

CS 152 Project Phase 1 Lab Report

Fall 2018 - Apollo Truong 861168740, Sidney Son 861214496

DEVELOPMENT

This is the first phase of the class project.

In this phase we used the *flex* tool to generate a lexical analyzer for the high-level source code language, "MINI-L".

We used the in-class hand-out to define the following:

```
letter      [a-zA-Z]
digit       [0-9]
number      {digit}+
identifier  {letter}({letter}|{digit}|[_]({letter}|{digit}))*
```

We then defined the tokens that will be read and set `printf()` statements to print the respective token for each symbol read.

Problems we ran into while developing the lexical analyzer were mostly syntax errors causing the output to print unwanted tokens. For example, we had misnamed the identifier by including an extra parenthesis which caused output to not recognize identifiers and output the identifier itself. Other syntax errors caused our program to have an overflow error and would not run our program.

USAGE

The usage can be simplified to the command:

```
$make
```

Information on Makefile

The file we wrote, `mini_l.lex` is followed by the command

```
$flex mini_l.lex
```

This generates the file

```
lex.yy.c
```

With this file, we then enter the following command

```
$gcc -o lexer lex.yy.c -lfl
```

The generated "lexer" file is now able to convert the MINI-L (.min) program into a list of corresponding list of tokens.

EXAMPLE

```
$cat fibonacci.min | lexer
```

Running the lexer on the files: `fibonacci.min` `dowhiletest.min` `ifelseiftest.min` will give the following output:

| cmd | cmd | cmd | cmd |
|--|--|--|--|
| ssondp@DESKTOP-6D5AS585 : /mnt/c/Users/pnsdn | ssondp@DESKTOP-6D5AS585 : /mnt/c/Users/pnsdn | ssondp@DESKTOP-6D5AS585 : /mnt/c/Users/pnsdn | ssondp@DESKTOP-6D5AS585 : /mnt/c/Users/pnsdn |
| sson00@bolt \$ make | BEGIN LOCALS | sson00@bolt \$ make | sson00@bolt \$ cat ifelseitest.min lexer |
| flex mini.lex | IDENT n | flex mini.lex | IDENT m |
| gcc -o lexer lex.yy.c -lfl | COLON | gcc -o lexer lex.yy.c -lfl | IDENT LTE |
| ~/152/projectphase1 | INTEGER | ~/152/projectphase1 | IDENT NUMBER |
| sson00@bolt \$ cat fibonacci.min lexer | SEMICOLON | sson00@bolt \$ cat dowhiletest.min lexer | NUMBER 5 |
| FUNCTION | IDENT FIB_n | IDENT PROGRAM | IDENT SEMICOLON |
| IDENT Fibonacci | COLON | IDENT IDENT | IDENT WRITE |
| SEMICOLON | INTEGER | IDENT mytest | IDENT IDENT |
| BEGIN_PARAMS | SEMICOLON | IDENT SEMICOLON | IDENT n |
| IDENT k | END LOCALS | IDENT SEMICOLON | IDENT SEMICOLON |
| COLON | IDENT n | INTEGER | IDENT END_PROGRAM |
| INTEGER | READ | IDENT COMMA | SEMICOLON |
| SEMICOLON | IDENT n | IDENT IDENT | BEGIN_PROGRAM |
| END_PARAMS | SEMICOLON | IDENT m | READ |
| BEGIN_LOCALS | IDENT FIB_n | IDENT COLON | IDENT n |
| END_LOCALS | ASSIGN | IDENT INTEGER | SEMICOLON |
| BEGIN_BODY | IDENT Fibonacci | IDENT SEMICOLON | IF |
| IF | L_PAREN | IDENT BEGIN_PROGRAM | IDENT n |
| L_PAREN | IDENT n | IDENT READ | LTE |
| IDENT k | R_PAREN | IDENT IDENT | NUMBER 100 |
| LT | SEMICOLON | IDENT n | WRITE |
| NUMBER 1 | WRITE | IDENT SEMICOLON | IDENT n |
| R_PAREN | IDENT FIB_n | IDENT IDENT | SEMICOLON |
| THEN | SEMICOLON | IDENT m | ELIF |
| RETURN | END_BODY | IDENT ASSIGN | IDENT n |
| NUMBER 1 | ~/152/projectphase1 | IDENT IDENT | NUMBER 10 |
| ENDIF | sson00@bolt \$ | IDENT n | IDENT o |
| SEMICOLON | broadcast message from root | IDENT SEMICOLON | ASSIGN |
| RETURN | server will reboot for main | IDENT DO | IDENT n |
| IDENT Fibonacci | | IDENT BEGINLOOP | SUB |
| L_PAREN | | IDENT WRITE | NUMBER 2 |
| IDENT k | | IDENT IDENT | SEMICOLON |
| SUB | | IDENT n | WRITE |
| NUMBER 1 | | IDENT SEMICOLON | IDENT o |
| R_PAREN | | IDENT IDENT | SEMICOLON |
| ADD | | IDENT ASSIGN | ELIF |
| IDENT Fibonacci | | IDENT L_PAREN | IDENT n |
| L_PAREN | | IDENT IDENT | LT |
| IDENT k | | IDENT n | NUMBER 50 |
| SUB | | IDENT ADD | IDENT o |
| NUMBER 2 | | IDENT NUMBER | ASSIGN |
| R_PAREN | | NUMBER 1 | IDENT n |
| SEMICOLON | | IDENT R_PAREN | SUB |
| END_BODY | | IDENT SEMICOLON | NUMBER 5 |
| FUNCTION | | IDENT ENDLOOP | SEMICOLON |
| IDENT main | | IDENT WHILE | WRITE |
| SEMICOLON | | IDENT IDENT | IDENT o |
| BEGIN_PARAMS | | IDENT n | SEMICOLON |
| END_PARAMS | | IDENT SUB | |

```

IDENT o
SEMICOLON
ELSE
IDENT m
ASSIGN
IDENT n
SUB
NUMBER 1
SEMICOLON
WRITE
IDENT n
SEMICOLON
ENDIF
SEMICOLON
WRITE
IDENT o
SEMICOLON
END_PROGRAM
~/152/projectphase1
sson007@bolt $

```

Running the diff command on in bash gave us a complete match of the respective `.token` files. Our Project Phase 1 is successful.

