→	<BuiltInFunctionCall>
[35]	<AtomicIdentifierExpression>	→	IDENTIFIER <AtomicIdentifierExpressionTail>
[36]	<AtomicIdentifierExpressionTail>	→	<FunctionCallTail>
[37]		→	EPSILON_VALUE
[38]	<UnaryExpression>	→	NEGATION <UnaryExpression>
[39]		→	<UnaryBitwiseNotExpression>
[40]	<UnaryBitwiseNotExpression>	→	BITWISE_NOT <UnaryBitwiseNotExpression>
[41]		→	<UnaryMinusPlusExpression>
[42]	<UnaryMinusPlusExpression>	→	MINUS <UnaryMinusPlusExpression>
[43]		→	PLUS <UnaryMinusPlusExpression>
[44]		→	<UnaryAtomicExpression>
[45]	<UnaryAtomicExpression>	→	<AtomicExpression>
[46]		→	LEFT_PARENTHESIS <Expression> RIGHT_PARENTHESIS
[47]	<BinaryExpression>	→	<BinaryLazyOrExpression> <BinaryExpression'>
[48]	<BinaryExpression'>	→	LAZY_OR <BinaryLazyOrExpression> <BinaryExpression'>
[49]		→	EPSILON_VALUE
[50]	<BinaryLazyOrExpression>	→	<BinaryLazyAndExpression> <BinaryLazyOrExpression'>
[51]	<BinaryLazyOrExpression'>	→	LAZY_AND <BinaryLazyAndExpression> <BinaryLazyOrExpression'>
[52]		→	EPSILON_VALUE
[53]	<BinaryLazyAndExpression>	→	<BinaryNumericExpression> <BinaryLazyAndExpression'>
[54]	<BinaryLazyAndExpression'>	→	GREATER_THAN <BinaryNumericExpression> <BinaryLazyAndExpression'>
[55]		→	LESS_THAN <BinaryNumericExpression> <BinaryLazyAndExpression'>
[56]		→	GREATER_OR_EQUALS_THAN <BinaryNumericExpression> <BinaryLazyAndExpression'>
[57]		→	LESS_OR_EQUALS_THAN <BinaryNumericExpression> <BinaryLazyAndExpression'>
[58]		→	EQUALITY <BinaryNumericExpression> <BinaryLazyAndExpression'>
[59]		→	INEQUALITY <BinaryNumericExpression> <BinaryLazyAndExpression'>
[60]		→	EPSILON_VALUE
[61]	<BinaryNumericExpression>	→	<BinaryTermExpression> <BinaryNumericExpression'>
[62]	<BinaryNumericExpression'>	→	PLUS <BinaryTermExpression> <BinaryNumericExpression'>
[63]		→	MINUS <BinaryTermExpression> <BinaryNumericExpression'>
[64]		→	BITWISE_OR <BinaryTermExpression> <BinaryNumericExpression'>
[65]		→	BITWISE_XOR <BinaryTermExpression> <BinaryNumericExpression'>
[66]		→	EPSILON_VALUE
[67]	<BinaryTermExpression>	→	<BinaryShiftedExpression> <BinaryTermExpression'>

[68]	<BinaryTermExpression'>	→	ARITHMETIC_SHIFT_LEFT <BinaryShiftedExpression> <BinaryTermExpression'>
[69]		→	ARITHMETIC_SHIFT_RIGHT <BinaryShiftedExpression> <BinaryTermExpression'>
[70]		→	EPSILON_VALUE
[71]	<BinaryShiftedExpression>	→	<BinaryFactorExpression> <BinaryShiftedExpression'>
[72]	<BinaryShiftedExpression'>	→	TIMES <BinaryFactorExpression> <BinaryShiftedExpression'>
[73]		→	DIVIDE <BinaryFactorExpression> <BinaryShiftedExpression'>
[74]		→	REMAINDER <BinaryFactorExpression> <BinaryShiftedExpression'>
[75]		→	BITWISE_AND <BinaryFactorExpression> <BinaryShiftedExpression'>
[76]		→	INVERSE_DIVIDE <BinaryFactorExpression> <BinaryShiftedExpression'>
[77]		→	EPSILON_VALUE
[78]	<BinaryFactorExpression>	→	<UnaryExpression> <BinaryFactorExpression'>
[79]	<BinaryFactorExpression'>	→	POWER <UnaryExpression> <BinaryFactorExpression'>
[80]		→	EPSILON_VALUE
[81]	<If>	→	IF <Expression> END_OF_INSTRUCTION <InstructionList> <IfEnd>
[82]	<IfEnd>	→	ELSE_IF <Expression> END_OF_INSTRUCTION <InstructionList> <IfEnd>
[83]		→	ELSE <InstructionList> END
[84]		→	END
[85]	<BuiltInFunctionCall>	→	READ_REAL LEFT_PARENTHESIS RIGHT_PARENTHESIS
[86]		→	READ_INTEGER LEFT_PARENTHESIS RIGHT_PARENTHESIS
[87]		→	INTEGER_CAST LEFT_PARENTHESIS <Expression> RIGHT_PARENTHESIS
[88]		→	REAL_CAST LEFT_PARENTHESIS <Expression> RIGHT_PARENTHESIS
[89]		→	BOOLEAN_CAST LEFT_PARENTHESIS <Expression> RIGHT_PARENTHESIS
[90]		→	PRINTLN LEFT_PARENTHESIS <Expression> RIGHT_PARENTHESIS
[91]	<FunctionCallTail>	→	LEFT_PARENTHESIS <Parameter> RIGHT_PARENTHESIS
[92]	<Parameter>	→	<Expression> <ParameterTail>
[93]		→	EPSILON_VALUE
[94]	<ParameterTail>	→	COMMA <Expression> <ParameterTail>
[95]		→	EPSILON_VALUE
[96]	<FunctionDefinition>	→	FUNCTION IDENTIFIER LEFT_PARENTHESIS <Argument> RIGHT_PARENTHESIS <InstructionList> <FunctionDefinitionEnd>
[97]	<FunctionDefinitionEnd>	→	RETURN <Expression> END
[98]		→	END
[99]	<Argument>	→	IDENTIFIER TYPE_DEFINITION <Type> <ArgumentTail>
[100]		→	EPSILON_VALUE

[101] <ArgumentTail> → COMMA IDENTIFIER TYPE_DEFINITION <Type> <ArgumentTail>

[102] → EPSILON_VALUE

2 First et Follow set

Variable	First	Follow
<Program>	BOOLEAN_CAST, PRINTLN, FOR EPSILON_VALUE INTEGER_CAST, FUNCTION END_OF_INSTRUCTION CONST, READ_INTEGER, LET WHILE, IDENTIFIER READ_REAL, REAL_CAST, IF	
<InstructionList>	BOOLEAN_CAST, PRINTLN, FOR EPSILON_VALUE INTEGER_CAST, FUNCTION END_OF_INSTRUCTION CONST, READ_INTEGER, LET WHILE, IDENTIFIER READ_REAL, REAL_CAST, IF	RETURN, ELSE_IF, ELSE, END
<IdentifierInstruction>	IDENTIFIER	END_OF_INSTRUCTION
<IdentifierInstructionTail>	ASSIGNATION, COMMA LEFT_PARENTHESIS TYPE_DEFINITION	END_OF_INSTRUCTION
<AssignationTail>	ASSIGNATION, COMMA	COMMA END_OF_INSTRUCTION
<ConstDefinition>	CONST	END_OF_INSTRUCTION
<Block>	LET	END_OF_INSTRUCTION
<Loop>	FOR, WHILE, IF	END_OF_INSTRUCTION
<ForTail>	END_OF_INSTRUCTION TERNARY_ELSE	END_OF_INSTRUCTION
<Type>	INTEGER_TYPE BOOLEAN_TYPE, REAL_TYPE	COMMA, RIGHT_PARENTHESIS END_OF_INSTRUCTION
<Expression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_ELSE, END
<TernaryIfExpression>	TERNARY_IF EPSILON_VALUE	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_ELSE, END
<TernaryElseExpression>	TERNARY_ELSE	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_ELSE, END

<AtomicExpression>	BOOLEAN_CAST, PRINTLN REAL, READ_INTEGER INTEGER_CAST, IDENTIFIER READ_REAL, REAL_CAST BOOLEAN, INTEGER	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<AtomicIdentifierExpression>	IDENTIFIER	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<AtomicIdentifierExpressionTail>	LEFT_PARENTHESIS EPSILON_VALUE	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS

<UnaryExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<UnaryBitwiseNotExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN INTEGER_CAST, BOOLEAN MINUS, LEFT_PARENTHESIS REAL, READ_INTEGER IDENTIFIER, READ_REAL REAL_CAST, INTEGER, PLUS	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<UnaryMinusPlusExpression>	BOOLEAN_CAST, PRINTLN INTEGER_CAST, BOOLEAN MINUS, LEFT_PARENTHESIS REAL, READ_INTEGER IDENTIFIER, READ_REAL REAL_CAST, INTEGER, PLUS	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS

<UnaryAtomicExpression>	LEFT_PARENTHESIS BOOLEAN_CAST, PRINTLN REAL, READ_INTEGER INTEGER_CAST, IDENTIFIER READ_REAL, REAL_CAST BOOLEAN, INTEGER	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<BinaryExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_IF, TERNARY_ELSE END
<BinaryExpression'>	EPSILON_VALUE, LAZY_OR	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_IF, TERNARY_ELSE END
<BinaryLazyOrExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_IF, TERNARY_ELSE END, LAZY_OR
<BinaryLazyOrExpression'>	LAZY_AND, EPSILON_VALUE	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS TERNARY_IF, TERNARY_ELSE END, LAZY_OR
<BinaryLazyAndExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS LAZY_AND, TERNARY_IF TERNARY_ELSE, END, LAZY_OR
<BinaryLazyAndExpression'>	LESS_OR_EQUALS_THAN GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, INEQUALITY EPSILON_VALUE, LESS_THAN	COMMA END_OF_INSTRUCTION RIGHT_PARENTHESIS LAZY_AND, TERNARY_IF TERNARY_ELSE, END, LAZY_OR

<BinaryNumericExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	LESS_OR_EQUALS_THAN COMMA, RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END
<BinaryNumericExpression'>	BITWISE_OR EPSILON_VALUE BITWISE_XOR, PLUS, MINUS	LESS_OR_EQUALS_THAN COMMA, RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END
<BinaryTermExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	LESS_OR_EQUALS_THAN COMMA, BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR, BITWISE_XOR MINUS END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END PLUS
<BinaryTermExpression'>	ARITHMETIC_SHIFT_LEFT EPSILON_VALUE ARITHMETIC_SHIFT_RIGHT	LESS_OR_EQUALS_THAN COMMA, BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR, BITWISE_XOR MINUS END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END PLUS
<BinaryShiftedExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	LESS_OR_EQUALS_THAN COMMA ARITHMETIC_SHIFT_LEFT BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR, BITWISE_XOR MINUS END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END ARITHMETIC_SHIFT_RIGHT PLUS

<BinaryShiftedExpression'>	INVERSE_DIVIDE, TIMES REMAINDER, EPSILON_VALUE BITWISE_AND, DIVIDE	LESS_OR_EQUALS_THAN COMMA ARITHMETIC_SHIFT_LEFT BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, LESS_THAN LAZY_OR, BITWISE_XOR MINUS END_OF_INSTRUCTION GREATER_THAN GREATER_OR_EQUALS_THAN EQUALITY, LAZY_AND, END ARITHMETIC_SHIFT_RIGHT PLUS
<BinaryFactorExpression>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, INTEGER_CAST BOOLEAN, MINUS LEFT_PARENTHESIS, REAL READ_INTEGER, IDENTIFIER READ_REAL, REAL_CAST INTEGER, PLUS	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<BinaryFactorExpression'>	POWER, EPSILON_VALUE	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<If>	IF	END_OF_INSTRUCTION
<IfEnd>	ELSE_IF, ELSE, END	END_OF_INSTRUCTION

<BuiltInFunctionCall>	BOOLEAN_CAST, PRINTLN READ_INTEGER INTEGER_CAST, READ_REAL REAL_CAST	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<FunctionCallTail>	LEFT_PARENTHESIS	LESS_OR_EQUALS_THAN INVERSE_DIVIDE BITWISE_OR RIGHT_PARENTHESIS INEQUALITY, TERNARY_IF TERNARY_ELSE, DIVIDE MINUS, GREATER_THAN LAZY_AND ARITHMETIC_SHIFT_RIGHT COMMA ARITHMETIC_SHIFT_LEFT TIMES, POWER, BITWISE_AND LESS_THAN, LAZY_OR BITWISE_XOR, REMAINDER END_OF_INSTRUCTION GREATER_OR_EQUALS_THAN EQUALITY, END, PLUS
<Parameter>	BOOLEAN_CAST BITWISE_NOT, PRINTLN NEGATION, EPSILON_VALUE INTEGER_CAST, BOOLEAN MINUS, LEFT_PARENTHESIS REAL, READ_INTEGER IDENTIFIER, READ_REAL REAL_CAST, INTEGER, PLUS	RIGHT_PARENTHESIS
<ParameterTail>	COMMA, EPSILON_VALUE	RIGHT_PARENTHESIS
<FunctionDefinition>	FUNCTION	END_OF_INSTRUCTION
<FunctionDefinitionEnd>	RETURN, END	END_OF_INSTRUCTION
<Argument>	EPSILON_VALUE IDENTIFIER	RIGHT_PARENTHESIS
<ArgumentTail>	COMMA, EPSILON_VALUE	RIGHT_PARENTHESIS

3 Action Table

	EPSILON_VALUE	IDENTIFIER	TYPE_DEFINITION	CONST	END	ASSIGNATION	END_OF_INSTRUCTION	BOOLEAN_TYPE	REAL_TYPE	INTEGER_TYPE	WHILE	FOR	TERNARY_ELSE	COMMA	TERNARY_IF	INTEGER	REAL	BOOLEAN	NEGATION	BITWISE_NOT	MINUS	PLUS	LEFT_PARENTHESIS	RIGHT_PARENTHESIS	LAZY_OR	LAZY_AND	GREATER_THAN	LESS_THAN	GREATER_OR_EQUALS_THAN	LESS_OR_EQUALS_THAN	EQUALITY	INEQUALITY	BITWISE_OR	BITWISE_XOR	ARITHMETIC_SHIFT_LEFT	ARITHMETIC_SHIFT_RIGHT	TIMES	DIVIDE	REMAINDER	BITWISE_AND	INVERSE_DIVIDE	POWER	IF	ELSE_IF	ELSE	LET	BOOLEAN_TYPE	REAL_TYPE	INTEGER_TYPE	READ_REAL	READ_INTEGER	INTEGER_CAST	REAL_CAST	BOOLEAN_CAST	PRINTLN	FUNCTION	RETURN
<Program>	1	1		1			1				1	1																														1								1	1	1	1	1	1	1	
<InstructionList>	9	2		3	9		8				5	5																														5	9	9	4				6	6	6	6	6	6	7	9
<IdentifierInstruction>		10																																																							
<IdentifierInstructionTail>			12			11								11								13																																			
<AssignationTail>						14								15																																											
<ConstDefinition>				16																																																				
<Block>																																														17												
<Loop>											19	20																														18															
<ForTail>							21						22																																												
<Type>								23	24	25																																						23	24	25							
<Expression>		26														26	26	26	26	26	26	26	26																														26	26	26	26	26	26		
<TernaryIfExpression>	28				28		28						28	28	27									28																																			
<TernaryElseExpression>													29																																												
<AtomicExpression>		30														31	32	33																																				34	34	34	34	34	34		
<AtomicIdentifierExpression>		35																																																							
<AtomicIdentifierExpressionTail>	37				37		37						37	37	37						37	37	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37																
<UnaryExpression>		39														39	39	39	38	39	39	39	39																														39	39	39	39	39	39		
<UnaryBitwiseNotExpression>		41														41	41	41		40	41	41	41																														41	41	41	41	41	41		
<UnaryMinusPlusExpression>		44														44	44	44			42	43	44																														44	44	44	44	44	44		
<UnaryAtomicExpression>		45														45	45	45					46																														45	45	45	45	45	45		
<BinaryExpression>		47														47	47	47	47	47	47	47	47																														47	47	47	47	47	47		
<BinaryExpression'>	49				49		49						49	49	49									49	48																																			
<BinaryLazyOrExpression>		50														50	50	50	50	50	50	50	50																														50	50	50	50	50	50		
<BinaryLazyOrExpression'>	52				52		52						52	52	52									52	52	51																																		
<BinaryLazyAndExpression>		53														53	53	53	53	53	53	53	53																														53	53	53	53	53	53		
<BinaryLazyAndExpression'>	60				60		60						60	60	60									60	60	60	54	55	56	57	58	59																											
<BinaryNumericExpression>		61														61	61	61	61	61	61	61	61																														61	61	61	61	61	61		
<BinaryNumericExpression'>	66				66		66						66	66	66						63	62		66	66	66	66	66	66	66	66	66	64	65																							
<BinaryTermExpression>		67														67	67	67	67	67	67	67	67																														67	67	67	67	67	67		
<BinaryTermExpression'>	70				70		70						70	70	70						70	70		70	70	70	70	70	70	70	70	70	70	70	68	69																				
<BinaryShiftedExpression>		71														71	71	71	71	71	71	71	71																														71	71	71	71	71	71		
<BinaryShiftedExpression'>	77				77		77						77	77	77						77	77		77	77	77	77	77	77	77	77	77	77	77	77	77	77	72	73	74	75	76																
<BinaryFactorExpression>		78														78	78	78	78	78	78	78	78																														78	78	78	78	78	78		
<BinaryFactorExpression'>	80				80		80						80	80	80						80	80		80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	79																
<If>																																												81												
<IfEnd>					84																																								82	83											
<BuiltInFunctionCall>																																																					85	86	87	88	89	90		
<FunctionCallTail>																							91																																			
<Parameter>	93	92														92	92	92	92	92	92	92	92	93																												92	92	92	92	92	92		
<ParameterTail>	95													94										95																																		
<FunctionDefinition>																																																								96	
<FunctionDefinitionEnd>					98																																																			97
<Argument>	100	99																						100																																		
<ArgumentTail>	102													101										102																																		