CS330 – Programming Languages

HW4 - ANTLR Expression Parser (Expr.g4)

Name Stuart Lech, Preston Mosley, Mohammad Khan Grade \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This assignment is designed to get you started experimenting with ANTLR parser generator in windows environment. You will create and experiment with the simple calculator grammar, described in chapter 4 of the ANTLR 4 book. You will then extend the expression grammar to include rules for an ^ operator.

The following (created by Steven, updated by Yu), will guide you through the process of (a) installing python, (b) installing ANTLR4 and testing with antlr4 and antlr4-parse. You can find additional discussions and explanation in the ANTLR4 book. The grammar is also discussed in chapter 4, *it is important that you read through the first 3 chapters before starting your assignment.*

----- Beginning of installation directions ----

(Reference: <https://github.com/antlr/antlr4/blob/master/doc/getting-started.md>)

**Starting with Python NOT installed**

1. Install python
   1. Go to python.org
   2. Downloads
   3. Download the latest version and follow instructions
   4. Choose “Disable path length limit”
2. Add python and scripts to system variables (some folders in the following paths are hidden)
   1. Open “Edit the environment and system variables”
   2. Select “Environmental Variables”
   3. Under “System variables”, select the “PATH” variable and click “Edit”
   4. Select “New” then browse to the “Python311” folder. The standard install location is “%USERPROFILE%\AppData\Local\Programs\Python”
   5. Select “New” then browse to the “Scripts” folder. The standard install location is “%USERPROFILE%\AppData\Local\Programs\Python\Python311”
   6. Any command windows will need to be closed and reopened so the changes will take effect.
3. Continue to Starting with Python installed

**Starting with Python installed**

1. Open windows “Command Prompt”
2. Browse to the folder you intend to use for your ANTLR folder you will be using for files.
3. Input the following command “pip install antlr4-tools”
4. Input the following command “antlr4”
5. Type “y” to install Java
6. Create file “Expr.g4” in the ANTLR folder with the following text
7. Try steps 8-18 to verify that ANTLR4 is properly installed and working.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Input “antlr4-parse Expr.g4 prog -tree” press “Enter”
2. Input “(A+B\*(413-Yu)+W/2” press “Enter”
3. Input ctrl+z “^Z” press “Enter” // ctrl-z indicate end of file
4. Output should match the example above
5. Input “antlr4-parse Expr.g4 prog -tokens -trace ” press “Enter”
6. Input “10+20\*30” press “Enter”
7. Input ctrl+z “^Z” press “Enter”
8. Output should match the example above
9. Input “antlr4-parse Expr.g4 prog -tokens -trace ” press “Enter”
10. Input “10+20\*30” press “Enter”
11. Input ctrl+z “^Z” press “Enter”

Now we are ready to continue to Java portion of the homework.

**Continue with Java**

1. Install Java SE Development Kit
   1. Go to java.com
   2. Click on the “Java SE Development Kit”
   3. About halfway down the page select the “Java 19” tab, this should be preselected
   4. Just under that select “Windows”
   5. Select the “X64 Installer”, “https://download.oracle.com/java/19/latest/jdk-19\_windows-x64\_bin.exe”
   6. Once the file is downloaded, run it to install
2. Add Java to system variables
   1. Open “Edit the environment and system variables”
   2. Select “Environmental Variables”
   3. Under “System variables”, select the “PATH” variable and click “Edit”
   4. Select “New” then browse to the “bin” folder. The standard install location is “C:\Program Files\Java\ jdk-19\bin” press “Ok” until out of edit variables
   5. Any command windows will need to be closed and reopened so the changes will take effect.
3. Download java .jar file
   1. Download “antlr-4.13.1-complete.jar” and place it in the ANTLR working folder
4. Add CLASSPATH to system variables
   1. Open “Edit the environment and system variables”
   2. Select “Environmental Variables”
   3. Under “User variables”, click “New”
   4. Input “CLASSPATH” for variable name
   5. Input “.;C:\Users\Administrator\antlr\antlr-4.13.1-complete.jar;” for variable value. (your path may be different)
   6. Press “Ok” until out of edit variables
   7. Any command windows will need to be closed and reopened so the changes will take effect
5. Create “Expr.g4” file in ANTLR folder
6. Create “ExprJoyRide.java” file in ANTLR folder
7. Input “antlr4 Expr.g4” to generate java files
8. Input “javac \*.java” to compile java files
9. Input “java ExprJoyRide.java” to run. There will be a couple of warnings and no prompt will be up.
10. Input “X=A+B\*(3-C/2)” and press “Enter”
11. Input “Y=X\*2” and press “Enter”
12. Input ctrl+z “^Z” and press “Enter”
13. The output should resemble the following

Text

Description automatically generated

1. Create “grun.bat” file in ANTLR folder
2. Input “grun Expr prog -gui” and press “Enter”
3. Input “W=Y+3\*X” and press “Enter”
4. Input “Y=W-X-2\*Z” and press “Enter”
5. Input crtl+z “^Z” and press “Enter”
6. The output should resemble the following

A picture containing graphical user interface

Description automatically generated

After getting everything installed and tested, you are now ready to add an extra operator to the Expr.g4 grammar and complete the homework, as described below:

1. Extend the Expr grammar to add definition for exponent operator ^. Note that ^ should be defined as right associative and have precedence higher than \* and /. Also note that starting in 4.2, <assoc=right> option must be specified at the beginning of a rule option. For example, if I were to make \* right associative, I would use

<assoc=right> expr ‘\*’ expr

1. Antlr4-parse to test your new Expr.g4 and generate a lispy tree with the following input (step 8 of starting with Python installed). Take screenshot and add it to your homework solution file.

Stuart= SIUE-B-3\*C

Lech= Spring\*2024+B-2^2^3

1. Go through steps 5-9 from the java section again to generate and compile java code and test your new grammar with both ExprJoyRide and grun. Use the following input for testing. Take screen shots of both DOS command window and GUI window (the one with the trees) and include them in your homework.

Stuart= SIUE-B-3\*C

Lech= Spring\*24+B-2^2^3

**Homework questions:**

Part 1 – Screenshots of activities

1. Screenshot of command window from step 2 (25 points)

A screenshot of a computer

Description automatically generated

1. Screen shot of command window from step 3 containing ExprJoyRide and grun tests. (30 points)

A black background with white lines

Description automatically generated

Part 2 - Answer the following questions as related to ANTLR Expr.g4 and expression grammar as shown below:

expr: expr ('\*'|'/') expr

| expr ('+'|'-') expr

| INT

| ID

| '(' expr ')'

;

Even though this expression grammar is left-ambiguous (i.e., impossible for LL(1) parsers), ANTLR4 is still able to process this grammar to create an unambiguous parser with correct precedence of \* over +.

1. Describe how ANTLR4 determines the operator precedence regarding ‘\*’ and ‘/’ vs. ‘+’ and ‘–‘. (15 points)

In ANTLR grammars, the precedence of operators is determined by their order in the grammar file, from top to bottom. Operators that appear earlier in the file are given higher precedence over those that appear later. Additionally, when operators are grouped together on the same line, like in the expression expr ('\*'|'/') expr, they share the same level of precedence. This structuring method allows the grammar to define the priority of operations without explicitly specifying precedence levels.

1. Describe how ANTLR4 determines the associativity of operators. (15 points)

ANTLR, by default, associates binary operators to the left. For situations where right associativity is required, such as for an exponentiation operator ^, it must be explicitly declared. However, the syntax for declaring right associativity, including the use and placement of <assoc=right>, can vary based on the version of ANTLR being used. In some versions, this may involve specific annotations or directives placed directly in the grammar rule definitions to control the associativity of operations.

1. Describe how ANTLR4 handles Left-recursive rule. (15 points)

ANTLR4 addresses left recursion by converting it into a form that uses repetition ((...) \*) to evaluate the precedence between operators. This approach allows ANTLR4 to process grammars with direct left recursion efficiently. However, it's important to note that ANTLR4 does not support indirect left recursion, which means that grammars requiring this form of recursion need to be refactored or restructured to fit within the direct left recursion capabilities of ANTLR4.

This assignment is worth 100 points and is due **10:00 PM on Monday (3/11)**. Submit your solution file on Moodle. Late submission will be accepted until 10:00 PM Wednesday (3/13). Standard late penalty applies. For this assignment, **you may work in a group of 2-3 people**, in which case only one doc needs to be submitted. Be sure to include the names of all group members. Note that even though this homework is due after the midterm term, you are strongly encouraged to read chapters 1-3 of ANTLR4 Reference book and complete the assignment (or at least make significant progress) before the exam as this material is included in the midterm Exam.