**Exercise 3: Lex Programs**

Aim: To use Lex tools to perform simple addition, counting vowels, and letters and token in a given piece of code

Algorithm:

1. Start the program.

2. Lex program consists of three parts.

a. Declaration %%

b. Translation rules %%

c. Auxilary procedure.

3. The declaration section includes declaration of variables, main test, constants and regular

definitions.

4. Translation rule of lex program are statements of the form

a. P1 {action}

b. P2 {action}

c.…

d. …

e. Pn {action}

5. Write a program in the vi editor and save it with .l extension.

6. Compile the lex program with lex compiler to produce output file as lex.yy.c. eg $ lexfilename.l$ cc lex.yy.c –ll

7. Compile that file with C compiler and verify the output.

Sample Input-Ouput:

1.

a

enter the value of a:5

b

enter the value of b:4

c

The sum of 5 and 4 is 9

2.

hello;

vowels=2, consonants=3, digits=0

3.

2+r=p

Digit=1

Operator=2

Identifier=2

Code:

%{

#include<stdio.h>

int a,b,c;

%}

%%

"a" printf("\nEnter the value of a:");scanf("%d", &a);

"b" printf("\nEnter the value of b:");scanf("%d", &b);

"c" printf("\nThe sum of %d and %d is %d", a, b, c=a+b);return 0;

%%

int main()

{

yylex();

return 0;

}

int yywrap()

{

return 0;

}

%{

#include<stdio.h>

int vow=0,num=0,cons=0;

%}

%%

[aeiouAEIOU] vow++;

[0-9] num++;

[bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ] cons++;

";" printf("\nvowels=%d,cosonants=%d,digits=%d",vow,cons,num);

%%

int main()

{

yylex();

return 0;

}

int yywrap()

{

return 0;

}

%{

#include<stdio.h>

#include<string.h>

char str[20];

int opc=0,dc=0,lc=0,varc=0,idc=0;

%}

digit [0-9]

op [%|\*|+|-|=]

id [for|if|while|do|const|break|default]

%%

{digit} dc++;

{op} opc++;

{id} idc++;

"\n" printf("Digit=%d\nOPerator=%d\nIdentifier=%d",dc,opc,idc);

%%

int main()

{

yylex();

return 0;

}

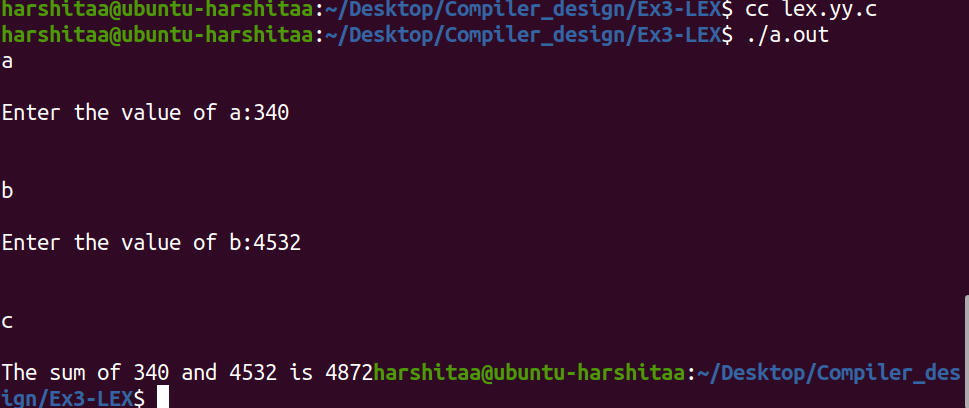
int yywrap()

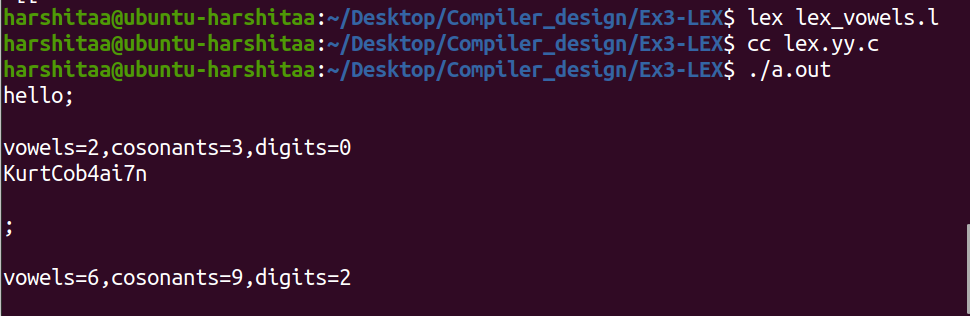
{

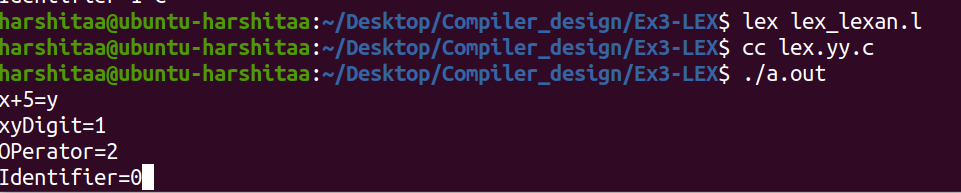
return 0;

}

Output:







Result:

Thus the lex programs are successfully executes.