COMS 4115: Programming Languages and Translators
Project Deliverable #2: Scanner and Parser

Due February 21st, 2018 at 11:59:00 PM

Richard Townsend, Columbia University

1 Background

This deliverable constitutes the syntax analysis portion of your compiler: the scanner and parser. Specifically, you will provide the necessary files to generate a scanner and parser for your language (as it stands), a toplevel file to execute them, and some simple tests showing that they work on portions of your language. Although these components may change over the rest of the semester (due to feedback, overly ambitious components that need to be changed or removed, etc.), you should have a general idea of the syntax of your language by the deadline of this deliverable.

2 Content Requirements

Your submission should include the following:

1. An OCamllex input file (.mll) specifying how to tokenize a stream of input characters.
2. An OCamlyacc input file (.mly) defining the tokens of your language, their precedence and associativity, and how to parse these tokens into syntactically valid phrases.
3. An OCaml file (.ml) defining the data types for an AST in your language.
4. A toplevel OCaml file (.ml) to drive the scanner and parser on some input, pretty-printing the resulting AST.
5. A directory of at least 10 small tests: 5 positive tests that should produce valid ASTs, 5 negative tests that should be rejected due to invalid syntax
6. A single script to execute your toplevel program against your tests, validating them against the expected output from your toplevel program. **This script should be written in Python or Bash.**
7. A README describing how to compile and execute your compiler (as it stands), how to run your test script, and any syntax that you still need to add to your compiler.

3 Rules and Regulations

- Submissions must be made via CourseWorks.
- Each group should submit a single .zip file containing all the files listed in Section 2. If your group name is 'XYZ', the file should be called **XYZ.zip**.
- All group members' names and email addresses should be included in the README file.

4 Grading

To be graded, your submission must follow all the rules and regulations listed above in Section 3. Your code must compile without errors or warnings to receive full credit.

In the following rubric, $n \leq 4$ is the number of code files in your submission that compile without errors or warnings, and $pos \leq 5$ and $neg \leq 5$ are how many positive and negative tests, respectively, correctly pass through your scanner and parser (i.e. your toplevel pretty-prints an AST for each positive test, and produces an error or exception otherwise).

Criteria	Total	Description
Function	60 pts	15 pts $\times$ n
Test Cases	20 pts	2 pts $\times$ $(pos + neg)$
Test Script	15 pts	Full marks if your script validates all tests correctly
README	5 pts	Full marks if we can compile your programs and run your test script following your README instructions