Group- 15		
Aman Sanghi- 2016A7PS0024P		
Sanchit Shrivastava- 2016A7PS0072P		
Sarthak Agarwal- 2016A7PS0135P		
Alex Mathai- 2016A7PS0339P		
Rule No.	**Grammar Rule**	**Abstract Syntax Tree Formation Rule**
1	<program> ===> <otherFunctions> <mainFunction>	<program>.node = new node(1,<otherFunctions>.list, <mainFunction>.node)
2	<mainFunction> ===> TK_MAIN <stmts> TK_END	<mainFunction>.node = <stmts>.node
3.1	<otherFunctions> ===> <function> <otherFunctions1>	<otherFunctions>.list = insert at head(<function>.node, <otherFunctions1>.list)
3.2	<otherFunctions> ===> eps	<otherFunctions>.list = NULL
4	<function> ===> TK_FUNID <input_par> <output_par> TK_SEM <stmts> TK_END	<function>.node = new node(4,TK_FUNID,<input_par>.list, <output_par>.list, <stmts>.list)
5	<input_par> ===> TK_INPUT TK_PARAMETER TK_LIST TK_SQL <parameter_list> TK_SQR	<input_par>.list = <parameter_list>.list
6.1	<output_par> ===> TK_OUTPUT TK_PARAMETER TK_LIST TK_SQL <parameter_list> TK_SQR	<output_par>.list = <parameter_list>.list
6.2	<output_par> ===> eps	<output_par>.list = NULL
7	<parameter_list> ===> <dataType> TK_ID <remaining_list>	<parameter_list>.list = insert at head(new node(7,<dataType>.node,TK_ID),<remaining_list>.list)
8.1	<dataType> ===> <primitiveDatatype>	<dataType>.node = <primitiveDatatype>.node
8.2	<dataType> ===> <constructedDatatype>	<dataType>.node = <constructedDatatype>.node
9.1	<primitiveDatatype> ===> TK_INT	<primitiveDatatype>.node = new leaf(TK_INT)
9.2	<primitiveDatatype> ===> TK_REAL	<primitiveDatatype>.node = new leaf(TK_REAL)
10	<constructedDatatype> ===> TK_RECORD TK_RECORDID	<constructedDatatype>.node = new leaf(TK_RECORDID)
11.1	<remaining_list> ===> TK_COMMA <parameter_list>	<remaining_list>.list = <parameter_list>.list
11.2	<remaining_list> ===> eps	<remaining_list>.list = NULL

12	<stmts> ===> <typeDefinitions> <declarations> <otherStmts> <returnStmt>	<stmts>.list=insert at head(<typeDefinitions>.list,insert at head(<declarations>.list,insert at head(<otherStmts>.list, <returnStmt>.list)))
13.1	<typeDefinitions> ===> <typeDefinition> <typeDefinitions1>	<typeDefinitions>.list = insert at head(<typeDefinition>.node, <typeDefinitions1>.list)
13.2	<typeDefinitions> ===> eps	<typeDefinitions>.list = NULL
14	<typeDefinition> ===> TK_RECORD TK_RECORDID <fieldDefinitions> TK_ENDRECORD TK_SEM	<typeDefinition>.node= new node(14,TK_RECORDID, <fieldDefinitions>.list)
15	<fieldDefinitions> ===> <fieldDefinition> <fieldDefinition> <moreFields>	<fieldDefinitions>.list = insert at head(<fieldDefinition>.node,insert at head(<fieldDefinition>.node,<moreFields>.list))
16	<fieldDefinition> ===> TK_TYPE <primitiveDatatype> TK_COLON TK_FIELDID TK_SEM	<fieldDefinition>.node = new node(16, <primitiveDatatype>.node,TK_FIELDID)
17.1	<moreFields> ===> <fieldDefinition> <moreFields1>	<moreFields>.list = insert at head(<fieldDefinition>.node,<moreFields1>.list)
17.2	<moreFields> ===> eps	<moreFields>.list = NULL
18.1	<declarations> ===> <declaration> <declarations1>	<declarations>.list = insert at head(<declaration>.node,<declarations1>.list)
18.2	<declarations> ===> eps	<declarations>.list = NULL
19	<declaration> ===> TK_TYPE <dataType> TK_COLON TK_ID <global_or_not> TK_SEM	<declaration>.node = new node(19,<dataType>.node, TK_ID,<global_or_not>.node)
20.1	<global_or_not> ===> TK_COLON TK_GLOBAL	<global_or_not>.node = new leaf(TK_GLOBAL)
20.2	<global_or_not> ===> eps	<global_or_not>.node = new leaf(TK_LOCAL)
21.1	<otherStmts> ===> <stmt> <otherStmts1>	<otherStmts>.list = insert at head(<stmt>.node, <otherStmts1>.list)
21.2	<otherStmts> ===> eps	<otherStmts>.list = NULL
22.1	<stmt> ===> <assignmentStmt>	<stmt>.node = <assignmentStmt>.node
22.2	<stmt> ===> <iterativeStmt>	<stmt>.node = <iterativeStmt>.node
22.3	<stmt> ===> <conditionalStmt>	<stmt>.node = <conditionalStmt>.node
22.4	<stmt> ===> <ioStmt>	<stmt>.node = <ioStmt>.node
22.5	<stmt> ===> <funCallStmt>	<stmt>.node = <funCallStmt>.node

23	<assignmentStmt> ===> <singleOrRecId> TK_ASSIGNOP <arithmeticExpression> TK_SEM	<assignmentStmt>.node = new node(23, <singleOrRecId>.node,'=',<arithmeticExpression>.node)
24	<singleOrRecId> ===> TK_ID <new_24>	<singleOrRecId>.node = new node(24,TK_ID,<new_24>.node)
25.1	<new_24> ===> eps	<new_24>.node = NULL
25.2	<new_24> ===> TK_DOT TK_FIELDID	<new_24>.node = new leaf(TK_FIELDID)
26	<funCallStmt> ===> <outputParameters> TK_CALL TK_FUNID TK_WITH TK_PARAMETERS <inputParameters> TK_SEM	<funCallStmt>.node = new node(26,<outputParameters>.node,TK_FUNID,<inputParameters>.node)
27.1	<outputParameters> ===> TK_SQL <idList> TK_SQR TK_ASSIGNOP	<outputParameters>.node = new node(27.1,<idList>.list,'=')
27.2	<outputParameters> ===> eps	<outputParameters>.node = NULL
28	<inputParameters> ===> TK_SQL <idList> TK_SQR	<inputParameters>.node = new node(28,<idList>.list)
29	<iterativeStmt> ===> TK_WHILE TK_OP <booleanExpression> TK_CL <stmt> <otherStmts> TK_ENDWHILE	<iterativeStmt>.node = new node(29, <booleanExpression>.node, insert at head(<stmt>.node,<otherStmts>.list))
30	<conditionalStmt> ===> TK_IF TK_OP <booleanExpression> TK_CL TK_THEN <stmt> <otherStmts> <elsePart>	<conditionalStmt>.node = new node(30,<booleanExpression>.node,insert at head(<stmt>.node,<otherStmts>.list),<elsePart>.node)
31.1	<elsePart> ===> TK_ELSE <stmt> <otherStmts> TK_ENDIF	<elsePart>.node = new node(31.1, insert at head(<stmt>.node,<otherStmts>.list))
31.2	<elsePart> ===> TK_ENDIF	<elsePart>.node = NULL
32.1	<ioStmt> ===> TK_READ TK_OP <singleOrRecId> TK_CL TK_SEM	<ioStmt>.node = new node(32.1,<singleOrRecId>.node)
32.2	<ioStmt> ===> TK_WRITE TK_OP <all> TK_CL TK_SEM	<ioStmt>.node = new node(32.2,<all>.node)
33.1	<all> ===> TK_ID <new_24>	<all>.node = new node(33.1,TK_ID,<new_24>.node)
33.2	<all> ===> TK_NUM	<all>.node = new leaf(TK_NUM)
33.3	<all> ===> TK_RNUM	<all>.node = new leaf(TK_RNUM)
34	<arithmeticExpression> ===> <term> <expPrime>	<expPrime>.inh = <term>.node <arithmeticExpression>.node = <expPrime>.syn
35.1	<expPrime> ===> <lowPrecedenceOperators> <term> <expPrime1>	<expPrime1>.inh = new node(35.1, <lowPrecedenceOperators>.val,<expPrime>.inh,<term>.node) <expPrime>.syn=<expPrime1>.syn

35.2	<expPrime> ===> eps	<expPrime>.syn = <expPrime>.inh
36	<term> ===> <factor> <termPrime>	<termPrime>.inh = <factor>.node <term>.node=<termPrime>.syn
37.1	<termPrime> ===> <highPrecedenceOperators> <factor> <termPrime1>	<termPrime1>.inh = new node(37.1,<highPrecedenceOperators>.val,<termPrime>.inh,<factor>.node) <termPrime>.syn= <termPrime1>.syn
37.2	<termPrime> ===> eps	<termPrime>.syn = <termPrime>.inh
38.1	<factor> ===> TK_OP <arithmeticExpression> TK_CL	<factor>.node = <arithmeticExpression>.node
38.2	<factor> ===> <all>	<factor>.node = <all>.node
39.1	<highPrecedenceOperators> ===> TK_MUL	<highPrecedenceOperators>.node = new leaf(TK_MUL)
39.2	<highPrecedenceOperators> ===> TK_DIV	<highPrecedenceOperators>.node = new leaf(TK_DIV)
40.1	<lowPrecedenceOperators> ===> TK_PLUS	<lowPrecedenceOperators>.node = new leaf(TK_PLUS)
40.2	<lowPrecedenceOperators> ===> TK_MINUS	<lowPrecedenceOperators>.node = new leaf(TK_MINUS)
41.1	<booleanExpression> ===> TK_OP <booleanExpression1> TK_CL <logicalOp> TK_OP <booleanExpression2> TK_CL	<booleanExpression>.node = new node(41.1,<booleanExpression1>.node,<logicalOp>.node, <booleanExpression2>.node)
41.2	<booleanExpression> ===> <var1> <relationalOp> <var2>	<booleanExpression>.node = new node(41.2,<var1>.node,<relationalOp>.node,<var2>.node)
41.3	<booleanExpression> ===> TK_NOT TK_OP <booleanExpression1> TK_CL	<booleanExpression>.node = new node(41.3,<booleanExpression1>.node)
42.1	<var> ===> TK_ID	<var>.node = new leaf(TK_ID)
42.2	<var> ===> TK_NUM	<var>.node = new leaf(TK_NUM)
42.3	<var> ===> TK_RNUM	<var>.node = new leaf(TK_RNUM)
43.1	<logicalOp> ===> TK_AND	<logicalOp>.node = new leaf(TK_AND)
43.2	<logicalOp> ===> TK_OR	<logicalOp>.node = new leaf(TK_OR)
44.1	<relationalOp> ===> TK_LT	<relationalOp>.node = new leaf(TK_LT)
44.2	<relationalOp> ===> TK_LE	<relationalOp>.node = new leaf(TK_LE)

44.3	<relationalOp> ===> TK_EQ	<relationalOp>.node = new leaf(TK_EQ)
44.4	<relationalOp> ===> TK_GT	<relationalOp>.node = new leaf(TK_GT)
44.5	<relationalOp> ===> TK_GE	<relationalOp>.node = new leaf(TK_GE)
44.6	<relationalOp> ===> TK_NE	<relationalOp>.node = new leaf(TK_NE)
45	<returnStmt> ===> TK_RETURN <optionalReturn> TK_SEM	<returnStmt>.node = new node (45,<optionalReturn>.list)
46.1	<optionalReturn> ===> TK_SQL <idList> TK_SQR	<optionalReturn>.list = <idList>.list
46.2	<optionalReturn> ===> eps	<optionalReturn>.list = NULL
47	<idList> ===> TK_ID <more_ids>	<idList>.list = insert at head(TK_ID,<more_ids>.list)
48.1	<more_ids> ===> TK_COMMA <idList>	<more_ids>.list = <idList>.list
48.2	<more_ids> ===> eps	<more_ids>.list = NULL