CS326 – Programming Languages, Concepts and Implementation

# Homework 4

# (Due March 19)

1. (24 pts) Translate the following expression into (a) postfix and (b) prefix notation:

   ```
   (b + sqrt(b×b – 4×a×c))/(2×a)
   ```

2. (26 pts) Some languages (e.g., Algol 68) do not employ short-circuit evaluation for Boolean expressions. However, in such languages an `if…then…else` construct (which only evaluates the arm that is needed) can be used as an expression that returns a value. Show how to use `if…then…else` to achieve the effect of short-circuit evaluation for `A and B` and for `A or B`.

3. (24 pts) Consider a midtest loop, here written in C, that processes all lines in the input until a blank line is found:

   ```
   for ( ; ; )
   {
      line = read_line();
      if (all_blanks(line)) break;
      process_line(line);
   }
   ```

   Show how you might accomplish the same task in C using a (a) `while` and (b) `do` loop, if `break` instructions were not available.

4. (26 pts) Write a *tail-recursive* function in Scheme to compute *n* factorial (n! = 1×2×…×*n*). You will probably want to define a "helper" function, as discussed in the textbook.

5. (Extra Credit - 10 pts) Give an example in C in which an in-line subroutine may be significantly faster than a functionally equivalent macro. Give another example in which the macro is likely to be faster. Hint: think about applicative versus normal-order evaluation of arguments.