Compiler Theory: Assignment 1

Marc Moreno Maza
University of Western Ontario
CS447b – Winter 2004

Posted: Tuesday, January 27, 2004
Due: Tuesday, February 17, 2004

1 Guidelines

Date of postage. Tuesday, January 27, 2004.

Normal date for submission. Tuesday, February 17, 2004 (no penalties).

Last date for submission. Sunday, February 22, 2004 (late penalties apply from Wednesday 18).

Due format. The assignment should be returned to the instructor by email. Your submission will have the form of a UNIX *archive* created with the UNIX command `tar`. This archive will have a structure similar to those of the example archives available on the course web site. More precisely, a `src` directory will contain your `flex` (or `lex`) source programs, a `test` directory will contain your test files, a `Makefile` will define the targets and a `README` will explain these targets. Moreover, your answers to the assignment questions (that are not computer programs) will be made in a file

- which may be a text file, or a `postscript` file or a `pdf` file (No other formats are allowed. Moreover, I recommend that you generate your postscript and pdf files under a recent UNIX/Linux distribution.)

- and which will be placed in a `doc` directory.

In addition, you could have a `bin` directory for binaries (as in the course example archives) and a `ps` directory for postscript versions of computer programs (this is convenient to print them!). The computer programs of this assignment are all input files for the `flex` (or `lex`) program, and C program or header files. The preferred version of `flex/lex` is `flex version 2.5.4`.

Your `Makefile` must work under Linux (with a recent distribution) or under a UNIX system of the GAUL network. You can reuse the Makefile of the archive `lex.tar.gz` from the course web site. Note that this Makefile was developed with `bash` under Red Hat 7.3 (Linux) and using GCC version 2.96.

Your `README` must specify which UNIX system you are using (Linux or SunOS) and which shell (command language interpreter) you are using (`tcsh` or `bash`). For the compilation options of `flex/lex` please refer to the Makefile of the archive `lex.tar.gz`.

Late penalties. See the course outlines.

Collaboration. You are expected to do this assignment on your own without assistance from anyone else in the class.

Marking. This assignment will be marked out of 40. The allocation of marks is shared equally between the four activities described below.

2 The Assignment

The objective is to write a lexer for the ALLCOT language. Please, refer to the document titled *The* ALLCOT *Language* and available on the course web site.

2.1 A Grammar for ALLCOT

Following the specifications for ALLCOT, write a grammar for this language. You may point out any inconsistencies in the language design and propose adjustments. Note that the goal here is not only to build a bigger (or even better) language, but to fix this one. So your grammar may address your concerns, and thoroughly describe any proposed syntactic or semantic changes.

2.2 Arrays for Allcot

Propose the addition of arrays to the grammar of Allcot. Please refer to the Aldor documentation for the syntax. Describe the semantics of your array definition. Provide a new grammar which includes a definition of arrays.

2.3 A Lexer for Allcot

Write a lexical analyzer for your version of Allcot using `lex` or `flex`. Your lexer should recognize the tokens and print error messages for the strings that are not valid Allcot lexemes. You will be using your lexer in future assignments, so design your lexer with this in mind.

2.4 Test Programs for Allcot

Write three Allcot test programs, each of them being about 100 lines long. Make sure that they cover all the syntax features of the language and that your lexer can process them correctly.