

San José State University  
Department of Computer Science

# CS 153

## Concepts of Compiler Design

Section 1  
Fall 2020  
Instructor: Ron Mak

### Assignment #4

**Assigned:** Thursday, September 17  
**Due:** Thursday, September 29 at 8:30 AM  
**Team assignment**, 120 points max

#### Pcl interpreter using ANTLR

The purpose of this assignment is to give you practice writing a parser and an interpreter for Pcl, a subset of Pascal. In effect, you will recreate the interpreter you wrote for Assignment #3, but this time you will use the ANTLR-generated lexer, parser, and backend `visit` methods.

The zip file [Asgn04Java.zip](#) contains the `Pc14.g4` grammar file to which you must add production rules for the **WHILE**, **FOR**, **IF**, and **CASE** statements. Make any other modifications to the grammar file that you deem necessary. To confirm that your grammar is correct, use either the ANTLR plug-in or `rrd-antlr4-0.1.2.jar` on the command line (<https://github.com/bkiers/rrd-antlr4>) to generate syntax diagrams.

Generate a graphical parse tree for the test source file `TestCase.txt` using either the ANTLR plug-in or the command-line `grun` Java program. The grammar file and the graphical parse tree together tell you what child nodes each tree node has.

The skeleton `Executor` class demonstrates how the `visit` methods can use the context objects (the `ctx` parameters) to access children of the parse tree nodes. To complete the `Executor` class, you can use code from your solution to Assignment #3 or from the [suggested solution](#). Look at the `Executor` class in the expression interpreter example [ExprLabeledJava.zip](#) for examples of `visit` methods. Note that the parse tree created by the ANTLR-generated parser has a different structure than the one created by our hand-coded parser.

After you've completed the `Pc14.g4` grammar file and the you've written the `visit` methods of the `Executor` class, you should be able to execute all the sample source files from Assignment #3. The runtime output should be the same as in that assignment.

## What to submit to Canvas

A zip file that contains:

- All of your Java source files and any additional input test programs you wrote.
- Your `Pc14.g4` grammar file.
- A PDF of your ANTLR-generated syntax diagrams (will require more than one page to fit).
- A PDF of your ANTLR-generated graphical parse tree from the `TestCase.txt` source file (will require more than one page to fit).
- Cut-and-paste text files of the runtime output from the following test source files from Assignment #3: `HelloWorld.txt`, `TestWhile.txt`, `TestIf.txt`, `TextFor.txt`, and `TestCase.txt`.

Submit to **Assignment #4: Pcl Interpreter using ANTLR**

## Rubric

Your submission will be graded according to these criteria:

Criteria	Max points
<b>Pcl4 grammar that includes:</b>	<b>30</b>
<ul style="list-style-type: none"><li>• WHILE statement</li><li>• IF statement</li><li>• FOR statement</li><li>• CASE statement</li><li>• syntax diagrams</li><li>• parse tree for <code>TestCase.txt</code></li></ul>	<ul style="list-style-type: none"><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li></ul>
<b>visit methods for:</b>	<b>65</b>
<ul style="list-style-type: none"><li>• assignment</li><li>• REPEAT</li><li>• WHILE</li><li>• IF</li><li>• FOR</li><li>• CASE</li><li>• expression</li><li>• simple expression</li><li>• term</li><li>• parenthesized expression</li><li>• variable</li><li>• integer constant</li><li>• real constant</li></ul>	<ul style="list-style-type: none"><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li></ul>
<b>Execution output from:</b>	<b>25</b>
<ul style="list-style-type: none"><li>• <code>HelloWorld.txt</code></li><li>• <code>TestWhile.txt</code></li><li>• <code>TestIf.txt</code></li><li>• <code>TextFor.txt</code></li><li>• <code>TestCase.txt</code></li></ul>	<ul style="list-style-type: none"><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li><li>• 5</li></ul>