**Experiment: 3**

**(a)Implement the lexical analyzer using JLex, flex or lex or other lexical analyzer generating stools.**

**Aim:** To write a LEX program to implement the lexical analyzer **(Syllabus Program)**

**Description:** Lexical Analysis is the first phase in compiler. In Lexical analysis the Code in HLL is broken and converted into small distinguishable parts called Lexemes, and these lexemes gives tokens which will be stored in Symbol table to be sent as input to the Syntax Analysis.

**Source Code:**

%{

#include<stdio.h>

char \*word[]={"keyword","identifier","operator","preprocessor","comment","invalid literal","reserved ", "number","string"};

void display (int);

%}

keyword "int"|"char"|"short"|"void"|"long"|"if"|"else"|"case"|"for"|"do"|"while"|"break"|"auto"|"static"|"const"|"enum"|"struct"

reserved "main"|"FILE"|"printf"|"scanf"|"puts"|"putc"|"getc"|"pow"

comments "//".\*|"/\*".\*"\*/"

operator "."|"{"|"}"|"("|")"|"["|"]"|"->"|"+"|"-"|"\*"|"/"|"="|"+="|;

preprocessor #.\*

string "\"".\*"\""

identifier [\_]\*[a-zA-Z][a-zA-Z0-9]\*

number [0-9]+[.]\*[0-9]\*

%%

{comments} { display(4);}

{preprocessor} { display(3);}

{reserved} { display(6);}

{keyword} { display(0); }

{operator} { display(2);}

{string} { display(8);}

{identifier} { display(1); }

{number} {display(7);}

[\n\t' '] {};

. {display(5); }

%%

void display(int n)

{

printf("\n%s --> %s\n",yytext,word[n]);

}

int yywrap()

{

return 1;

}

int main(int argc,char \*\*argv)

{

if (argc > 1)

{

yyin = fopen(argv[1],"r");

if(!yyin)

{

printf("could not open %s \n",argv[1]);

exit(0);

}

}

yylex();

return 0;

}

**Input file :** U

#include<stdio.h>

main()

{

printf(“Hello World!!!”);

}

**Output:**



**b) Program to count no of:**

**a) +ve and –ve integers**

**b) +ve and –ve fractions using LEX**

%{

#include<stdio.h>

int posint=0, negint=0,posfraction=0, negfraction=0;

%}

%%

[-][0-9]+ {negint++;}

[+]?[0-9]+ {posint++;}

[+]?[0-9]\*\.[0-9]+ {posfraction++;}

[-][0-9]\*\.[0-9]+ {negfraction++;}

%%

int yywrap()

{

return 1;

}

main(int argc, char \*argv[])

{

if(argc!=2)

{

printf("Usage: <./a.out> <sourcefile>\n");

exit(0);

}

yyin=fopen(argv[1],"r");

yylex();

printf("No of +ve integers=%d\n No of –ve integers=%d\n No of +ve fractions=%d\n No of –ve fractions=%d\n", posint, negint,posfraction, negfraction);

}

**Input file :** input1.txt

2

4

-6

-7

0

0

-2

-6

-2

0

0

8

9

10

78

90

-4

23.5

12.5

-12.7

-2.5

**Output:**

****

4**.(a). Program to count the number of vowels and consonants in a given string.**

%{

#include<stdio.h>

int vowels=0;

int cons=0;

%}

%%

[aeiouAEIOU] {vowels++;}

[a-zA-Z] {cons++;}

%%

int yywrap()

{

return 1;

}

main()

{

printf("Enter the string.. at end press ^d\n");

yylex();

printf("No of vowels=%d\nNo of consonants=%d\n",vowels,cons);

}

**Output:**

****

**(b).Program to count the number of characters, words, spaces, end of lines in a given input file.**

%{

#include<stdio.h>

int c=0,w=0,s=0,l=0;

%}

WORD [^ \t\n,\.:]+

EOL [\n]

BLANK [ ]

%%

{WORD} {w++; c=c+yyleng;}

{BLANK} {s++;}

{EOL} {l++;}

. {c++;}

%%

int yywrap()

{

return 1;

}

main(int argc, char \*argv[])

{

if(argc!=2)

{

printf("Usage: <./a.out> <sourcefile>\n");

exit(0);

}

yyin=fopen(argv[1],"r");

yylex();

printf("No of characters=%d\nNo of words=%d\nNo of spaces=%d\n No of lines=%d",c,w,s,l);

}

**Input file :** input.txt

Prasad V Potluri Siddhartha Institute of Technology

II B.tech CSE Section-1 Students

Compiler Design Lab

Simple Lex programs

**Output:**

****