HKU COMP3235 Assignment 2

Due: Wednesday, March 29, 2017

Late penalty: -15% per day.

[[16.3.2017 - sas has been upgraded; please use "sas2" for Q2.]]

[[9.3.2017 - Assignment spec posted.]]

Q1. Postfix or prefix is better than infix as a notation for arithmetic expressions in that the former is always unambiguous (no issues with operator precedence or associativity).

a. Define a context-free grammar for a "postfix" integer calculator, with terminals integer (or number), +, -, *, /, and non-terminal E (expression).
b. Construct the LR(1) automaton (i.e., the configurating sets and their transitions) and generate the parsing table.
c. Convert from LR(1) to LALR(1); show the automaton and its parsing table.
d. Implement the corresponding calculator using flex and bison. Make sure you test it with enough postfix expressions as inputs. (If you find postfix expressions clumsy to type, you can easily convert the calculator demo'ed in class - the recursive descent one or the LR one - to generate a postfix expression from an infix input.)
e. Convert the grammar to an LL(1) grammar, if it is not already LL(1).
f. Implement the LL(1) grammar as a recursive-descent parser, hence calculator, that accepts postfix inputs.

Q2. Extend the c4 calculator language with the following two features. The result is c5. Implement both the interpretor (c5i) and the compiler (c5c).

a. BREAK and CONTINUE, whose semantics follow those of C. Use this little program to test.
b. Singly array: x[e] where x is one of a to z (the 26 registers of the stack machine, which can be treated as a piece of RAM of size 26), and e is an expression. Semantics: a[0] is actually a, a[1] is actually b, j[3] is actually m, and so on. As array index out of bound could be disastrous, so check for that in your implementation(s) (you may just print a message and exit the program when encountering such an error).
c. Write a bubble sort program (bubble.sc) to sort 10 input numbers into increasing order.