Group 26

Corrected LL (1) Grammar

1. <program> ===> <otherFunctions> <mainFunction>
2. <mainFunction> ===> **TK_MAIN** <stmts> **TK_END**
3. <otherFunctions> ===> <function> <otherFunctions> | ∈
4. <function> ===> **TK_FUNID** <input_par> <output_par> **TK_SEM** <stmts> **TK_END**
5. <input_par> ===> **TK_INPUT TK_PARAMETER TK_LIST TK_SQL** <parameter_list> **TK_SQR**
6. <output_par> ===> **TK_OUTPUT TK_PARAMETER TK_LIST TK_SQL** <parameter_list> **TK_SQR** | ∈
7. <parameter_list> ===> <dataType> **TK_ID** <remaining_list>
8. <dataType> ===> <primitiveDatatype> | <constructedDatatype>
9. <primitiveDatatype> ===> **TK_INT** | **TK_REAL**
10. <constructedDatatype> ===> **TK_RECORD TK_RUID** | **TK_UNION TK_RUID** | **TK_RUID**
11. <remaining_list> ===> **TK_COMMA** <parameter_list> | ∈
12. <stmts> ===> <typeDefinitions> <declarations> <otherStmts> <returnStmt>
13. <typeDefinitions> ===> <definitionOrTypedef> <typeDefinitions> | ∈
14. <definitionOrTypedef> ===> <definetypestmt> | <typeDefinition>
15. <typeDefinition> ===> **TK_RECORD TK_RUID** <fieldDefinitions> **TK_ENDRECORD**
16. <typeDefinition> ===> **TK_UNION TK_RUID** <fieldDefinitions> **TK_ENDUNION**
17. <fieldDefinitions> ===> <fieldDefinition> <fieldDefinition> <moreFields>
18. <fieldDefinition> ===> **TK_TYPE** <dataType> **TK_COLON TK_FIELDID TK_SEM**
19. <moreFields> ===> <fieldDefinition> <moreFields> | ∈
20. <declarations> ===> <declaration> <declarations> | ∈
21. <declaration> ===> **TK_TYPE** <dataType> **TK_COLON TK_ID** <global_or_not> **TK_SEM**
22. <global_or_not> ===> **TK_COLON TK_GLOBAL** | ∈
23. <otherStmts> ===> <stmt> <otherStmts> | ∈
24. <stmt> ===> <assignmentStmt> | <iterativeStmt> | <conditionalStmt> | <ioStmt> | <funCallStmt>
25. <assignmentStmt> ===> <singleOrRecId> **TK_ASSIGNOP** <arithmeticExpression> **TK_SEM**
26. <singleOrRecId> ===> **TK_ID** <ifRecField>
27. <ifRecField> ===> **TK_DOT TK_FIELDID** <ifRecField> | ∈
28. <funCallStmt> ===> <outputParameters> **TK_CALL TK_FUNID TK_WITH TK_PARAMETERS** <inputParameters> **TK_SEM**
29. <outputParameters> ===> **TK_SQL** <idList> **TK_SQR TK_ASSIGNOP** | ∈
30. <inputParameters> ===> **TK_SQL** <idList> **TK_SQR**
31. <iterativeStmt> ===> **TK_WHILE TK_OP** <booleanExpression> **TK_CL** <stmt> <otherStmts> **TK_ENDWHILE**
32. <conditionalStmt> ===> **TK_IF TK_OP** <booleanExpression> **TK_CL TK_THEN** <stmt> <otherStmts> <handle_else>
33. <handle_else> ===> **TK_ENDIF** | **TK_ELSE** <stmt> <otherStmts> **TK_ENDIF**
34. <ioStmt> ===> **TK_READ TK_OP** <singleOrRecId> **TK_CL TK_SEM** | **TK_WRITE TK_OP** <singleOrRecId> **TK_CL TK_SEM**
35. <arithmeticExpression> ===> <term> <expression'>
36. <expression'> ===> <sum_operators> <term> <expression'> | ∈
37. <term> ===> <factor> <term'>
38. <term'> ===> <mult_operators> <factor> <term'> | ∈
39. <factor> ===> **TK_OP** <arithmeticExpression> **TK_CL** | <var>
40. <sum_operators> ===> **TK_PLUS** | **TK_MINUS**

41. <mult_operators> ===> **TK_MUL** | **TK_DIV**
42. <booleanExpression> ===> **TK_OP** <booleanExpression> **TK_CL** <logicalOp> **TK_OP** <booleanExpression> **TK_CL**
43. <booleanExpression> ===> <var> <relationalOp> <var>
44. <booleanExpression> ===> **TK_NOT TK_OP** <booleanExpression> **TK_CL**
45. <var> ===> <singleOrRecId> | **TK_NUM** | **TK_RNUM**
46. <logicalOp> ===> **TK_AND** | **TK_OR**
47. <relationalOp> ===> **TK_LT** | **TK_LE** | **TK_EQ** | **TK_GT** | **TK_GE** | **TK_NE**
48. <returnStmt> ===> **TK_RETURN** <optionalReturn> **TK_SEM**
49. <optionalReturn> ===> **TK_SQL** <idList> **TK_SQR** | ∈
50. <idList> ===> **TK_ID** <more_ids>
51. <more_ids> ===> **TK_COMMA** <idList> | ∈
52. <definetypestmt> ===> **TK_DEFINETYPE** <A> **TK_RUID TK_AS TK_RUID**
53. <A> ===> **TK_RECORD** | **TK_UNION**

FIRST SETS

1. **FIRST(** <program> **)** = { TK_MAIN, TK_FUNID }
2. **FIRST(** <mainFunction> **)** = { TK_MAIN }
3. **FIRST(** <otherFunctions> **)** = { TK_FUNID, ∈ }
4. **FIRST(** <function> **)** = { TK_FUNID }
5. **FIRST(** <input_par> **)** = { TK_INPUT }
6. **FIRST(** <output_par> **)** = { TK_OUTPUT, ∈ }
7. **FIRST(** <parameter_list> **)** = { TK_RUID, TK_RECORD, TK_INT, TK_UNION, TK_REAL }
8. **FIRST(** <dataType> **)** = { TK_RUID, TK_RECORD, TK_UNION, TK_INT, TK_REAL }
9. **FIRST(** <primitiveDatatype> **)** = { TK_INT, TK_REAL }
10. **FIRST(** <constructedDatatype> **)** = { TK_RECORD, TK_RUID, TK_UNION }
11. **FIRST(** <remaining_list> **)** = { ∈, TK_COMMA }
12. **FIRST(** <stmts> **)** = { TK_RETURN, TK_WRITE, TK_ID, TK_CALL, TK_SQL, TK_RECORD, TK_READ, TK_IF, TK_UNION, TK_WHILE, TK_DEFINETYPE, TK_TYPE }
13. **FIRST(** <typeDefinitions> **)** = { TK_RECORD, TK_DEFINETYPE, TK_UNION, ∈ }
14. **FIRST(** <definitionOrTypedef> **)** = { TK_UNION, TK_RECORD, TK_DEFINETYPE }
15. **FIRST(** <typeDefinition> **)** = { TK_UNION, TK_RECORD }
16. **FIRST(** <fieldDefinitions> **)** = { TK_TYPE }
17. **FIRST(** <fieldDefinition> **)** = { TK_TYPE }
18. **FIRST(** <moreFields> **)** = { TK_TYPE, ∈ }
19. **FIRST(** <declarations> **)** = { ∈, TK_TYPE }
20. **FIRST(** <declaration> **)** = { TK_TYPE }
21. **FIRST(** <global_or_not> **)** = { ∈, TK_COLON }
22. **FIRST(** <otherStmts> **)** = { TK_WRITE, TK_ID, TK_SQL, TK_CALL, ∈, TK_IF, TK_WHILE, TK_READ }
23. **FIRST(** <stmt> **)** = { TK_IF, TK_CALL, TK_SQL, TK_READ, TK_WHILE, TK_WRITE, TK_ID }
24. **FIRST(** <assignmentStmt> **)** = { TK_ID }
25. **FIRST(** <singleOrRecId> **)** = { TK_ID }

26. **FIRST(** <ifRecField> **)** = { TK_DOT, ∈ }
27. **FIRST(** <funCallStmt> **)** = { TK_CALL, TK_SQL }
28. **FIRST(** <outputParameters> **)** = { TK_SQL, ∈ }
29. **FIRST(** <inputParameters> **)** = { TK_SQL }
30. **FIRST(** <iterativeStmt> **)** = { TK_WHILE }
31. **FIRST(** <conditionalStmt> **)** = { TK_IF }
32. **FIRST(** <handle_else> **)** = { TK_ELSE, TK_ENDIF }
33. **FIRST(** <ioStmt> **)** = { TK_READ, TK_WRITE }
34. **FIRST(** <arithmeticExpression> **)** = { TK_OP, TK_RNUM, TK_ID, TK_NUM }
35. **FIRST(** <expression'> **)** = { TK_MINUS, ∈, TK_PLUS }
36. **FIRST(** <term> **)** = { TK_OP, TK_RNUM, TK_NUM, TK_ID }
37. **FIRST(** <term'> **)** = { ∈, TK_MUL, TK_DIV }
38. **FIRST(** <factor> **)** = { TK_RNUM, TK_OP, TK_ID, TK_NUM }
39. **FIRST(** <sum_operators> **)** = { TK_PLUS, TK_MINUS }
40. **FIRST(** <mult_operators> **)** = { TK_MUL, TK_DIV }
41. **FIRST(** <booleanExpression> **)** = { TK_NUM, TK_OP, TK_RNUM, TK_NOT, TK_ID }
42. **FIRST(** <var> **)** = { TK_ID, TK_RNUM, TK_NUM }
43. **FIRST(** <logicalOp> **)** = { TK_AND, TK_OR }
44. **FIRST(** <relationalOp> **)** = { TK_EQ, TK_LT, TK_NE, TK_GE, TK_GT, TK_LE }
45. **FIRST(** <returnStmt> **)** = { TK_RETURN }
46. **FIRST(** <optionalReturn> **)** = { TK_SQL, ∈ }
47. **FIRST(** <idList> **)** = { TK_ID }
48. **FIRST(** <more_ids> **)** = { TK_COMMA, ∈ }
49. **FIRST(** <definetypestmt> **)** = { TK_DEFINETYPE }
50. **FIRST(** <A> **)** = { TK_RECORD, TK_UNION }

FOLLOW SETS

1. **FOLLOW(** <program> **)** = { $ }
2. **FOLLOW(** <mainFunction> **)** = { $ }
3. **FOLLOW(** <otherFunctions> **)** = { TK_MAIN }
4. **FOLLOW(** <function> **)** = { TK_FUNID, TK_MAIN }
5. **FOLLOW(** <input_par> **)** = { TK_OUTPUT, TK_SEM }
6. **FOLLOW(** <output_par> **)** = { TK_SEM }
7. **FOLLOW(** <parameter_list> **)** = { TK_SQR }
8. **FOLLOW(** <dataType> **)** = { TK_COLON, TK_ID }
9. **FOLLOW(** <primitiveDatatype> **)** = { TK_ID, TK_COLON }
10. **FOLLOW(** <constructedDatatype> **)** = { TK_ID, TK_COLON }
11. **FOLLOW(** <remaining_list> **)** = { TK_SQR }
12. **FOLLOW(** <stmts> **)** = { TK_END }
13. **FOLLOW(** <typeDefinitions> **)** = { TK_WRITE, TK_ID, TK_WHILE, TK_CALL, TK_RETURN, TK_SQL, TK_IF, TK_READ, TK_TYPE }

14. **FOLLOW(** <definitionOrTypedef> **)** = { TK_SQL, TK_UNION, TK_ID, TK_TYPE, TK_WRITE, TK_RECORD, TK_WHILE, TK_CALL, TK_IF, TK_READ, TK_DEFINETYPE, TK_RETURN }
15. **FOLLOW(** <typeDefinition> **)** = { TK_RECORD, TK_READ, TK_SQL, TK_TYPE, TK_DEFINETYPE, TK_UNION, TK_WHILE, TK_IF, TK_RETURN, TK_ID, TK_CALL, TK_WRITE }
16. **FOLLOW(** <fieldDefinitions> **)** = { TK_ENDUNION, TK_ENDRECORD }
17. **FOLLOW(** <fieldDefinition> **)** = { TK_TYPE, TK_ENDRECORD, TK_ENDUNION }
18. **FOLLOW(** <moreFields> **)** = { TK_ENDRECORD, TK_ENDUNION }
19. **FOLLOW(** <declarations> **)** = { TK_RETURN, TK_READ, TK_IF, TK_ID, TK_WHILE, TK_CALL, TK_SQL, TK_WRITE }
20. **FOLLOW(** <declaration> **)** = { TK_RETURN, TK_SQL, TK_TYPE, TK_WRITE, TK_ID, TK_IF, TK_WHILE, TK_READ, TK_CALL }
21. **FOLLOW(** <global_or_not> **)** = { TK_SEM }
22. **FOLLOW(** <otherStmts> **)** = { TK_ENDIF, TK_ELSE, TK_RETURN, TK_ENDWHILE }
23. **FOLLOW(** <stmt> **)** = { TK_CALL, TK_WRITE, TK_ELSE, TK_RETURN, TK_WHILE, TK_ENDWHILE, TK_ID, TK_ENDIF, TK_IF, TK_SQL, TK_READ }
24. **FOLLOW(** <assignmentStmt> **)** = { TK_IF, TK_ID, TK_ENDIF, TK_CALL, TK_READ, TK_SQL, TK_WHILE, TK_ENDWHILE, TK_ELSE, TK_RETURN, TK_WRITE }
25. **FOLLOW(** <singleOrRecId> **)** = { TK_EQ, TK_ASSIGNOP, TK_GE, TK_LE, TK_LT, TK_NE, TK_MINUS, TK_SEM, TK_PLUS, TK_DIV, TK_CL, TK_GT, TK_MUL }
26. **FOLLOW(** <ifRecField> **)** = { TK_LE, TK_MINUS, TK_NE, TK_CL, TK_GT, TK_GE, TK_PLUS, TK_LT, TK_ASSIGNOP, TK_SEM, TK_DIV, TK_EQ, TK_MUL }
27. **FOLLOW(** <funCallStmt> **)** = { TK_READ, TK_ID, TK_WHILE, TK_SQL, TK_ENDWHILE, TK_CALL, TK_WRITE, TK_ELSE, TK_IF, TK_ENDIF, TK_RETURN }
28. **FOLLOW(** <outputParameters> **)** = { TK_CALL }
29. **FOLLOW(** <inputParameters> **)** = { TK_SEM }
30. **FOLLOW(** <iterativeStmt> **)** = { TK_ENDIF, TK_ENDWHILE, TK_ELSE, TK_READ, TK_WRITE, TK_CALL, TK_IF, TK_WHILE, TK_RETURN, TK_SQL, TK_ID }
31. **FOLLOW(** <conditionalStmt> **)** = { TK_ENDWHILE, TK_CALL, TK_READ, TK_RETURN, TK_WHILE, TK_ENDIF, TK_SQL, TK_ID, TK_WRITE, TK_IF, TK_ELSE }
32. **FOLLOW(** <handle_else> **)** = { TK_IF, TK_CALL, TK_WRITE, TK_RETURN, TK_ID, TK_ENDWHILE, TK_ENDIF, TK_ELSE, TK_READ, TK_WHILE, TK_SQL }
33. **FOLLOW(** <ioStmt> **)** = { TK_ELSE, TK_SQL, TK_ENDIF, TK_WRITE, TK_IF, TK_RETURN, TK_READ, TK_CALL, TK_WHILE, TK_ENDWHILE, TK_ID }
34. **FOLLOW(** <arithmeticExpression> **)** = { TK_SEM, TK_CL }
35. **FOLLOW(** <expression'> **)** = { TK_SEM, TK_CL }
36. **FOLLOW(** <term> **)** = { TK_CL, TK_SEM, TK_MINUS, TK_PLUS }
37. **FOLLOW(** <term'> **)** = { TK_CL, TK_MINUS, TK_SEM, TK_PLUS }
38. **FOLLOW(** <factor> **)** = { TK_DIV, TK_CL, TK_PLUS, TK_MUL, TK_MINUS, TK_SEM }
39. **FOLLOW(** <sum_operators> **)** = { TK_NUM, TK_OP, TK_RNUM, TK_ID }
40. **FOLLOW(** <mult_operators> **)** = { TK_NUM, TK_OP, TK_ID, TK_RNUM }
41. **FOLLOW(** <booleanExpression> **)** = { TK_CL }
42. **FOLLOW(** <var> **)** = {TK_EQ, TK_LE, TK_PLUS, TK_LT, TK_NE, TK_MUL, TK_SEM, TK_DIV, TK_MINUS, TK_CL, TK_GE, TK_GT }
43. **FOLLOW(** <logicalOp> **)** = { TK_OP }
44. **FOLLOW(** <relationalOp> **)** = { TK_NUM, TK_ID, TK_RNUM }

45. **FOLLOW(** <returnStmt> **)** = { TK_END }
46. **FOLLOW(** <optionalReturn> **)** = { TK_SEM }
47. **FOLLOW(** <idList> **)** = { TK_SQR }
48. **FOLLOW(** <more_ids> **)** = { TK_SQR }
49. **FOLLOW(** <definetypestmt> **)** = { TK_CALL, TK_SQL, TK_READ, TK_RECORD, TK_RETURN, TK_WRITE, TK_UNION, TK_DEFINETYPE, TK_ID, TK_TYPE, TK_IF, TK_WHILE }
50. **FOLLOW(** <A> **)** = { TK_RUID }

STUDENTS:

- RISHI GUPTA: 2021A7PS0690P
- UTKARSH SHARMA: 2021A7PS0693P
- SAUMYA SHARMA: 2021A7PS0544P
- AKSHAT BAJPAI: 2021A7PS0573P
- SPARSH GOENKA: 2021A7PS2413P