CS3020/CS6240: Mini-Programming Assignment #1
Study of Existing PL Lexers and Parsers
Due Wednesday, October 7th, 2015 at 11:59 PM

Introduction This assignment asks to you do a *programming language (PL) theoretic* and *compiler-theoretic* study of existing languages and compilers with a focus on implementation of lexical analysis and parser phases. You will have to choose at least two among the existing implementations and write a technical report focusing on various aspects of the programming language and the particular compiler/translator/interpreter implementation. Some suggestions are:

- C/Cxx compilers like GCC (before 2.9.5 and most recent) and LLVM;
- Java compilers (many of them);
- Flex, Bison, ANTLR tools;
- R, Python/Perl, OCaml, Ruby interpreters;
- Html, Javascript interpreters.

You do not have to limit yourself to the above list and you can choose any *reasonable* interesting large-enough implementation. The two different implementations you choose should be from at least two different sources. Some suggestions are to take at least one tool/interpreter (similar to the tools/interpreters listed above) or, a comparative study of two different implementations of the same language, or focus on the functional programming languages.

You can do this along with your team member with at most two members per team. It is suggested (not mandatory) that you team up with the same partner as the one for your Cool compiler project phases.

Submission This study is aimed at increasing your code reading skills that are essential for any compiler designer/writer. Hence, you are *not* expected to write any code for completing this assignment. You will however have to submit a *technical report* on your study. The report has to focus on the PL basics and particularly focus on the lexical analysis and parser phases. The typical questions that I expect to be answered in the report are the following:

- What are the specific features of this PL that make writing a compiler/translator/interpreter interesting?
- Why was the particular scanner-generator/parser-generator chosen?
- What is special about the particular lexical-analysis/parsing phases? How were the conflicts resolved?
- ...

Evaluation Your report should show a *good* understanding of the implementation. The usual rules of plagiarism apply to your report. In particular **do not** directly copy the material from manuals/websites. Your report should professionally refer the website(s) from where you downloaded the code and the manual(s) you referred to. Very good reports will fetch bonus points.