CS24210: Using lex

Edel Sherratt

1 November 2000

1. A first example
   (a) Look at the the file zippy.l on the CS24210 web pages. Look at the regular expression defining the token "zippy".
   (b) Download zippy.l, and use the command **lex zippy.l** to generate the file lex.yy.c from zippy.l.
   (c) Compile lex.yy.c using the command **gcc -o zippy lex.yy.c -ll** This will give you an executable program called "zippy" which implements the lex specification in zippy.l.
   (d) Try running zippy. Type some characters; then press return. Try typing the word 'zippy'. What happens? Type CTRL-D to finish.
   (e) What happens if you input anything other than 'zippy'?
2. Extending zippy.l
   (a) Add some more rules to zippy.l so that it recognises more words; for example, you could make it recognise 'slow', 'medium' etc.
   (b) Run lex on your new file, and compile and execute it as before.
3. Recognising integers
   (a) Download the file numbers.l from the CS24210 web pages. Study the regular definition for "digit", and a regular expression that describes integers consisting of one or more digits.
   (b) Run lex on this numbers.l, compile lex.yy.c, and try running the resulting scanner.
   (c) Now try extending numbers.l so that it also recognises floating point numbers.
4. Converting uppercase to lowercase.
   (a) Download the file upper2lower.l from the CS24210 web pages. This file contains a lex program to convert uppercase letters to lowercase.
   (b) Use lex to process this file; then compile it and try running the resulting program.
   (c) See if you can create a lex program to convert lowercase letters to uppercase.
5. Try writing a lex program to convert dates like
   14 FEB 99
   to a form like
   14 / Feb / 99