# **CPSC 254 Lab 5 Group 2-3 Test Cases**

## R28 <Primary>

## R27 <Factor>

## R26.1 <Term'>

## R26 <Term>

## R25 <Expression>

## R25.1 <Expression'>

## R24 <Relop>

## R23 <Condition>

## R22 <While>

## R21 <Scan>

## R20 <Print>

## R19 <Return>

## R18 <If>

* The output from the <If> rule is either a “if (<Condition>) <Statement> fi” or “if (<Condition>) <Statement> else <Statement> fi”
* For the “if (<Condition>) <Statement> fi” result:
  + Input: reading in a text file that contains a complete and valid “if” statement.
  + Expected output: for the procedure\_If() function to return true if the “if” statement contains a condition and at least one statement.
* For the “if (<Condition>) <Statement> else <Statement> fi” result:
  + Input: reading in a text file that contains a complete and valid “if else” statement.
  + Expected output: for the procedure\_If() function to return true if the “if else” statement has at least 2 statements, one for the “if” and another for the “else”
* These test cases should be sufficient since it tests the “if” statement on its own and as well as testing it if it is followed by an “else” statement.

## R17 <Assign>

## R16 <Compound>

## R15 <Statement>

## R14 <Statement List>

## R13 <IDs>

## R12 <Declaration>

* The output from the <Declaration> rule are <Qualifier> and <IDs>
* For the <Qualifier> and <IDs> result:
  + Input: reading in a text file that contains a qualifier (a keyword applied to a data type) such as “const int” and IDs with valid and unique names like num1, car\_model, account\_3.
  + Expected output: for the procedure\_Declaration function to return true. The vector should contain tokens for a valid qualifier and accepted ID names.
* These test cases should be sufficient since it tests applying a keyword to a data type to create a qualified data type. Also, testing a variety of formats for valid ID names with a combination of letters, numbers, and using underscores.

## R11 <Declaration List>

## R10 <Opt Declaration List>

## R9 <Body>

## R8 <Qualifier>

## R7 <Parameter>

* <Parameter> outputs an <ID> or a <Qualifier>
* In order to test the <ID> result:
  + We create a token object that holds “IDs” and we update the lexeme with some name “a,b,c”
  + It’s than later pushed back to a token vector all\_tokens
* In order to test the <Qualifier> result:
  + The same logic from before applies, we create a token object that holds “Qualifier” this time, and we update the lexeme with the string “const”
  + It’s also pushed back onto all\_tokens
* The function then tests the results by running procedure\_Parameter using the all\_tokens vector.
* The result is than checked conditionally, if it returned successfully, we get a Test Passed, or Test failed if failed to return properly. This tells us whether the function works properly.

## R6 <Parameter List>

## R5 <Opt Parameter List>

* The output from the <Opt Parameter List> rule is either a <Parameter List> or <Empty>
* For the <Empty> result:
  + Input: reading in an empty text file. Or providing an empty vector of tokens.
  + Expected output: for the procedure\_Opt\_Parameter\_List function to return true and for there to be no tokens/ an empty vector.
* For the <Parameter List> result:
  + Input: reading in a text file that has example parameters such as (int num1, float sum) or generating tokens for the vector in the function.
  + Expected output: for the procedure\_Opt\_Paramter\_List function to return true. The tokens in the vector would contain the variables’ data type and name.
* These test cases should be sufficient since it covers both outputs from empty to a list of parameters.

## R4 <Function>

## R3 <Function Definitions>

## R2 <Opt Function Definitions>

## R1 <Rat20F>