

Tokenizing: Lex has been used extensively for lexical scanning, the first phase of compiling. In this phase the input program is tokenized or broken up into tokens belonging to programming categories like variables and integers. You will also use lex for other categories, some of which are useful for web applications, for example (simplified) web addresses (URLs) and HTML tags. Yet others, like phone numbers, are useful in text-processing.

Assignment: Write a Lex program that recognizes the *Lex Expressions* specified below and carries out the corresponding *Actions*. Create a test file that provides a thorough test of your program. Some test data should come close to your patterns without satisfying them. Explain how your program achieves its goals and how your inputs provide a thorough test.

Submitting: The online syllabus for your section will have instructions on *How to submit the Lex project* just as it did for Prolog.

Lex Expressions: Write a Lex program to recognize the string sets specified below. Make your patterns as short as possible, using as many Lex definitions as you need, to help shorten the patterns. Lex permits a definition to make use of an earlier definition. Take advantage of this to make your definitions no longer than necessary.

- variable astring of letters and digits beginning with a letter (capital or lowercase)
- integer a string of one or more digits, optionally preceded by a plus or minus sign
- real an integer (as above) followed immediately by (i) a decimal point and one or more digits or (ii) an “E” and an integer (as above) or both:(i) then (ii).
- newline the end-of-line character
- whitespace a string of blanks and tabs, but no newlines (whitespace within a line)
- phone number a 3-digit area code, parenthesized and/or followed by hyphen, and then 7 more digits with a hyphen before the 4th from last.
- email address a string of letters and underscores that starts with a letter, followed by an at-sign (“@”), then 2, 3 or 4 strings of letters separated by periods.
- web address optional “http://”, then 2, 3 or 4 strings of letters separated by periods, then optionally more material consisting of slash (“/”), an optional tilde (“~”), any number of strings of letters separated by slashes, and then a period (“.”) and “html”.
- dollar sign used to signal the end of input; use it only as the last character

Actions: The C or C++ code associated with the Lex expressions should produce output as specified below.

	<u>Pattern</u>	<u>Action</u>
1	variable	an at-sign (“@”), the variable and a newline
2	integer	a number-sign (“#”), the integer and a newline
3	real	two number-signs (“##”), the real and a newline
4	newline	2 newlines
5	whitespace	just a single blank
6	phone number	a percent sign (“%”), the phone number and a newline
7	email address	HTML code and a newline; e.g., xxx@yyy.zzz.org becomes xxx@yyy.zzz.org
8	web address	HTML code and a newline, e.g., http://www.zzz.org becomes http://www.zzz.org
9	dollar sign	used to signal the end of input; use it only as the last character