Compiler Design lab

Lex

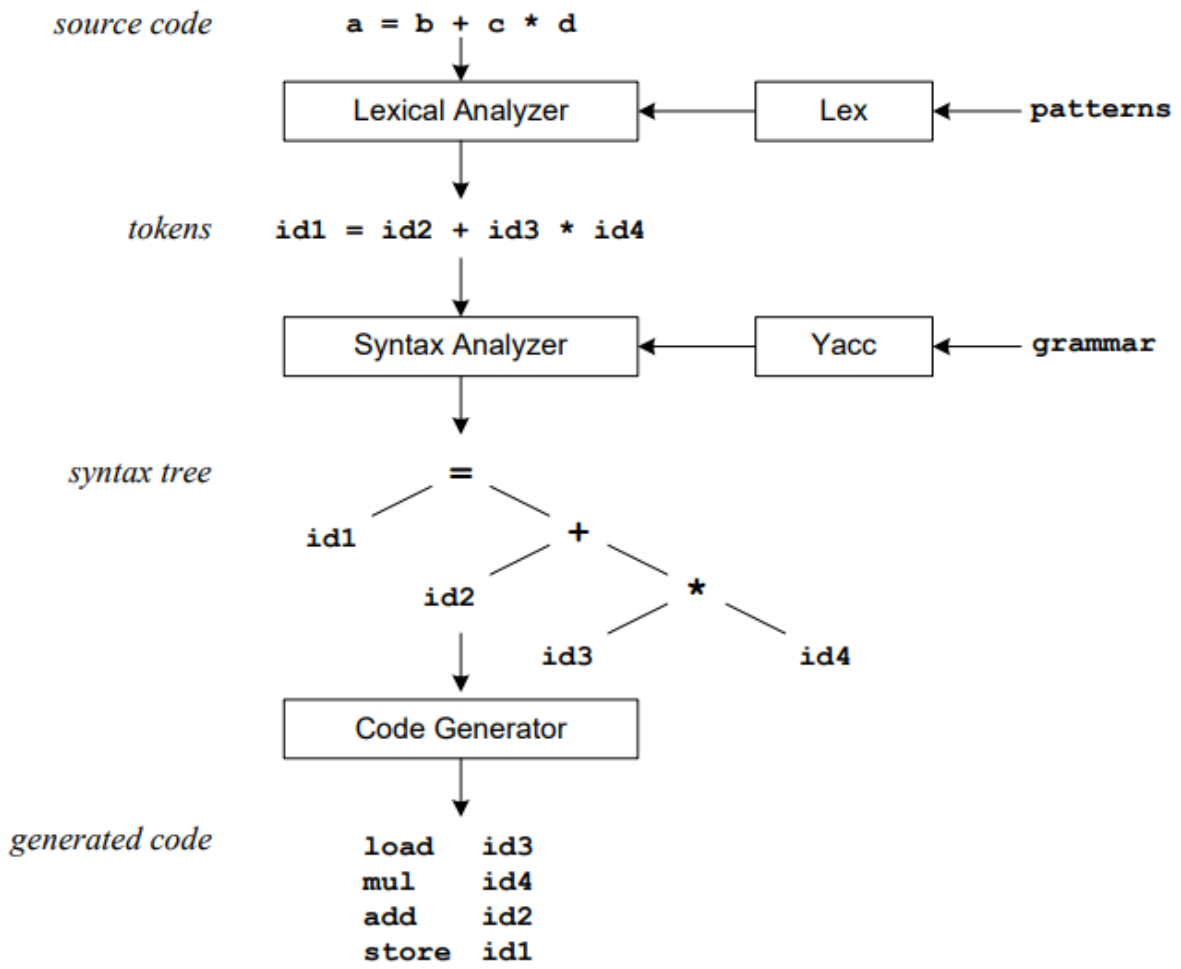
26/02/2022

Brief History

- Writing compiler was time consuming process
- By Lesk and Johnson in 1975
- Their utilities simplify compiler writing

Compilation sequence

- Input patterns/C code file to lex
- It generates C code (ll.yy.c)
- Lexical analyzer matches strings to input and converts it to tokens



Lex program

- 3 parts
  - Definition part
  - Pattern-Action part/Rule part
  - Subroutines

TEMPLATE

```
{%

Definitions

%}

%%

Rules

%%

Subroutines
```

Pattern-Action Part

Pattern Matching Primitives

Metacharacter	Matches
`.`	any character except newline
`\n`	newline
`*`	zero or more copies of the preceding expression
`+`	one or more copies of the preceding expression
`?`	zero or one copy of the preceding expression
`^`	beginning of line
`$`	end of line
`a|b`	`a` or `b`
`(ab)+`	one or more copies of `ab` (grouping)
`"a+b"`	literal "`a+b`" (C escapes still work)
`[]`	character class

Pattern-Action Part …contd.

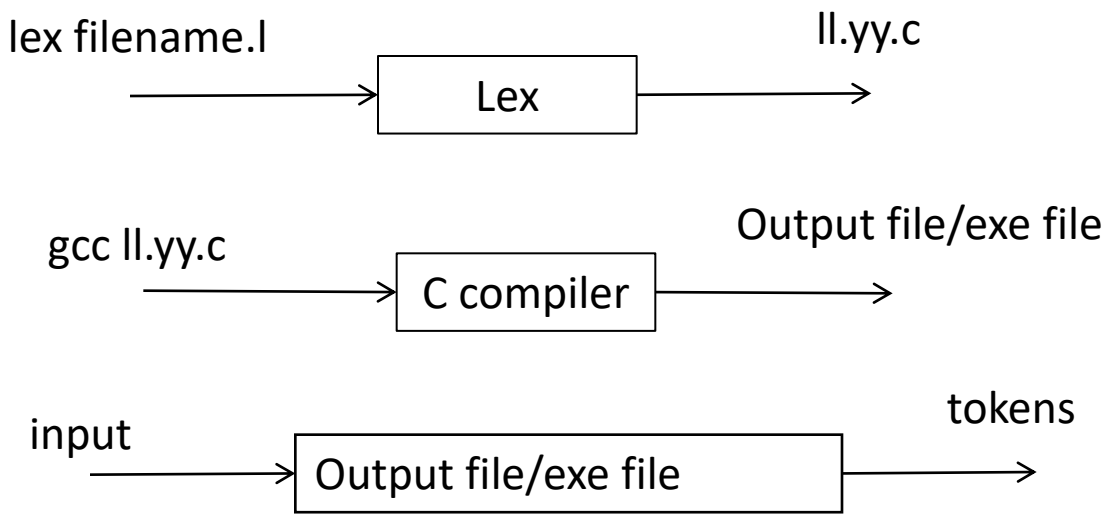
Pattern Matching Examples

Expression	Matches
`abc`	`abc`
`abc*`	`ab abc abcc abccc ...`
`abc+`	`abc abcc abccc ...`
`a(bc)+`	`abc abcbc abcbcbc ...`
`a(bc)?`	`a abc`
`[abc]`	one of: `a`, `b`, `c`
`[a-z]`	any letter, a-z
`[a\-z]`	one of: `a`, `-`, `z`
`[-az]`	one of: `-`, `a`, `z`
`[A-Za-z0-9]+`	one or more alphanumeric characters
`[ \t\n]+`	whitespace
`[^ab]`	anything except: `a`, `b`
`[a^b]`	one of: `a`, `^`, `b`
`[a|b]`	one of: `a`, `|`, `b`
`a|b`	one of: `a`, `b`

Lex predefined variables

Name	Function
`int yylex(void)`	call to invoke lexer, returns token
`char *yytext`	pointer to matched string
`yyleng`	length of matched string
`yylval`	value associated with token
`int yywrap(void)`	wrapup, return 1 if done, 0 if not done
`FILE *yyout`	output file
`FILE *yyin`	input file
`INITIAL`	initial start condition
`BEGIN`	condition switch start condition
`ECHO`	write matched string

Flow of lexical analyzer



Example code

```
digit [0-9]
letter [A-Za-z]
%{
        int count;
%}
%%
/* match identifier */
{letter}({letter}|{digit})*        count++;
%%
int main(void)
{
yylex();
printf("number of identifiers = %d\n", count);
return 0;
}
```

Install

For Ubuntu

sudo apt-get install flex

For Windows

https://simran2607.medium.com/compiler-design-using-flex-and-bison-in-windows-a9642ebd0a43

Run

For Ubuntu

- lex filename.l
- gcc ll.yy.c –o outputfilename
- ./a.out OR ./outputfilename

For Windows

- flex filename.l
- gcc ll.yy.c –o outputfilename
- a.exe OR outputfilename.exe

Demo

Thank you!