Ollscoil Teicneolaíochta na Mumhan
Munster Technological University

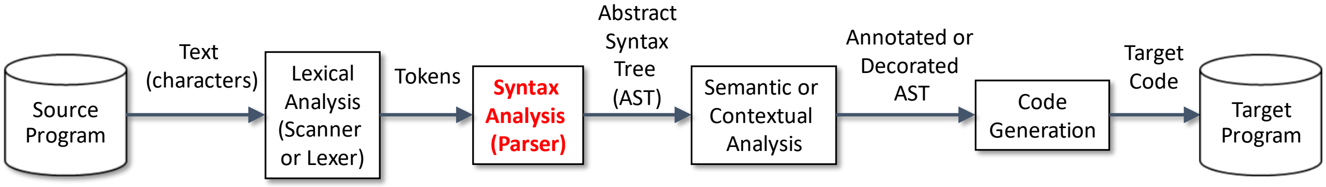
Programming Language Design
Department of Computer Science
Dr. Francisco Ortin
francisco.ortin@mtu.ie

Laboratory 4 – Syntax Recognition

The objective of this laboratory is to implement a syntax recognizer for C--. A syntax recognizer is a program that returns whether an input file fulfills the syntax rules of a given language.

Recall, the syntax analysis phase:

a) Recognizes the language syntax (*this lab*), and
b) Returns the corresponding AST (following lab).


Source Program
Text (characters)
Lexical Analysis (Scanner or Lexer)
Tokens
Syntax Analysis (Parser)
Abstract Syntax Tree (AST)
Semantic or Contextual Analysis
Annotated or Decorated AST
Code Generation
Target Code
Target Program

Instructions

The language recognizer should be specified in ANTLR. The generated parser will syntactically recognize the C-- language. If no syntax error exists for the input program, the main program will exit showing no error message. Otherwise, the parser will show an error indicating the line and column numbers (this is automatically done by the parser generated by ANTLR).

The following files are provided:

- Input.txt: An example C-- input file.
- Main.java: Main program that receives an input file name and calls the syntax recognizer generated by ANTLR.
- TestRigGUI.java: Main program that parsers the "input.txt" program and displays its parser tree using ANTLR test rig. This program is for debugging purposes. If ANTLR is not working as expected, the parse tree is valuable information to show how the grammar productions were derived.
  Notice: This functionality is included in the ANTLR plugin for IntelliJ, by right-clicking over the left-hand side of a production and selecting "Test Rule program". If you are using IntelliJ and its ANTLR plugin, you will not need to use TestRig.
- TestRigTrace.java: Main program that parsers the "input.txt" program in trace mode. It shows how the grammar productions are derived (one-step left-most derivations).
- TestRigTokens.java: Main program that parsers the "input.txt" program and shows the collection of tokens recognized. This is useful to check whether the lexer is working as expected.

Go ahead

Create a lab04 project and include the files provided (above), together with your lexer specification and implementation from the previous lab. Then, extend the Cmm.g4 file to implement your syntax analyzer, and use input.txt to test it.

Notice: the C-- language is specified in the language-specification.pdf file (lab02).