# Compiler Theory: Assignment 3

**Marc Moreno Maza**
**University of Western Ontario**
CS447a – Fall 2004

Posted: Tuesday, November 16, 2004
Due: Wednesday, December 8, 2004

## 1 Guidelines

**Date of postage.** Tuesday, November 16, 2004.

**Normal date for submission.** Wednesday, December 8, 2004 (no penalties).

**Last date for submission.** Monday, December 13, 2004 (no late penalties).

**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/lex` and `yacc` 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 gen-

erate 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/lex` and `yacc` program, together with 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 `yacc.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` and `yacc` please refer to the Makefile of the archive `yacc.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 35 for the compiler and 5 for the test programs.

# 2 The Assignment

The objective is to write a compiler for the ALLCOT language, with C as the target language. Please, refer to the document titled *The* ALLCOT *Language* and to the last year assignment 3 and available on the course web site.

- This document discusses the semantics of the ALLCOT language. Also, it suggests some simplifications to the language in order to keep things simple.

- **For this assignment**
  - implementing inheritance between domains is **not required** but inheritance between categories is mandatory,
  - the only required buil-in types are `Integer` and `String`;
- The assignment 3 of 2003 discusses translation to C for the MOOL language. The remarks and hints there apply here, to theAllcot language.

## 2.1 Bonus

- If your compiler produces 3 address code for the sequence of statements you will get **two extra marks** on the final mark.
- In addition, if your compiler saves on temporaries as described in Section 5 of the Chapter on Intermediate Code Generatetion, you will get **two additional extra marks** on the final mark.

## 2.2 A compiler for Allcot

Using `flex/lex` and `yacc`, write a compiler for your version of Allcot.

- From every valid input Allcot file, your compiler must generate a valid and corresponding C file.
- From every non-valid input Allcot file, your compiler must report an error with a friendly error message.

## 2.3 Test Programs for Allcot

Update the test files of your previous assignment according to your version of Allcot. Therefore, you must have three *significant* test files that are valid input Allcot file. You must also have some small invalid input Allcot files, say about 5. **You must check that the generated C files compile and run properly.**

## 2.4 Marking scheme

```
-----------------------------------------
--% CS447a - Fall 2004 -  Assignment 3 %--
-----------------------------------------

Student ID:

Total Mark:


A compiler for Allcot   ( / 35 points)
--------------------

o Ratio of success in compiling valid Allcot programs:

o Ratio of success in rejecting non-valid Allcot programs:

o Ratio of success in the correctness of the generated C:

o Ratio of success in running the compiled C programs:

o Accuracy of error Messages:


Test programs for Allcot   ( / 5 points)
------------------------

o Language construct covered by valid Allcot programs:

o Language construct covered by non-valid Allcot programs:
```