Homework #3: CMPT-379

Distributed on Thu, Sep 29; due on Thu, Oct 6

Anoop Sarkar – anoop@cs.sfu.ca

You will need to refer to the **Decaf** language definition for this homework.

(1) Write a **Decaf** program that computes the Catalan number `catalan(n)` for any fixed integer `n` and prints the value using the `print_int` library function. Submit the program as the file `catalan.decaf`

(2) Implement the tokenizer for **Decaf** using the your code from previous homeworks. Submit a program that implements the approach written in C++ (or in ANSI C) that can be run as follows:

```
lexan decafTokens.txt program.decaf
```

where `decafTokens.txt` is a specification of tokens in **Decaf** using finite-state machines based on your implementation from Homework #2 and `program.decaf` is a filename which contains a **Decaf** program. The token names should be the terminal symbols in the CFG of Question 3 below. Follow the usual conditions for submission of your code that were necessary in previous homeworks. Make sure that all supplementary files are included in your submission and that they are in the right location for your program to work.

For questions about the token definitions refer to the **Decaf** language definition. For questions about the output format of your tokenizer, refer to the sample **Decaf** program and the sample output of the tokenizer provided to you in the location mentioned on the course web page.

(3) Write down a context-free grammar for the structure of **Decaf** programs based on the reference grammar in the **Decaf** language definition (make sure that the non-terminal and terminal symbols used in the CFG correspond as much as possible to the symbols used in the reference grammar). Submit a file called `decafGrammar.txt` which contains the CFG in the following text format: For the CFG: $\langle \text{start} \rangle \rightarrow A \langle \text{start} \rangle B \mid \epsilon$ the text format you should use is:

```
start A start B
start
```

Follow the convention that the non-terminals in this text format are written in the same format as identifiers in **Decaf** but are in lowercase (e.g. `start`, and for hyphenated non-terminals like *method-name* replace the hyphen with an underscore, e.g. `method_name`) and write the terminal symbols in the same format as identifiers but entirely in uppercase (e.g. `A`).

You should verify the correctness of your CFG either by examining it closely or you can verify aspects of the CFG by writing some simple code for checking whether non-terminals are used in the right-hand side of rules but not defined on the left-hand side anywhere else in the CFG, whether the terminal symbols are valid tokens, etc. You do not need to submit this testing code.