Exercise10: ANTLR

1 Objectives

Using ANTLR, you will create a Java based lexer and **parser** that reads in an expression from the command line and outputs the value of that expression.

ANTLR reference can be found at http://www.antlr.org and

https://theantlrguy.atlassian.net/wiki/display/ANTLR4/ANTLR+4+Documentation

The steps are:

1) Create/Edit a ".g4" file which consists of LEXER and PARSER rules.
2) Use ANTLR on the ".g4" file to automatically GENERATE java codes for the lexer and parser.
3) Use the Parser as you see fit. It will return a PARSE TREE. You can traverse it to analyze it or to generate CODE.

We have several EXAMPLES that you should first view and tryout and understand. That will give you a good idea of the format of the ".g4" files.

The structure of the ".g4" files can be found at
https://theantlrguy.atlassian.net/wiki/display/ANTLR4/Grammar+Structure

2 Warm Up: Try Some Examples

1. First, open blackboard, go to Course Contents, and then download exercise10.zip file into your workspace (U:\workspace or something like that!). Then, unzip.
2. Go to folder ex0
3. There are three files .in ,.g4 and TestGrammer.java . .in is the file that contains the input, .g4 has the grammar rules.
4. Read the HelloWorld.g4 file and try and understand it.
5. Now type **java -jar ../antlr-4.4-complete.jar HelloWorld.g4**
   This will automatically generate lexer and parser in java!
6. Now, compile all the java files by typing "javac *.java"
7. You can check the parser/lexer by typing
   java org.antlr.v4.runtime.misc.TestRig HelloWorld start
   The program waits for you to type in input. You can type "Hello World!" and the ^D to indicate end of file OR you could have redirected "HelloWorld.in" to the program
   java org.antlr.v4.runtime.misc.TestRig HelloWorld start < HelloWorld.in
   Note the syntax errors! That happens because we have a "W" instead of "w". Fix that in the input. You may still get one error because of an extra new line (ignore that one).
8. You can view the parse tree by typing (note the –gui option)
   java org.antlr.v4.runtime.misc.TestRig HelloWorld start -gui
9. Now, take a look at the TestGrammer.java program to see how the lexer and parser are invoked from a program to get access to the parse tree!
10. Run TestGrammer by typing
    java TestGrammer HelloWorld.in HelloWorld.g4 start
11. Now – make sure you understand all the steps!
12. **Go to the NEXT exercise and try out all the steps (steps4-steps10).**

NOTE1: Try and fix syntax errors by fixing the input files

NOTE2: Always first try using TestGrammer, and then TestRig – make sure to use the –gui option.

3 SIMPLE CALCULATOR USING ANTLR

3.1 Reverse Polish Notation or Postfix Notation.

http://en.wikipedia.org/wiki/Reverse_Polish_notation

Write antlr grammer rules (i.e. create a ".g4" file) that accepts multiple expressions written in Reverse Polish Notation. Your grammer must support numerical operations i.e. +, -, *,/, and %, and also logical operations i.e. &&, ||, ! and relational operations (<, <=, ==, !=, >, >=). Skip white space.

You can assume your language will accept only INT and BOOLEAN types. Also, assume that all operations (except !) have two operands.

Your parser will also have to **EVALUATE** each statement. Statements are to separated by ";".

A RPN.g4 file has been provided which you can use to start your solution.

Example of input file:

10 20 + 30 * ; 10 20 <= 13 20 >= && true || ;

Your output:

Aside from checking for syntax errors, it should also print

900;true;

4 Submission:

Zip your files and submit on black board. Remember there is only one submission per group. Make sure to include all the files that are needed in order to run your program.