3

17/sci01/051

Nasir Firdaus Opeyemi

LOOP AND SWITCH SCANNERS

There are two primary methods for implementing a scanner. The first is a program that is hard-coded to perform the scanning tasks. The second uses regular expression and finite automata theory to model the scanning process.

A "loop & switch" implementation consists of a main loop that reads characters one by one from the input file and uses a switch statement to process the character just read.

The output is a list of tokens and lexemes from the source program. The following program fragment shows a skeletal implementation of a simple loop and switch scanner. The main program calls InitScanner and loops calling ScanOneToken until EOF. ScanOneToken reads the next character from the file and switches off that char to decide how to handle what is coming up next in the file. The return values from the scanner can be passed on to the parser in the next phase.

However, Loop-and-switch scanners are sometimes called ad hoc scanners, (indicating their design and purpose of solving a specific instance rather a general problem).

For a sufficiently reasonable set of token types, a hand coded, loop and switch scanner might be all that’s needed.

it requires no other tools.

The gcc front-end uses an ad hoc scanner, in fact On the other hand, gcc’s C lexer is over 2,500 lines of code; verifying that such an amount of code is correct is much harder if your lexer does not see the extent of use that gcc’s front-end experiences.