CS4850 – SummerII 2006

Lexical Analysis and Lex (contd)

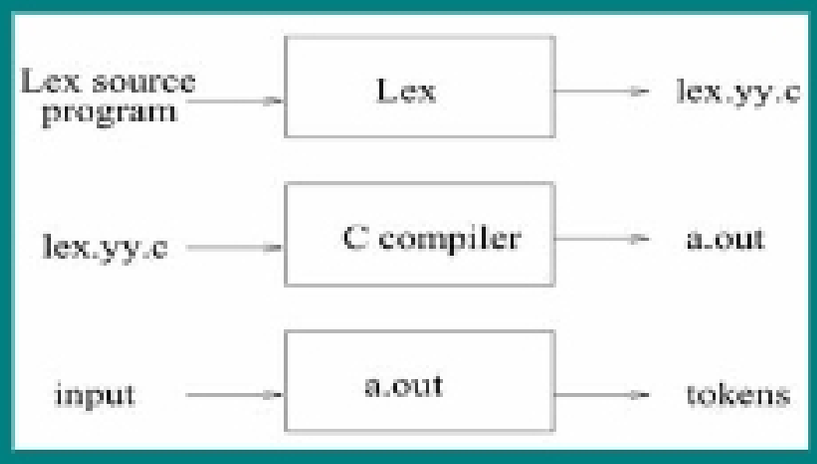
4.1

Lex Primer

- Lex is a tool for creating lexical analyzers.
- Lexical analyzers *tokenize* input streams.
- Tokens are the *terminals* of a language.
- Regular expressions define *tokens* .

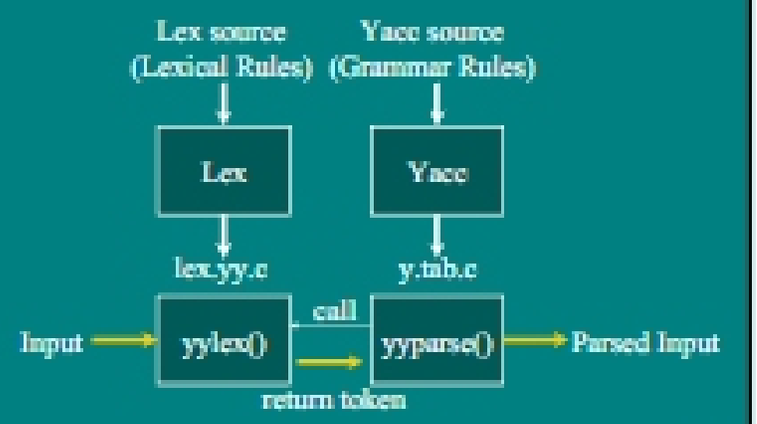
4.2

Usage Paradigm of Lex



4.3

To Use Lex and Yacc Together



4.4

Lex Internals Mechanism

- Converts regular expressions into DFAs.
- DFAs are implemented as table driven state machines.

4.5

Running Lex

- To run lex on a source file, use the command:
  lex source.l
- This produces the file lex.yy.c which is the C source for the lexical analyzer.
- To compile this, use:
  cc -o prog -O lex.yy.c -ll

4.7

Versions and Reference Books

- AT&T lex, GNU flex, and Win32 version
- lex & yacc ,2/e by John R.Levine, Tony Mason & Doug Brown, O'Reilly
- Mastering Regular Expressions, by Jeffrey E.F. Friedl, O'Reilly

4.8

General Format of Lex Source

```
%{
    C Declarations and includes
%}

<name> <regexp>
<name> <regexp>
...

%%

<regexp> <action>
<regexp> <action>
...

%%

User Subroutines (C code)
```

4.9

- Input specification file is in 3 parts
  - Declarations: Definitions
  - Rules: Token Descriptions and actions
  - Auxiliary Procedures: User-Written code
- Three parts are separated by %%
- Tips: ***The first part defines patterns, the third part defines actions, the second part puts together to express "If we see some pattern, then we do some action".***

4.10