CSE 3302 Lab Assignment 1

Due September 21, 2012

Goal:

Understanding of elementary scanning.

Requirements:

1. Design, code, and test a program to tokenize the lexical elements of a simple `C`-like language. Your second assignment will involve parsing this language. The output of your program will be the tokens, one per line, along with their classification (indicated in italics below).

   a. Input lines will have no more than 80 symbols.

   b. Comments are to be discarded. These will be under the usual `/* . . . */` and `//` conventions. End-of-lines and spaces are to be ignored as "whitespace".

   c. A *name* consists of no more than 10 lower-case alphabetic characters.

   d. A *pic* (positive integer constant) is a positive (decimal) value no larger than `1000000`.

   e. A *bool* is either `T` or `F`.

   f. A *numop* is any of the binary operations `+`, `-`, `*`, or `/`.

   g. A *compop* is any of the binary operations `<`, `>`, `==`, `!=`, `<=`, or `>=`.

   h. A *boolop* is any of the operations `&&`, `||`, or `!` (unary).

   i. A *condop* is a `?`.

   j. A *condsep* is a `:`.

   k. A *lparen* is a `(`.

   l. A *rparen* is a `)`.

2. Email your program to `mehra.nourozborazjany@mavs.uta.edu` by 12:45 p.m. on September 21.

Getting Started:

1. You may use any language you wish, but please check with the TA or me if you plan to use anything "exotic". If you use JavaScript, then you may use an input box for pasting the input file along with an output box.

2. You are allowed to use available code, but stay within the boundaries of this assignment. Be sure to provide appropriate attribution.

3. Tokens should be handled greedily, e.g. scanning "<=" will never be a separate < and =.

4. Your program should stop immediately upon detecting an error. Just give a brief message including the most recent symbol. In a few cases, an error could be detected either in your scanner or by the parser.

5. The use of regular expressions is encouraged. An example for JavaScript is on the webpage.