Assumptions

→ In the order of statements, <definetypestmts> are placed after <typeDefinitions>

→ Records and Unions can now have records/unions as their fields (nesting)

→ Added semicolons to funCallStmt and defineTypeStmt for uniformity

→ Binded colon with global-or-not

→ Read function must be called only with TK_ID as argument and not TK_NUM or TK_RNUM constants.

→ Records/Unions fields can be also now be written with write

→ <stmt><otherStmts> was replaced with <otherStmts>

→ Added TK_OP and TK_CL in conditional statements.

→ Removed ambiguity from arithmetic Expression

→ Incorporated precedence order in arithmetic Expression

→ Introduced firstOp and secondOp instead of using operator to facilitate precedence

→ Made negation of booleanExpression more intuitive by adding parenthesis around the negated booleanExpression.

Group 5	Jatin Aggarwal - 2018B4A70884P Kush Mehta - 2018B5A70956P Harsh Thurjhaiwala - 2018B5A70691P Patel Darsh Rajesh - 2018B4A70532P Purushoroth Anirudh - 2018B5A70897P

1.) <program> ==> <otherFunctions> <mainFunction>

2.) <mainFunction> ==> TK_MAIN <stmts> TK_END.

3.) <otherFunctions> ==> <function> <otherFunctions> | ϵ

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 | ϵ

7.) <parameter_list> ==> <dataType> TK_ID <remaining_list>

8.) <dataType> ==> <primitiveDatatype> | <constructed Datatype>

9.) <primitiveDatatype> ==> TK_INT | TK_REAL

10.) <constructedDatatype> ==> TK_RECORD TK_RUID | TK_UNION TK_RUID

11.) <remaining_list> ==> TK_comma <parameter list> | ϵ

12.) <stmts> ==> <typeDefinitions> <declarations> <otherStmts> <returnStmt>

13.) <typeDefinitions> ==> <typeDefinition> <typeDefinitions> | ϵ

14.) <typeDefinition> ==> TK_UNION TK_RUID <fieldDefinitions> TK_ENDRECORD

15.) <typeDefinition> ==> TK_UNION TK_RUID <fieldDefinitions> <moreFields>

16.) <fieldDefinitions> ==> <fieldDefinition> <fieldDefinition> <moreFields>

17.) <fieldDefinition> ==> TK_TYPE <dataType> TK_COLON TK_FIELDID TK_SEM

18.) <moreFields> ==> <fieldDefinition> <moreFields> | ϵ

19.) <declarations> ==> <declaration> <declarations> | ϵ

20.) <declaration> ==> TK_TYPE <dataType> TK_COLON TK_ID <global_or_not> TK_SEM.

21.) <global_or_not> ==> TK_COLON TK_GLOBAL | ϵ

22.) <otherStmts> ==> <stmt> <otherStmts> | ϵ

23.) <stmt> ==> <assignmentStmt> | <iterative Stmt> | <conditional Stmt> | <io stmt> | <fun call Stmt>

24) <assignmentStmt> ==> <SingleOrRecId> TK_ASSIGNOP <arithmeticExpression> TK_SEM

25) <singleOrRecId> ==> TK_ID <singleOrRecPrime>

26) <singleOrRecOrNum> ==> <singleOrRecId> | TK_NUM | TK_RNUM

27) <singleOrRecPrime> ==> TK_DOT TK_FIELDID <singleOrRecPrime> | ϵ

28) <funCallStmt> ==> <outputParameters> TK_CALL TK_funid TK_with
TK_PARAMETERS <inputParameters> TK_SEM

29) <outputParameters> ==> TK_SQL <idList> TK_SQR TK_ASSIGNOP | ϵ

30) <inputParameters> ==> TK_SQL <idList> TK_SQR.

31) <iterativeStmt> ==> TK_WHILE TK_OP <booleanExpression> TK_CL <otherStmts>
TK_ENDWHILE

32) <conditionalStmt> ==> TK_IF TK_OP <booleanExpression> TK_CL TK_THEN <otherStmts>
<conditionalPrime>

33) <conditionalPrime> ==> TK_ELSE <otherStmts> TK_ENDIF | TK_ENDIF

34) <ioStmt> ==> TK_READ TK_OP TK_ID TK_CL TK_SEM | TK_WRITE TK_OP <singleOrRecOrNum>
TK_CL TK_SEM

35) <arithmeticExpression> ==> ~~<arithmeticPrime> <operator> <arithmeticExpression> | TK_OP~~
~~<arithmeticExpression> TK_CL | <var>~~ <mulDivPrime> <addOrSub>

36) <arithmeticPrime> ==> <var>

37) <operator> ==> TK_PLUS | TK_MUL | TK_MINUS | TK_DIV

38) <booleanExpression> ==> TK_OP <booleanExpression> TK_CL <booleanExtension> | TK_NOT
TK_OP <booleanExpression> TK_CL <booleanExpression> | <booleanPrime>

39) <booleanExtension> ==> <logicalOp> TK_OP <booleanExpression> TK_CL | ϵ

40) <booleanPrime> ==> <var> <relationalOp> <var>

41) <var> ==> TK_ID | TK_NUM | TK_RNUM

42) <logicalOp> ==> TK_AND | TK_OR

42) <relationalOp> ==> TK_LT | TK_LE | TK_EQ | TK_GT | TK_GE | TK_NE

44) <returnStmt> ==> TK_RETURN <optionalReturn> TK_SEM

45) <optionalReturn> ==> TK_SQL <idList> TK_SQR | ϵ

46) <idlist> ==> TK_ID <more_ids>

47) <more ids> ==> TK_comma <idlist> | ε

48) <definetype stmts> ==> <definetype stmt> <definetype stmts> | ε

49) <definetype stmt> ==> TK_DEFINETYPE <A> TK_RUID TK_AS TK_RUID TK_SEM

50) <A> ==> TK_RECORD | TK_UNION

51) <addOrSub> ==> <secondOp> <arithmetic Expression> | ε

52) <mulDivArith> ==> <parenArith> <mulOrDiv>

53) <mulOrDiv> ==> <firstop> <mulDivArith> | ε

54) <parentArith> ==> TK_OP <arithmetic Expressions> TK_CL | TK_ID | TK_NUM | TK_RNUM

55) <firstOP> ==> TK_MUL | TK_DIV

56) <secondOP> ==> TK_PLUS | TK_MINUS

1) First (<program>) = {TK_FUNID, TK_MAIN}

2) First(<main function>) = {TK_MAIN}

3) First(<other Functions>) = {TK_FUNID, ε}

4) First (<function>) = {TK_FUNID}

5) First (<input_par>) = {TK_INPUT}

6) First (<output_par>) = {TK_OUTPUT, ε}

7) First (<parameter list>) = {TK_INT, TK_REAL, TK_RECORD, TK_UNION}

8) First (<data Type>) = {TK_INT, TK_REAL, TK_RECORD, TK_UNION}

9) First (<primitive Datatype>) = {TK_INT, TK_REAL}

10) First (<constructed Datatype>) = {TK_RECORD, TK_UNION}

11) First(<remaining List>) = {TK_comma, ε}

12) First (<stmts>) = {TK_RECORD, TK_UNION, TK_TYPE, TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN} TK_DEFINETYPE

13) First (<typeDefinitions>) = {TK_RECORD, TK_UNION, ε}

14) First (<typeDefinition>) = {TK_RECORD, TK_UNION}

~~15) First (<type definition>) = {TK_RECORD, TK_UNION}~~

16) First (<field definitions>) = {TK_TYPE}

17) First (<field definition>) = {TK_TYPE}

18) First (<more Fields>) = {TK_TYPE, ε}

19) First (<declarations>) = {TK_TYPE, ε}

20) First (<declaration>) = {TK_TYPE}

21) First (<global-or-not>) = {TK_COLON, ε}

22) First (<other stmts>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, ε}

23) First (<stmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL}.

extra RUID in txt file

24) First(<assignment Stmt>) = {TK_ID}

25) First(<singleOrRecId>) = {TK_ID}

26) First(<singleOrRecNum>) = {TK_NUM, TK_RNUM, TK_ID}

27) First(<singleOrRecPrime>) = {ε, TK_DOT}

not present

28) First(<funCallStmt>) = {TK_SQL, TK_CALL}

29) First(<outputParameters>) = {TK_SQL, ε}

30) First(<inputParameters>) = {TK_SQL}

31) First(<iterativeStmt>) = {TK_WHILE}

32) First(<conditionalStmt>) = {TK_IF}

33) First(<conditionalPrime>) = {TK_ELSE, TK_ENDIF}

elsePart === conditionalPrime

34) First(<ioStmt>) = {TK_READ, TK_WRITE}

35) First(<arithmeticExpression>) = {TK_OP, TK_ID, TK_NUM, TK_RNUM}

36) ~~First(<arithmeticPrime>) = {TK_ID, TK_NUM, TK_RNUM}~~

37) ~~First(<operator>) = {TK_PLUS, TK_MUL, TK_MINUS, TK_DIV}~~

38) First(<booleanExpression>) = {TK_OP, TK_NOT, ~~TK_NOT~~, TK_ID, TK_NUM, TK_RNUM}

39) First(<booleanExtension>) = {ε, TK_AND, TK_OR}

40) First(<booleanPrime>) = {TK_ID, TK_NUM, TK_RNUM}

Not Present

41) First(<var>) = {TK_ID, TK_NUM, TK_RNUM}

42) First(<logicalOp>) = {TK_AND, TK_OR}

43) First(<relationalOp>) = {TK_LT, TK_LE, TK_EQ, TK_GT, TK_GE, TK_NE}

44) First(<returnStmt>) = {TK_RETURN}

45.) First (<optionalReturn>) = { TK_SQL, ε }

46.) First (<idList>) = { TK_ID }

47.) First (<more ids>) = { TK_COMMA , ε }

48.) First (<definetype stmts>) = { ε, TK_DEFINETYPE }

49.) First (<definetype stmt>) = { TK_DEFINETYPE }

50.) First (<A>) = { TK_RECORD, TK_UNION }

51.) First(^<mulDivArith>) = { TK_OP, TK_ID , TK_NUM, TK_RNUM }

52.) First(^<addOrSub>) = { ε, TK_PLUS, TK_MINUS }

53.) First(^<mulOrDiv>) = { ε, TK_MUL, TK_DIV }

54.) First (<parentArith>) = { TK_OP, TK_ID , TK_NUM, TK_RNUM }

55.) First (<firstOp>) = { TK_MUL, TK_DIV }

56.) First (<secondOp>) = { TK_PLUS, TK_MINUS }

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_ID$, TK_COLON}

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_TYPE, ~~TK_UNION~~, TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_DEFINETYPE, TK_RETURN}

14) Follow (<typeDefinition>) = {TK_RECORD, TK_UNION, TK_TYPE, TK_RETURN, TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_DEFINETYPE}

15) ~~Follow (<typeDefinition>)~~ = {~~TK_RECORD, TK_UNION, TK_TYPE, TK_RETURN, TK_ID, TK_WHILE, TK_IF~~, TK_READ, TK_WRITE, ~~TK_SQL, TK_CALL, TK_DEFINETYPE~~}

16) Follow (<fieldDefinitions>) = {TK_ENDRECORD, TK_ENDUNION ~~TK_END~~}

17) Follow (<fieldDefinition>) = {TK_TYPE, TK_ENDRECORD, TK_ENDUNION}

18) Follow (<moreFields>) = {TK_ENDRECORD, TK_ENDUNION}

19) Follow (<declarations>) = {~~TK_DEFINETYPE~~, TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, ~~TK_RETURN~~}, TK_RETURN}

20) Follow (<declaration>) = {TK_TYPE, TK_RETURN, TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, ~~TK_DEFINETYPE~~}

21) Follow (<global-or-not>) = {TK_SEM}

22) Follow (<otherStmts>) = {TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

23) Follow (<stmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

24) Follow (<assignment stmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

25) Follow (<SingleOrRecId>) = {TK_ASSIGNOP, TK_CL}

26) Follow (<SingleOrRecNum>) = {TK_CL}

27) Follow (<SingleOrRecPrime>) = {TK_ASSIGNOP, TK_CL}

28) Follow (<fun call stmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

29) Follow (<outputparameters>) = {TK_CALL}

30) Follow (<inputparameters>) = {TK_SEM}

31) Follow (<IterativeStmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

32) Follow (<conditional stmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

33) Follow (<conditionalPrime>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

34) Follow (<iostmt>) = {TK_ID, TK_WHILE, TK_IF, TK_READ, TK_WRITE, TK_SQL, TK_CALL, TK_RETURN, TK_ENDWHILE, TK_ELSE, TK_ENDIF}

35) Follow (<arithmeticExpression>) = {TK_CL, TK_SEM}

~~36) Follow (<arithmeticPrime>) = {TK_PLUS, TK_MUL, TK_MINUS, TK_DIV}~~

~~37) Follow (<operator>) = {TK_OP, TK_ID, TK_NUM, TK_RNUM}~~

37) Follow (<booleanExpression>) = {TK_CL}

38) Follow (<

39) Follow (<booleanExtension>) = { TK_CL }

40) Follow (<booleanPrime>) = { TK_CL }

41) ~~Follow (<var>) = { TK_PLUS, TK_MUL, TK_MINUS, TK_DIV, TK_CL, TK_SEM, TK_ID, TK_NUM, TK_RNUM }~~

42) Follow (<logicalOp>) = { TK_OP }

43) Follow (<relationalOp>) = { TK_ID, TK_NUM, TK_RNUM }

44) Follow (<returnStmt>) = { TK_END }

45) Follow (<optionalReturn>) = { TK_SEM }

46) Follow (<idList>) = { TK_SQR }

47) Follow (<moreIds>) = { TK_SQR }

48) Follow (<defineTypeStmts>) = { TK_RETURN, TK_DEFINETYPE } TK_TYPE, TK_ID, TK_IF, TK_CALL, TK_SQL, TK_WHILE, TK_READ, TK_WRITE }

49) Follow (<defineTypeStmt>) = { TK_DEFINETYPE, TK_RETURN, TK_TYPE, TK_ID, TK_IF, TK_CALL, TK_SQL, TK_WHILE, TK_READ, TK_WRITE }.

50) Follow (<A>) = { TK_RUID }

51) Follow (<addOrSub>) = { TK_CL, TK_SEM }

52) Follow (<mulDivArith>) = { TK_PLUS, TK_MINUS, TK_CL, TK_SEM }

53) Follow (<mulOrDiv>) = { TK_PLUS, TK_MINUS, TK_CL, TK_SEM }

54) Follow (<parenthArith>) = { TK_MUL, TK_DIV, TK_PLUS, TK_MINUS, TK_CL, TK_SEM }

55) Follow (<firstOp>) = { TK_OP, TK_ID, TK_NUM, TK_RNUM }

56) Follow (<secondOp>) = { TK_OP, TK_ID, TK_NUM, TK_RNUM }

57) Follow (<var>) = { TK_LT, TK_LE, TK_EQ, TK_GT, TK_GE, TK_NE, TK_CL }