Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

Semantic Rules For AST Creation

1. *<program> ===> <otherFunctions> <mainFunction>*
   <program>.node = makeNode("program", <otherFunctions>.node, <mainFunction>.node)

2. *<mainFunction>===> TK_MAIN <stmts> TK_END*
   <mainFunction>.node = makeNode(“main”, <stmts>.node)

3. *<otherFunctions>===> <function><otherFunctions>$_1$*
   <otherFunctions>.node = makeNode("func_Seq", <function>.node, <otherFunctions>$_1$.node)

4. *<otherFunctions>===>eps*
   <otherFunctions>.node = NULL

5. *<function>===>TK_FUNID <input_par> <output_par> TK_SEM <stmts> TK_END*
   <function>.node = makeNode(TK_FUNID.name, <input_par>.node, <output_par>.node, <stmts>.node)

6. *<input_par>===>TK_INPUT TK_PARAMETER TK_LIST TK_SQL <parameter_list> TK_SQR*
   <input_par>.node = <parameter_list>.node

7. *<output_par>===>TK_OUTPUT TK_PARAMETER TK_LIST TK_SQL <parameter_list> TK_SQR*
   <output_par>.node = <parameter_list>.node

8. *<output_par>===>eps*
   <output_par>.node = NULL

9. *<parameter_list>===><dataType> TK_ID <remaining_list>*
   <parameter_list>.node = makeNode(“par_list”, <dataType>.type, TK_ID.token, <remainingList>.node)

10. *<dataType>===> <primitiveDatatype>*
    <dataType>.type = <primitiveDatatype>.type

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

11. *<dataType>===> <constructedDatatype>*
    <dataType>.type = <constructedDatatype>.type

12. *<primitiveDatatype>===> TK_INT*
    <primitiveDatatype>.type = “INT”

13. *<primitiveDatatype>===> TK_REAL*
    <primitiveDatatype>.type = “REAL”

14. *<constructedDatatype>===>TK_RECORD TK_RECORDID*
    <constructedDatatype>.type = TK_RECORDID.type

15. *<remaining_list>===>TK_COMMA <parameter_list>*
    <remaining_list>.node = <parameter_list>.node

16. *<remaining_list>===>eps*
    <remaining_list>.node = NULL

17. *<stmts>===><typeDefinitions> <declarations> <otherStmts><returnStmt>*
    <stmts>.node = makeNode(“stmts”, <typeDefinitions>.node, <declarations>.node, <otherStmts>.node, <returnStmt>.node)

18. *<typeDefinitions>===><typeDefinition><typeDefinitions>$_1$*
    <typeDefinitions>.node = makeNode(“type_defs”, <typeDefinition>.node, <typeDefinitions>$_1$.node)

19. *<typeDefinitions>===>eps*
    <typeDefinitions>.node = NULL

20. *<typeDefinition>===>TK_RECORD TK_RECORDID <fieldDefinitions> TK_ENDRECORD TK_SEM*
    <typeDefinition>.node = makeNode(“type_def”, TK_RECORDID.token, <fieldDefinitions>.node)

21. *<fieldDefinitions>===> <fieldDefinition>$_1$<fieldDefinition>$_2$<moreFields>*
    <fieldDefintions>.node = makeNode(“field_defs”, <fieldDefinition>$_1$.node, <fieldDefinition>$_2$.node, <moreFields>.node)

22. *<fieldDefinition>===> TK_TYPE <primitiveDatatype> TK_COLON TK_FIELDID TK_SEM*
    <fieldDefinition>.node = makeNode(“field_def”, <primitiveDatatype>.type, TK_FIELDID.token)

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

23. *<moreFields>===><fieldDefinition><moreFields>$_1$*
    <moreFields>.node = makeNode(“more_fields”, <fieldDefinition>.node, <moreFields>$_1$.node)

24. *<moreFields>===>eps*
    <moreFields>.node = NULL

25. *<declarations> ===> <declaration><declarations>$_1$*
    <declarations>.node = makeNode(“declarations”, <declaration>.node, <declarations>$_1$.node)

26. *<declarations> ===> eps*
    <declarations>.node = NULL

27. *<declaration> ===> TK_TYPE <dataType> TK_COLON TK_ID <global_or_not> TK_SEM*
    <declaration>.node = makeNode(“declaration”, <dataType>.type, TK_ID.token, <global_or_not>.is_global)

28. *<global_or_not> ===> TK_COLON TK_GLOBAL*
    <global_or_not>.is_global = TRUE

29. *<global_or_not>===>eps*
    <global_or_not>.is_global = FALSE

30. *<otherStmts>===> <stmt><otherStmts>$_1$*
    <otherStmts>.node = makeNode(“other_stmts”, <stmt>.node, <otherStmts>$_1$.node)

31. *<otherStmts>===>eps*
    <otherStmts>.node = NULL

32. *<stmt>===> <assignmentStmt>*
    <stmt>.node = <assignmentStmt>.node

33. *<stmt>===> <iterativeStmt>*
    <stmt>.node = <iterativeStmt>.node

34. *<stmt>===> <conditionalStmt>*
    <stmt>.node = <conditionalStmt>.node

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

35. *<stmt>===> <ioStmt>*
    <stmt>.node = <ioStmt>.node

36. *<stmt>===> <funCallStmt>*
    <stmt>.node = <funCallStmt>.node

37. *<assignmentStmt>===><SingleOrRecId> TK_ASSIGNOP <arithmeticExpression> TK_SEM*
    <assignmentStmt>.node = makeNode("assign_stmt", <SingleOrRecId>.node, <arithmeticExpression>.node)

38. *<SingleOrRecId>===>TK_ID <new_24>*
    <SingleOrRecId>.node = makeNode("single_or_rec_id", TK_ID.token, <new_24>.node)

39. *<new_24> ===> eps*
    <new_24>.node = NULL

40. *<new_24> ===> TK_DOT TK_FIELDID*
    <new_24>.node = makeNode("new_24", TK_FIELDID.token)

41. *<funCallStmt>===><outputParameters> TK_CALL TK_FUNID TK_WITH TK_PARAMETERS <inputParameters> TK_SEM*
    <funCallStmt>.node = makeNode("funCall_stmt", <outputParameters>.node, TK_FUNID.token, <inputParameters>.node)

42. *<outputParameters> ==> TK_SQL <idList> TK_SQR TK_ASSIGNOP*
    <outputParameters>.node = makeNode("output_pars", <idList>.node)

43. *<outputParameters> ==> eps*
    <outputParameters>.node = NULL

44. *<inputParameters>===> TK_SQL <idList> TK_SQR*
    <inputParameters>.node = makeNode("input_pars", <idList>.node)

45. *<iterativeStmt>===> TK_WHILE TK_OP <booleanExpression> TK_CL <stmt><otherStmts> TK_ENDWHILE*
    <iterativeStmt>.node = makeNode("iterative_stmt", <booleanExpression>.node, <stmt>.node, <otherStmts>.node)

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

46. *<conditionalStmt> ===> TK_IF TK_OP <booleanExpression> TK_CL TK_THEN <stmt> <otherStmts>< elsePart>*
    <conditionalStmt>.node = makeNode(“condition_stmt”, <booleanExpression>.node, <stmt>.node, <otherStmts>.node, <elsePart>.node)

47. *<elsePart> ===> TK_ELSE <stmt> <otherStmts> TK_ENDIF*
    <elsePart>.node = makeNode(“else_part”, <stmt>.node, <otherStmts>.node)

48. *<elsePart> ===> TK_ENDIF*
    <elsePart>.node = NULL

49. *<ioStmt>===>TK_READ TK_OP <SingleOrRecId> TK_CL TK_SEM*
    <ioStmt>.node = makeNode(“io_stmt_read”, <SingleOrRecId>.node)

50. *<ioStmt>===>TK_WRITE TK_OP <allVar> TK_CL TK_SEM*
    <ioStmt>.node = makeNode(“io_stmt_write”, <allVar>.node)

51. *<allVar> ===> TK_ID <var_mid>*
    <allVar>.node = makeNode(“all_var_id”, TK_ID.token, <var_mid>.node)

52. *<var_mid> ===> TK_DOT TK_FIELDID*
    <var_mid>.node = makeNode(“var_mid”, TK_FIELDID.token)

53. *<var_mid> ===> eps*
    <var_mid>.node = NULL

54. *<allVar> ===> TK_NUM*
    <allVar>.node = makeNode(“all_var_num”, TK_NUM.token)

55. *<allVar> ===> TK_RNUM*
    <allVar>.node = makeNode(“all_var_rnum”, TK_RNUM.token)

56. *<arithmeticExpression>===><term> <expression>*
    <expression>.inh = <term>.node
    <arithmeticExpression>.node = <expression>.node

57. *<expression>===> <operator1> <term> <expression>$_1$*
    <expression>$_1$.inh = makeNode(<operator1>.name, <expression>.inh, <term>.node)
    <expression>.node = <expression>$_1$.node

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

58. <expression>===>eps
<expression>.node = <expression>.inh

59. <term>===> <factor><term'>
<term'>.inh = <factor>.node
<term>.node = <term'>.node

60. <term'>===> <operator2> <factor> <term'>$_1$
<term'>$_1$.inh = makeNode(<operator2>.name, <term'>.inh, <factor>.node)
<term'>.node = <term'>$_1$.node

61. <term'>===> eps
<term'>.node = <term'>.inh

62. <factor>===> TK_OP <arithmeticExpression> TK_CL
<factor>.node = <arithmeticExpression>.node

63. <factor>===> <all>
<factor>.node = <all>.node

64. <all> ===> TK_NUM
<all>.node = makeNode("all_num", TK_NUM.token)

65. <all> ===> TK_RNUM
<all>.node = makeNode("all_rnum", TK_RNUM.token)

66. <all> ===> TK_ID <temp>
<all>.node = makeNode("all_id", TK_ID.token, <temp>.node)

67. <temp> ===> eps
<temp>.node = NULL

68. <temp> ===> TK_DOT TK_FIELDID
<temp>.node = makeNode("temp", TK_FIELDID.token)

69. <operator1> ===> TK_PLUS
<operator1>.name = "PLUS"

70. <operator1> ===> TK_MINUS
<operator1>.name = "MINUS"

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

71. *<operator2> ===> TK_MUL*
    <operator2>.name = “MUL”

72. *<operator2> ===> TK_DIV*
    <operator2>.name = “DIV”

73. *<booleanExpression>===>TK_OP <booleanExpression>$_1$ TK_CL <logicalOp> TK_OP <booleanExpression>$_2$ TK_CL*
    <booleanExpression>.node = makeNode(<logicalOp>.name, <booleanExpression>$_1$.node, <booleanExpression>$_2$.node)

74. *<booleanExpression>===> <var>$_1$ <relationalOp> <var>$_2$*
    <booleanExpression>.node = makeNode(<relationalOp>.name, <var>$_1$.node, <var>$_2$.node)

75. *<booleanExpression>===> TK_NOT TK_OP <booleanExpression>$_1$ TK_CL*
    <booleanExpression>.node = makeNode(“NOT”, <booleanExpression>$_1$.node)

76. *<var>===> TK_ID*
    <var>.node = makeNode(“var_id”, TK_ID.token)

77. *<var>===> TK_NUM*
    <var>.node = makeNode(“var_num”, TK_NUM.token)

78. *<var>===> TK_RNUM*
    <var>.node = makeNode(“var_rnum”, TK_RNUM.token)

79. *<logicalOp>===>TK_AND*
    <logicalOp>.name = “AND”

80. *<logicalOp>===>TK_OR*
    <logicalOp>.name = “OR”

81. *<relationalOp>===> TK_LT*
    <relationalOp>.name = “LT”

82. *<relationalOp>===> TK_LE*
    <relationalOp>.name = “LE”

83. *<relationalOp>===> TK_EQ*
    <relationalOp>.name = “EQ”

Group 33:
2016A7PS0036P Megh Thakkar
2016A7PS0103P Sahil Singla
2016A7PS0110P Sankalp Sangle
2016A7PS0150P Patel Parth

84. <relationalOp>===> TK_GT
<relationalOp>.name = “GT”

85. <relationalOp>===> TK_GE
<relationalOp>.name = “GE”

86. <relationalOp>===> TK_NE
<relationalOp>.name = “NE”

87. <returnStmt>===>TK_RETURN <optionalReturn> TK_SEM
<returnStmt>.node = makeNode(“return_stmt”, <optionalReturn>.node)

88. <optionalReturn>===>TK_SQL <idList> TK_SQR
<optionalReturn>.node = makeNode(“return_pars”, <idList>.node)

89. <optionalReturn>===>eps
<optionalReturn>.node = NULL

90. <idList>===> TK_ID <more_ids>
<idList>.node = makeNode(“id_list”, TK_ID.token, <more_ids>.node)

91. <more_ids>===> TK_COMMA <idList>
<more_ids>.node = <idList>.node

92. <more_ids>===> eps
<more_ids>.node = NULL

Note1: Unless specified as inherited, all attributes are by default synthesized.

Note2: makeNode()’s arguments: First argument specifies the label to be assigned to the node, whereas rest of the arguments specify the children of the node.