Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
TE C32
2103162

Experiment NO: 2

Aim: To study and implement programs LEX and YACC tool.

Theory:

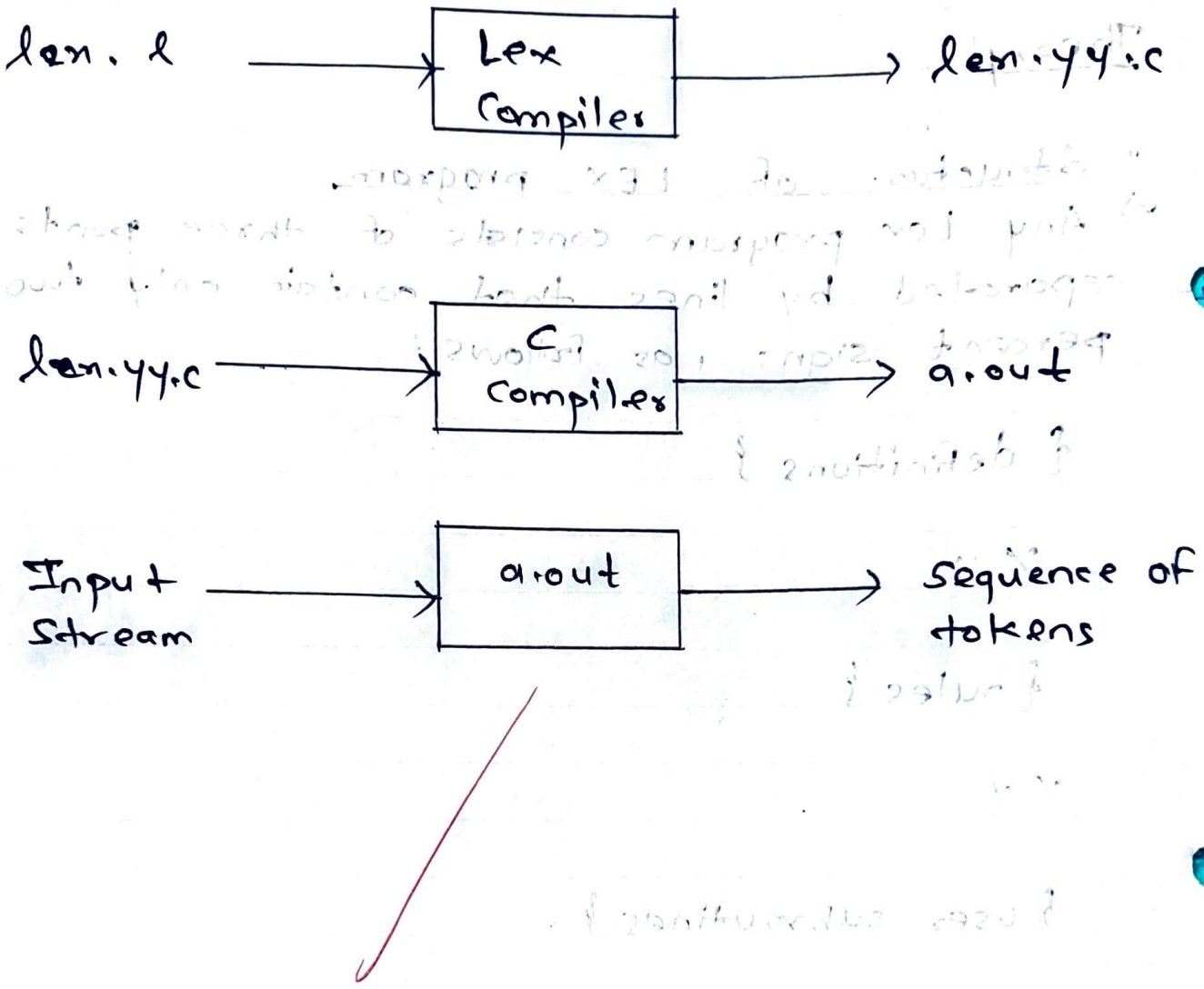
- **Structure of LEX program**

⇒ Any Lex program consists of three parts, separated by lines that contain only two percent signs, as follows:

```
{ definitions }

%%

{ rules }

%%

{ user subroutines }
```

- The definition section is the place to define macros and to import header files written in C.
- It includes variable declarations, constant declarations, and regular definitions.
- The rules section is the most important section; it associates patterns with C statements.

LEX Compiler :



THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

- The rule section contains the specification of token and the associated actions.
- This section is of form:

```
P1      {action 1}
P2      {action 2}
P3      {action 3}
.  .  .  .  .  .  .
Pn      {action n}
```

- Here, Pi is a regular expression.
- The user subroutines contains code called by the ~~until~~ rules in the rules section.

- Compilation steps:

1) flex program.l

2) gcc lex.yy.c -lfl

3) ./a.out

- Conclusion:

In this experiment, we learnt about the data structure of the lex program

QA
22/02/2024

Code-Output:

```
%{
/* Program to show the message when an ENTER key is pressed*/
%}

%%

[\n] {
        printf("\n\nHi.....Good Morning...\n");
        return;
        }

%%

main()
{
        yylex();
}
```

```
root@dellaio304: /home/complab304pc4/Desktop/163
root@dellaio304:/home/complab304pc4/Desktop/163# flex program4.l
root@dellaio304:/home/complab304pc4/Desktop/163# flex program1.l
root@dellaio304:/home/complab304pc4/Desktop/163# gcc lex.yy.c -lfl
program1.l: In function ‘yylex’:
program1.l:9:9: warning: ‘return’ with no value, in function returning non-void
    9 |         return;
      |         ^~~~~~
lex.yy.c:607:21: note: declared here
  607 | #define YY_DECL int yylex (void)
      |                     ^~~~~
lex.yy.c:627:1: note: in expansion of macro ‘YY_DECL’
  627 | YY_DECL
      | ^~~~~~~
program1.l: At top level:
program1.l:14:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
   14 | main()
      | ^~~~
root@dellaio304:/home/complab304pc4/Desktop/163# ./a.out



Hi.....Good Morning...
root@dellaio304:/home/complab304pc4/Desktop/163#
```

```
%{

char name[10];

%}

%%

[\n] {printf("\n Hi........%s......Good Morning\n",name); return 1;}

%%

void main()

{

char opt;

do {

printf("\nWhat is your name?"); scanf("%s",name);
yylex();
printf("\nPress y to continue"); scanf("%c",&opt);

}

while(opt=='y');
}
```

```
root@dellaio304: /home/complab304pc4/Desktop/163
root@dellaio304:/home/complab304pc4/Desktop/163# ./a.out

What is your name?Om

 Hi........Om......Good Morning

Press y to continue
```

```
%{
void display(int, char *);
int flag;
%}

%%
[a|e|i|o|u]+ {
        flag = 1;
        display(flag, yytext);
        }
.       {
                flag = 0;
                display(flag, yytext);
}

%%

void main()
{
        printf("\nEnter a character to check whether it is vowel or NOT\n");
        yylex();
}

void display(int flag, char *t)
{
        if(flag==1)
                printf("\nThe given character %s is a vowel\n", t);
        else
                printf("\nThe given character %s not is a vowel\n", t);
}
```

```
root@dellaio304: /home/complab304pc4/Desktop/163
root@dellaio304:/home/complab304pc4/Desktop/163# flex program3.l
root@dellaio304:/home/complab304pc4/Desktop/163# gcc lex.yy.c -lfl
root@dellaio304:/home/complab304pc4/Desktop/163# ./a.out

Enter a character to check whether it is vowel or NOT
i

The given character i is a vowel

k

The given character k not is a vowel
```

```
%{
void display(char[], int);
%}

%%
[a-zA-Z]+[\n] {
        int flag = 1;
        display(yytext, flag);
        return;
        }
[0-9]+[\n] {
        int flag = 0;
        display(yytext, flag);
        return;
}
.+ {
        int flag = -1;
        display(yytext, flag);
        return;
}


%%
void display(char string[], int flag)
{
        if(flag==1)
                printf("\nThe string\n\t%s\t\t is a word\n", string);
        else if(flag==0)
                printf("\nThe given string\"%d\"is a digit\n", atoi(string));
        else
                printf("\nThe given string is either a word or a digit\n");
}

main()
{
        printf("\nEnter a string to check whether it is word or digit\n");
        yylex();
}
```

```
root@dellaio304: /home/complab304pc4/Desktop/163
root@dellaio304:/home/complab304pc4/Desktop/163# ./a.out

Enter a string to check whether it is word or digit
564

The given string"564"is a digit
root@dellaio304:/home/complab304pc4/Desktop/163#
```

```
%{
#include<stdio.h>
%}

%%

[0-9]+          {printf("NUMBER: %s\n", yytext);}
[ ]             {printf("WHITE SPACE: %s\n", yytext);}
("n"|"t")       {printf("ESCAPE SEQUENCE: %s\n", yytext);}
[,;]            {printf("SEPERATOR: %s\n", yytext);}
[()]            {printf("PARANTHESIS: %s\n", yytext);}
[{}]            {printf("BRACKETS: %s\n", yytext);}
[-+/*%=]        {printf("OPERATORS: %s\n", yytext);}
("else"|"if"|"int"|"void"|"main"|"return"|"for"|"while") {printf("KEYWORDS: %s\n", yytext);}
[a-zA-z]+       {printf("IDENTIFIER: %s\n", yytext);}

%%

int main(){
        yylex();
        return 0;
}
```

```
root@dellaio304: /home/complab304pc4/Desktop/163
root@dellaio304:/home/complab304pc4/Desktop/163# ./a.out
(Om) {Shete} 0550
PARANTHESIS: (
IDENTIFIER: Om
PARANTHESIS: )
WHITE SPACE:
BRACKETS: {
IDENTIFIER: Shete
BRACKETS: }
WHITE SPACE:
NUMBER: 0550
```